

CHUYÊN ĐỀ 1. TÌM HIỂU MỘT VÀI KIỂU DỮ LIỆU TUYẾN TÍNH

BÀI 1. MÔ HÌNH DỮ LIỆU NGĂN XẾP VÀ HÀNG ĐỢI

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Biết được mô hình dữ liệu ngăn xếp và hàng đợi.
- Hiểu được cơ chế hoạt động LIFO của ngăn xếp.
- Hiểu được cơ chế hoạt động FIFO của hàng đợi.

2. Năng lực

Năng lực chung:

- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học, biết lắng nghe và trả lời nội dung trong bài học.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.

Năng lực Tin học:

- Mô tả được mô hình dữ liệu ngăn xếp và hàng đợi thông qua cơ chế hoạt động của các kiểu dữ liệu này.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trung thực: Thực hiện đúng phần việc của bản thân và hợp tác làm việc nhóm khi được giao nhiệm vụ. Có ý thức báo cáo kết quả một cách chính xác.
- Trách nhiệm: Hoàn thành các bài tập theo yêu cầu của GV thông qua hệ thống câu hỏi, phiếu học tập, thông qua sản phẩm.

II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Tài liệu, Máy tính, Máy trình chiếu
- HS: SCD, vở ghi, máy tính, đọc trước bài 1. Mô hình dữ liệu ngăn xếp và hàng đợi

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động khởi động

- Mục tiêu: HS làm quen với những hiện tượng, sự vật trong cuộc sống có liên quan đến mô hình ngăn xếp và hàng đợi. Tạo hứng thú học tập cho HS.
- Nội dung: Quan sát và trả lời Phiếu học tập số 1.

PHIẾU HỌC TẬP SỐ 1

Em hãy quan sát các hình ảnh về đồ vật và hiện tượng trong hình 1.a, 1.b và trả lời 2 câu



hỏi sau:

Trong chồng đĩa, các đĩa đánh số từ 1 đến 5, từ dưới lên trên hãy cho biết đĩa nào được xếp vào trước và đĩa nào được xếp vào sau? Ai sẽ là người được rút tiền trước tại cây ATM? Người xếp hàng cuối cùng sẽ được rút tiền khi nào?

a) Chồng đĩa b) Xếp hàng rút tiền tại ATM

c) Sản phẩm: Trả lời câu hỏi trong phiếu học tập số 1

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, mỗi nhóm thảo luận trong thời gian 2 phút và ghi kết quả trả lời ra giấy.

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ	GV yêu cầu HS quan sát và hoàn thành phiếu học tập số 1.
Thực hiện nhiệm vụ	HS tiếp nhận nhiệm vụ, thảo luận nhóm hoàn thành phiếu học tập số 1.
Báo cáo, thảo luận	GV: Thu kết quả trả lời của từng nhóm, cho HS quan sát kết quả các nhóm và nhận xét.
Kết luận, nhận định	<ul style="list-style-type: none"> - GV nhận xét câu trả lời của các nhóm. + Với hình a. Chồng đĩa ta thấy ngay đĩa nào cho vào trước thì sẽ lấy ra sau cùng, đĩa nào cho vào sau cùng sẽ lấy ra đầu tiên. Đây chính là cơ chế hoạt động của mô hình dữ liệu ngăn xếp (stack). + Với hình b Ai xếp hàng trước sẽ được rút tiền trước, ai xếp sau sẽ rút tiền sau theo đúng thứ tự, Đây chính là cơ chế hoạt động của mô hình hàng đợi (queue). Hai mô hình này sẽ được tìm hiểu trong bài học ngày hôm nay.

2. Hoạt động hình thành kiến thức

Hoạt động 1: Tìm hiểu mô hình dữ liệu ngăn xếp

a) Mục tiêu: Hiểu được cơ chế hoạt động của ngăn xếp, các thao tác chính và ý nghĩa của các thao tác này.

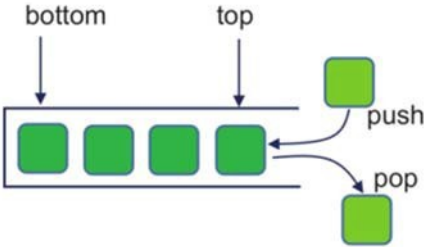
b) Nội dung hoạt động: Trả lời phiếu học tập số 2, câu hỏi củng cố.

PHIẾU HỌC TẬP SỐ 2: Hãy ghép mỗi mục ở cột A với một mục ở cột B cho phù hợp.	
Cột A	Cột B
1. Ngăn xếp (stack) là gì? 2. Trình bày cơ chế hoạt động của mô hình dữ liệu ngăn xếp. 3. Thao tác đưa phần tử x vào đỉnh ngăn xếp S. 4. Tạo một ngăn xếp rỗng. 5. Lấy ra một phần tử từ đỉnh của ngăn xếp S và trả về phần tử này. 6. Trả về phần tử tại vị trí đỉnh của ngăn xếp S, S không thay đổi. 7. Kiểm tra ngăn xếp rỗng, trả về giá trị True nếu S rỗng.	a. push(S, x). b. pop(S). c. Là một dãy tuyến tính các phần tử dữ liệu. d. Thao tác đưa phần tử vào và lấy phần tử ra tại cùng một đầu của ngăn xếp. e. top(S). f. hoạt động theo cơ chế LIFO. g. S=Stack. h. isEmptyStack(S). i. thao tác đưa dữ liệu vào, lấy dữ liệu ra ở hai đầu.

c) Sản phẩm: HS hoàn thành được phiếu học tập số 2.

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, mỗi nhóm thảo luận thống nhất phương án trả lời.

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ 1	GV yêu cầu HS thảo luận nhóm để hoàn thành phiếu học tập số 2.
Thực hiện nhiệm vụ 1	HS tiếp nhận nhiệm vụ, nghiên cứu SCĐ thảo luận nhóm hoàn thành phiếu học tập số 2. GV quan sát HS tự học, thảo luận, trợ giúp kịp thời khi các em cần hỗ trợ, giải đáp.
Báo cáo, thảo luận	GV: Gọi đại diện các nhóm trình lên bảng viết đáp án của nhóm mình. GV yêu cầu các nhóm quan sát, nhận xét chấm điểm cho từng nhóm.

<p>Kết luận, nhận định</p>	<p>Phiếu học tập số 2: 1c, 2f, 3a, 4g, 5b, 6e, 7h.</p> <p>- GV nhận xét, chuẩn hoá kiến thức cho HS ghi nhớ. Mô hình dữ liệu ngăn xếp được mô tả như sau:</p>  <p>+ Là một dãy tuyến tính các phần tử dữ liệu.</p> <p>+ Có thao tác đưa phần tử vào, lấy phần tử ra tại cùng 1 đầu.</p> <p>+ Quy ước đỉnh (top) của ngăn xếp để đưa vào, lấy ra.</p> <p>+ Hoạt động theo cơ chế LIFO (vào sau ra trước).</p> <p>+ Các thao tác cơ bản trên dữ liệu ngăn xếp:</p> <ul style="list-style-type: none"> - Tạo một ngăn xếp rỗng: $S = \text{Stack}()$. - Đưa 1 phần tử x vào đỉnh ngăn xếp s: $\text{push}(S, x)$. - Lấy ra 1 phần tử từ đỉnh ngăn xếp: $\text{pop}(S)$. - Giá trị của phần tử đỉnh ngăn xếp: $\text{top}(S)$. - Kiểm tra ngăn xếp rỗng (trả về giá trị True) : $\text{isemptystack}(S)$. Ví dụ: $S = \text{Stack}()$. <p>$\text{push}(S, 2)$; $\text{push}(S, 1)$; $\text{push}(S, 5)$ kết quả trong S chứa 3 phần tử là 2, 1, 5.</p>
<p>Chuyển giao nhiệm vụ 2</p>	<p>Trả lời 2 câu hỏi củng cố trong sách trang 7.</p> <p>1. Muốn lấy phần tử ở đáy ngăn xếp phải làm thế nào?</p> <p>2. Cho S là 1 ngăn xếp rỗng. Em hãy cho biết khi thực hiện các lệnh sau thì S sẽ chứa những phần tử nào:</p> <p>$\text{Push}(S, 1)$; $\text{push}(S, 5)$; $\text{pop}(S)$; $\text{push}(S, 10)$;</p>
<p>Thực hiện nhiệm vụ 2</p>	<p>HS tiếp nhận nhiệm vụ, nghiên cứu SCD thảo luận nhóm hoàn thành 2 câu hỏi củng cố.</p> <p>GV quan sát, trợ giúp kịp thời khi các em cần hỗ trợ, giải đáp.</p>
<p>Báo cáo, thảo luận</p>	<p>GV: Gọi đại diện các nhóm trình lên bảng viết đáp án của nhóm mình. GV yêu cầu các nhóm quan sát, nhận xét chấm điểm cho từng</p>
<p>Kết luận, nhận định</p>	<p>GV gọi 1 HS chốt lại phương án đúng.</p> <p>1: Để lấy phần tử ở đáy phải $\text{pop}(S)$ tất cả các phần tử trong Stack ra thì mới lấy được phần tử ở đáy ngăn xếp.</p> <p>2: cho kết quả 1, 10.</p>

Hoạt động 2: Tìm hiểu mô hình dữ liệu hàng đợi

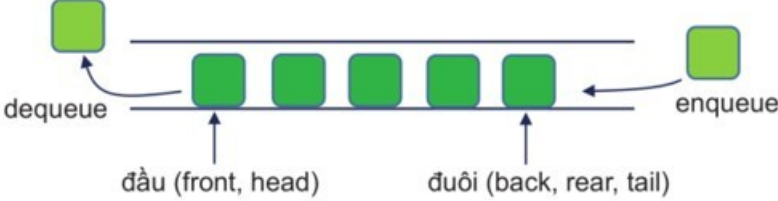
- a) Mục tiêu: Hiểu được cơ chế hoạt động của hàng đợi, các thao tác cơ bản của và ý nghĩa của các thao tác này.
b) Nội dung hoạt động: Trả lời phiếu học tập số 3, câu hỏi củng cố.

PHIẾU HỌC TẬP SỐ 3: Hãy ghép mỗi mục ở cột A với một mục ở cột B cho phù hợp.	
Cột A	Cột B
1. Hàng đợi (queue) là gì? 2. Trình bày cơ chế hoạt động của mô hình dữ liệu hàng đợi. 3. Thao tác đưa phần tử x vào cuối hàng đợi Q. 4. Tạo một hàng đợi rỗng. 5. Lấy ra một phần tử tại đầu của hàng đợi Q và trả về phần tử này. 6. Trả về phần tử đầu của hàng đợi Q và Q không thay đổi. 7. Kiểm tra hàng đợi rỗng, trả về giá trị True nếu Q rỗng, ngược lại trả về False.	a. Q=Queue(). b. enqueue(Q,x). c. Là một dãy tuyến tính các phần tử dữ liệu. d. Thao tác đưa phần tử vào và lấy phần tử ra tại cùng một đầu của hàng đợi. e. dequeue(Q). f. Hoạt động theo cơ chế FIFO. g. front(Q). h. Thao tác đưa dữ liệu vào ở một đầu, lấy dữ liệu ở một đầu khác. i. isEmptyQueue(Q).

c) Sản phẩm: HS hoàn thành được phiếu học tập số 3.

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, mỗi nhóm thảo luận thống nhất phương án trả lời.

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ 1	GV yêu cầu HS thảo luận nhóm, nghiên cứu SCD để hoàn thành phiếu học tập số 3.
Thực hiện nhiệm vụ 1	HS tiếp nhận nhiệm vụ, nghiên cứu SCD thảo luận nhóm hoàn thành phiếu học tập số 3 và 2 câu hỏi củng cố. GV quan sát HS tự học, thảo luận, trợ giúp kịp thời khi các em cần hỗ trợ, giải đáp.
Báo cáo, thảo luận	GV: Gọi đại diện các nhóm trình lên bảng viết đáp án của nhóm mình. GV yêu cầu các nhóm quan sát, nhận xét chấm điểm cho từng nhóm.
Kết luận, nhận định	Phiếu học tập số 3: 1c, 2f, 3b, 4a, 5e, 6g, 7i. - GV nhận xét, chuẩn hoá kiến thức cho HS ghi nhớ. Mô hình dữ liệu hàng đợi được mô tả như sau:

	<p>+ Là một dãy tuyến tính các phần tử dữ liệu.</p> <p>+ Có thao tác đưa phần tử vào cuối hàng đợi, lấy một phần tử ra tại đầu hàng đợi.</p> <p>+ Quy ước đầu dùng để đưa dữ liệu ra đuôi của hàng đợi (back). Đầu ngược lại dùng để lấy dữ liệu ra (front) của hàng đợi.</p> <p>+ Hoạt động theo cơ chế FIFO (vào trước, ra trước).</p>  <p>+ Các thao tác cơ bản trên dữ liệu ngăn xếp:</p> <ul style="list-style-type: none"> - Tạo một hàng đợi rỗng: <code>Q=Queue()</code>. - Đưa 1 phần tử x vào cuối hàng đợi Q: <code>enqueue(Q,x)</code>. - Lấy ra 1 phần tử tại đầu của hàng đợi Q và trả về phần tử này: <code>dequeue(Q)</code>. - Trả về phần tử đầu của hàng đợi Q và Q không thay đổi: <code>front(Q)</code>. - Kiểm tra hàng rỗng (trả về giá trị True) : <code>isEmptyQueue(Q)</code>. Ví dụ 1: <code>Q=Queue()</code>. <p>Ví dụ 2: <code>x=dequeue(Q)</code>; lấy giá trị đầu hàng đợi Q lưu vào x. <code>enqueue(Q,x)</code> đẩy giá trị x vào cuối hàng đợi Q.</p>
Chuyển giao nhiệm vụ 2	<p>GV nêu câu hỏi, HS thảo luận, suy nghĩ phát biểu:</p> <ol style="list-style-type: none"> 1. Chỉ ra những điểm giống và khác nhau giữa ngăn xếp và hàng đợi. 2. Sau khi thực hiện các lệnh sau, hỏi trong hàng đợi Q có những giá trị nào? <p><code>Q=Queue()</code> <code>Enqueue(Q,2)</code>; <code>enqueue(Q,10)</code>; <code>dequeue(Q)</code>; <code>enqueue(Q,1)</code>; <code>dequeue(Q)</code></p>
Thực hiện nhiệm vụ 2	<p>HS tiếp nhận nhiệm vụ, nghiên cứu SCD thảo luận nhóm hoàn thành 2 câu hỏi củng cố.</p> <p>GV quan sát HS tự học, thảo luận, trợ giúp kịp thời khi các em cần hỗ trợ, giải đáp.</p>
Báo cáo, thảo luận	<p>GV: Gọi đại diện các nhóm trình lên bảng viết đáp án của nhóm mình. GV yêu cầu các nhóm quan sát, nhận xét chấm điểm cho từng nhóm.</p> <ol style="list-style-type: none"> 1. Giống nhau: Stack và Queue là các cấu trúc dữ liệu được sử dụng để lưu trữ các yếu tố dữ liệu và nó dựa trên một số các ví dụ

Khác nhau:	Sự khác biệt lớn nhất giữa Stack và Queue là Stack sử dụng phương thức Stack chỉ có một đầu mở để pushing và popping các phần tử dữ liệu, còn Queue có 2. Trong Q có giá trị 1.
------------	---

3. Hoạt động luyện tập

- Mục tiêu: củng cố kiến thức đã học.
- Nội dung: Trả lời câu hỏi 1, 2 phần luyện tập.
- Sản phẩm: Bài làm của HS, kỹ năng giải quyết nhiệm vụ học tập.
 - Dãy số mới thu được là đảo ngược của dãy số đưa vào.
 - Cho dãy số 2, 5, 1, 0, 10, khi đưa các giá trị lẻ vào ngăn xếp sau đó lấy ra ta thu được dãy số: 1, 5.
- Tổ chức thực hiện: HS làm ngay vào vở ghi, GV kiểm tra bài làm của HS qua vở ghi HS và gọi hai HS chữa bài.
GV gọi HS khác nhận xét, đánh giá, chốt đáp án.

4. Hoạt động vận dụng

- Mục tiêu: HS vận dụng được kiến thức để làm bài tập.
 - Nội dung: Làm bài tập 1, 2 phần vận dụng SCĐ trang 8.
- Bài 1:**
- + Xếp hàng mua vé tàu là mô hình hàng đợi, ai đến trước mua trước, ai đến sau mua sau.
 - + Bánh quy xếp vào trong hộp đựng, Bánh nào xếp vào đầu tiên sẽ ăn sau, bánh xếp sau cùng ở trên cùng sẽ được ăn trước.
- Bài 2.** Cách thực hiện như sau:
- Bước 1. Lần lượt lấy các số từ ngăn xếp S ra và đưa vào hàng đợi Q. Bước 2. Lần lượt lấy các số từ hàng đợi Q và đưa vào ngăn xếp S.
- Sau các bước trên ngăn xếp S sẽ chứa các phần tử theo thứ tự ngược lại so với dãy ban đầu nằm trong S.
- Sản phẩm: Bài làm trong vở của HS.
 - Tổ chức thực hiện: HS làm ở nhà, tiết học sau kiểm tra vở ghi HS và gọi một số HS chữa bài.
 - GV nhận xét, đánh giá, đưa ra một vài ý mà HS còn thiếu, chốt đáp án.

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phần vận dụng.
- Đọc trước bài 2 kiểu dữ liệu ngăn xếp.

BÀI 2. KIỂU DỮ LIỆU NGĂN XẾP

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Biểu diễn được kiểu dữ liệu ngăn xếp bằng list trong Python.
- Giải thích và viết được các chương trình con có sử dụng các hàm cơ bản của kiểu dữ liệu.

2. Năng lực

Năng lực chung:

- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học, biết lắng nghe.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.

Năng lực Tin học:

- Biểu diễn được ngăn xếp bằng mảng một chiều.
- Giải thích và viết được các chương trình con sử dụng các hàm cơ bản của kiểu dữ liệu ngăn xếp.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành các bài tập theo yêu cầu của GV thông qua hệ thống câu hỏi, phiếu học tập, thông qua sản phẩm.

II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, Máy tính, Máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước bài 3. Thực hành với kiểu dữ liệu ngăn xếp.

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động khởi động

- Mục tiêu: Tạo hứng thú học tập cho HS.
- Nội dung: HS trả lời câu hỏi sau:
Theo em, những kiểu dữ liệu sau có thể được dùng để thiết lập dữ liệu ngăn xếp không?
Tại sao?
+ Sử dụng kiểu mảng có chiều dài cố định N, với số tự nhiên N khá lớn.
+ Sử dụng kiểu dữ liệu danh sách liên kết.
+ Sử dụng kiểu dữ liệu list của python.
- Sản phẩm: Câu trả lời của HS.
- Tổ chức thực hiện: chia lớp thành ba nhóm thảo luận và ghi ra giấy câu trả lời của nhóm. GV gọi đại diện lần lượt ba nhóm trả lời.
GV: Nhận xét có thể dùng được ba kiểu dữ liệu trên để biểu diễn mô hình dữ liệu ngăn xếp được. Nếu có thì các thao tác chính như pop() và push() sẽ được thiết lập như thế nào? Chúng ta cùng tìm hiểu trong bài học này.

2. Hoạt động hình thành kiến thức

Hoạt động 2.1: Tìm hiểu biểu diễn ngăn xếp bằng mảng một chiều.

- Mục tiêu: HS biết được cách sử dụng kiểu dữ liệu list trong Python để mô tả ngăn xếp thông qua phiếu học tập số 1.
- Nội dung:

1. Nhắc lại kiểu dữ liệu list của Python, các hàm cơ bản, phương thức thường hay dùng trong list.

- List là một dãy các phần tử tuyến tính, đánh chỉ số từ 0.
 - Có thể tạo list rỗng bằng lệnh `A = []` hoặc `A = list()`.
 - List trong Python có thể có không hạn chế các phần tử.
 - Bổ sung thêm phần tử `x` vào cuối của danh sách `A` bằng phương thức `A.append(x)`.
 - Phần tử cuối của list `A` có chỉ số `len(A)-1`.
 - Hàm `A.pop()` sẽ xoá phần tử cuối của `A` và trả về giá trị phần tử này.
2. Ngăn xếp được cài đặt bằng mảng 1 chiều (kiểu list). Các thao tác sau trên ngăn xếp cần được mô tả bằng lệnh tương ứng bằng các lệnh trên list của Python.

Thao tác trên ngăn xếp	Lệnh tương ứng trên list của Python
Tạo ngăn xếp rỗng	Tạo list rỗng
<code>push(S,x)</code>	Bổ sung <code>x</code> vào list bằng lệnh <code>append()</code>
<code>pop(S)</code>	Phương thức <code>pop()</code>
<code>top(S)</code> – đỉnh của <code>S</code>	Phần tử đáy của ngăn xếp là phần tử đầu tiên của list
<code>bottom</code> – đáy của <code>S</code>	Phần tử đỉnh của ngăn xếp là phần tử cuối cùng của list

Câu 1. Điền vào chỗ trống:

PHIẾU HỌC TẬP SỐ 1

- Thêm một phần tử `x` vào cuối danh sách `A`:.....
- Khởi tạo danh sách `A`:.....
- Số lượng phần tử trong danh sách `A`:.....
- Phần tử cuối cùng trong danh sách `A` có địa chỉ là:.....
- Xoá phần tử cuối cùng trong danh sách `A`:..... Câu 2.

Ghép nối các dòng cột A với cột B cho đúng:

A.	B.
a. <code>S=[]</code>	1. Đỉnh của ngăn xếp là phần tử cuối cùng của danh sách
b. <code>push(S,x)</code>	2. Lấy ra và trả về phần tử ở đỉnh ngăn xếp
c. <code>pop(S)</code>	3. Khởi tạo ngăn xếp <code>S</code>
d. <code>top</code> (Đỉnh của ngăn xếp)	4. Đẩy phần tử có giá trị <code>x</code> vào ngăn xếp <code>S</code>
e. <code>bottom</code> (Đáy của ngăn xếp)	5. Là phần tử đầu tiên của danh sách

c) Sản phẩm: Bài làm của HS.

d) Tổ chức thực hiện: chia lớp thành 6 nhóm, các nhóm thảo luận và hoàn thành phiếu học tập số 1.

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ	GV: Phát phiếu học tập số 1.

Thực hiện nhiệm vụ	HS nhận phiếu học tập, nghiên cứu sách giáo khoa, thảo luận để trả lời các câu hỏi trong phiếu học tập số 1. GV: Quan sát các nhóm làm bài.
Báo cáo, thảo luận	GV gọi mỗi nhóm 2 HS lên bảng ghi kết quả bài làm của nhóm mình lên bảng. HS: Lên bảng trình bày.
Kết luận, nhận định	GV: Nhận xét kết quả làm của từng nhóm. Khẳng định chung các nhóm đều khẳng định rằng có thể biểu diễn ngăn xếp bằng list của Python. Hai phép toán push(S,x) : đẩy x vào S. pop(S): lấy phần tử cuối của mảng. Phần tử đỉnh (top) của ngăn xếp là phần tử cuối cùng của danh sách. Câu 1 (SCD trang 10): 1, 3, 5. Câu 2(SCD trang 10): 2, 3, 1.
Chuyển giao nhiệm vụ	Yêu cầu cả lớp làm nhanh bài 1, 2 SCD trang 10 phần củng cố kiến thức.
Thực hiện nhiệm vụ	HS làm bài vào vở.
Báo cáo, thảo luận	GV kiểm tra kết quả làm bài của HS trong vở.
Kết luận, nhận định	GV: Chấm điểm cho 2 HS làm bài xong nhanh nhất, nhận xét bài làm của HS, phân tích chạy chương trình để HS hiểu thao tác push(s,x) và pop(S). HS: Quan sát chương trình và kết quả chạy chương trình.

Hoạt động 2.2: Tìm hiểu các phép toán của kiểu dữ liệu ngăn xếp.

a) Mục tiêu: HS hiểu và cài đặt được các hàm cơ bản của ngăn xếp bằng list của Python.

b) Nội dung: Các phép toán của kiểu dữ liệu ngăn xếp.

1. Hàm Stack() khởi tạo ngăn xếp rỗng.
2. Hàm push(S,x) dùng để thêm x vào đỉnh (top) của ngăn xếp.
3. Hàm isEmptyStack(S) trả về True nếu ngăn xếp S rỗng, ngược lại trả về False.
4. Hàm pop(S) dùng để lấy ra các phần tử tại đỉnh (top) của ngăn xếp S, Nếu ngăn xếp rỗng hàm này báo lỗi ngoại lệ ValueError.
5. Hàm top(S) trả về.

c) Sản phẩm: Bài làm của HS trong vở ghi.

Sản phẩm dự kiến:

2. Các phép toán của kiểu dữ liệu ngăn xếp

a. Hàm Stack() khởi tạo ngăn xếp rỗngP:

```
def Stack():
    return []
```

b. Hàm push(S,x) dùng để thêm x vào đỉnh (top) của ngăn xếp:

```
def push(S,x):
    S.append(x)
```

c. Hàm isEmptyStack(S) trả về True nếu ngăn xếp S rỗng, ngược lại trả về False:

```
def
    isEmptyStack(
        S): return
        len(S)==0
```

d. Hàm pop(S) dùng để lấy ra các phần tử tại đỉnh (top) của ngăn xếp S, Nếu ngăn xếp rỗng hàm này báo lỗi ngoại lệ ValueError:

```
def pop(S):
    if isEmptyStack(S):
        raise ValueError('Ngăn xếp
        rỗng') else: return S.pop()
```

e. Hàm top(S) trả về phần tử tại đỉnh ngăn xếp S:

```
def pop(S):
    if isEmptyStack(S):
        raise ValueError('Ngăn xếp
        rỗng') else: return S[len(S)-1]
```

Cũng

có: 1.

S=Stack()

```
x=pop(S)
print(x,end='
')
```

kết quả:15 8 20

2. Hàm pop(S)

```
def pop(S):
    if isEmptyStack(S):
        raise ValueError('Ngăn xếp rỗng không thể thực hiện
        được') else: return S.pop()
```

Hàm top(S)

```
def top(S):
    if isEmptyStack(S):
        raise ValueError('Ngăn xếp rỗng không thể thực hiện
        được') else: return S[len(S)-1]
```

3. Khi sử dụng list để biểu diễn ngăn xếp thì ta luôn có chỉ số phần tử đáy là 0, chỉ số d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận, giải thích ý nghĩa các hàm, các câu lệnh trong mỗi hàm, viết các hàm cơ bản của Ngăn xếp vào vở và gõ trên máy tính.

Hoạt động của GV và HS Bước 1: Chuyển giao nhiệm vụ

GV: Hãy nghiên cứu SCD và viết các phép toán của kiểu dữ liệu ngăn xếp:

1. Hàm Stack() khởi tạo ngăn xếp rỗng.
2. Hàm push(S,x) dùng để thêm x vào đỉnh (top) của ngăn xếp.
3. Hàm isEmptyStack(S) trả về True nếu ngăn xếp S rỗng, ngược lại trả về False.
4. Hàm pop(S) dùng để lấy ra các phần tử tại đỉnh (top) của ngăn xếp S, Nếu ngăn xếp rỗng hàm này báo lỗi ngoại lệ ValueError.
5. Hàm top(S) trả về.

Bước 2: Thực hiện nhiệm vụ

HS: Viết 5 hàm ra vở, giải thích ý nghĩa của từng câu lệnh. GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận

HS: Mở vở cho GV kiểm tra.

GV: Kiểm tra vở ghi của HS, gọi 5 HS lên bảng mỗi HS viết một hàm.

Bước 4: Kết luận, nhận định

GV nhận xét và gọi một số HS đứng tại vị trí giải thích ý nghĩa của từng câu lệnh trong các hàm trên bảng.

Chạy chương trình cho HS quan sát kết quả.

HS: Nghe, quan sát và ghi nhớ các hàm.

Bước 1: Chuyển giao nhiệm vụ

GV: Nêu yêu cầu cho HS.

1. Với dãy câu lệnh trên khi chạy chương trình màn hình in ra kết quả như thế nào?
2. Sửa lại hàm pop(S) và top(S) nếu ngăn xếp rỗng thì thông báo “Ngăn xếp rỗng không thể thực hiện được lệnh này.”
3. Vì sao các hàm cơ bản trên ngăn xếp được cài bằng danh sách không cần sử dụng biến top và bottom?

Bước 2: Thực hiện nhiệm vụ

HS: Thảo luận, trả lời, viết ra vở ghi.

Bước 3: Báo cáo, thảo luận

Gọi 3 HS bất kì trong ba nhóm lên bảng trình bày.

GV: Gọi một số HS nhận xét. GV chạy chương trình trên máy tính cho HS quan sát.

Bước 4: Kết luận, nhận định

GV: Nhận xét đánh giá cho điểm bài làm tốt của một số

HS. HS: Ghi nhớ.

3. Hoạt động luyện tập

- a) Mục tiêu: HS hiểu được các hàm cơ bản của ngăn xếp.
- b) Nội dung: làm bài tập 1, 2 phần luyện tập SCĐ trang 12.
- c) Sản phẩm: Bài làm của HS trong vở ghi hoặc trên máy tính của HS.
- d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận, giải thích ý nghĩa các hàm, các câu lệnh trong mỗi hàm, viết các hàm.

Hoạt động của GV và HS	Sản phẩm dự kiến
Bước 1: Chuyển giao nhiệm vụ GV: Yêu cầu HS làm bài 1, 2 SCĐ trang 12 phần luyện tập. Bước 2: Thực hiện nhiệm vụ HS: Thảo luận, trả lời, viết ra vở ghi. Bước 3: Báo cáo, thảo luận Gọi 2 HS bất kì trong ba nhóm lên bảng trình bày. GV: Gọi 2 HS nhận xét. Bước 4: Kết luận, nhận định GV: Nhận xét đánh giá cho điểm bài làm của 2 HS. HS: Ghi nhớ.	Hàm length(S) def length(S): return len(S) 2. Các lệnh đã thực hiện push(S,2) push(S,7) push(S,6) pop(S) pop(S) push(S,1) pop(S) pop()

4. Hoạt động vận dụng

- a) Mục tiêu: HS vận dụng kiến thức về kiểu dữ liệu ngăn xếp viết chương trình đáp ứng yêu cầu của bài.
- b) Nội dung: Làm bài tập 1, 2 SCD trang 12 phần vận dụng.
- c) Sản phẩm: Bài làm của HS trong vở ghi hoặc trong máy tính.
- d) Tổ chức thực hiện: HS làm ở nhà, tiết sau GV gọi 2 HS lên bảng chữa bài.

Sản phẩm dự kiến:

Bài 1. Hàm kiểm tra chuỗi bt chỉ bao gồm các ký tự “(”, “)” có hợp lệ hay không có thể viết như sau: Hàm sẽ trả lại True nếu biểu thức đúng, ngược lại trả về False.

```
from Stack import * def checkbt(a):
S=Stack() kq=True for i in a:
if i=='(':
push(S,i)
if i==')':
if isEmptyStack(S): kq=False
break else:
pop(S)
return isEmptyStack(S) and kq a=input()
if checkbt(a): print(a+'là biểu thức
đúng')
else:
print(a+'không là biểu thức
đúng')
```

Chương trình lưu trong tệp Stack.py

```
def Stack():
    return []
def push(S,x):
    S.append(x)
def isEmptyStack(S): return len(S)==0
def pop(S):
    if isEmptyStack(S):
        raise ValueError('Ngăn xếp
        rỗng')
    else:
        return S.pop()
def top(S):
    if isEmptyStack(S):
        raise ValueError('Ngăn xếp
        rỗng')
    else:
        return S[len(S)-1]
```

Bài 2. Chương trình thiết lập kiểu dữ liệu ngăn xếp dựa trên mảng dữ liệu cho trước có độ dài cố định N có thể như sau (lưu vào tệp **Stack_segment.py**).

```
N=20
T=[None]*N
topIdx=-1
def Stack():
    global topIdx
    topIdx=-1
    return T
def isEmptyStack(S): return topIdx==-1
def stackOverflow(S): return topIdx==N-1
def push(S,x):
    global topIdx
    if stackOverflow(S):
        raise ValueError('Ngăn xếp bị tràn')
    else:
        topIdx+=1
        S[topIdx]=x
def pop(S):
    global topIdx
    if isEmptyStack(S):
        raise ValueError('Ngăn xếp rỗng')
    else:
        x=S[topIdx]
        topIdx-=1
        return x
def top(S):
    if isEmptyStack(S):
        raise ValueError('Ngăn xếp rỗng')
    else:
        return S[topIdx]
```

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phần vận dụng.
- Đọc trước bài 3. Thực hành với kiểu dữ liệu ngăn xếp.

BÀI 3. THỰC HÀNH VỚI DỮ LIỆU NGĂN XẾP

Thời gian thực hiện: 2 tiết thực hành

I. MỤC TIÊU

1. Kiến thức

- Biết cách sử dụng cấu trúc dữ liệu ngăn xếp để giải quyết các bài toán thực tế.

2. Năng lực

Năng lực chung:

- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Thực hiện các nhiệm vụ và làm các bài tập giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.

Năng lực Tin học: Có khả năng phân tích yêu cầu của bài toán để sử dụng kiểu dữ liệu ngăn xếp một cách phù hợp.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành các nhiệm vụ và bài tập theo yêu cầu của GV thông qua các bài làm, thông qua sản phẩm.

II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, Máy tính, Máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước bài 3.

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động khởi động

- Mục tiêu: Tạo hứng thú học tập cho HS.
- Nội dung: HS trả lời câu hỏi sau:
Theo em, có thể sử dụng kiểu dữ liệu ngăn xếp để mô tả chức năng quay lại trang web đã duyệt trong các trình duyệt thông dụng như Google Chrome hay Bing được không?
- Sản phẩm: Câu trả lời của HS.
- Tổ chức thực hiện: HS thảo luận và đứng tại vị trí trả lời.

2. Hoạt động thực hành

Nhiệm vụ 1: Viết chương trình mô phỏng quá trình duyệt web.

- Mục tiêu: HS cài đặt được chương trình mô phỏng duyệt web.
- Nội dung: Viết chương trình mô phỏng quá trình duyệt web của người dùng bằng cách sử dụng kiểu dữ liệu ngăn xếp. chương trình nhập vào số nguyên 1 để nhập địa chỉ web, ấn số 2 để quay trở lại, ấn số 3 để kết thúc.
- Sản phẩm: Bài làm của HS trên máy tính.

Sản phẩm dự kiến: `from Stack import * back=Stack()`

```
option=0
while option!=3:
    option=int(input('Hãy nhập vào lựa chọn:\n'))
    if option==1:
        website=input('Hãy nhập vào địa chỉ trang web muốn đi đến\n') push(back,website)
    print('Đang đi đến trang web:'+website) if option==2:
    if isEmptyStack(back):
```

```
print('Không tồn tại lịch sử duyệt web') else:
website=pop(back)
print('Đang đi đến trang web:'+website)
```

d) Tổ chức thực hiện: HS làm bài trên máy tính.

Hoạt động của GV và HS

Bước 1: Chuyển giao nhiệm vụ

GV: Yêu cầu HS đọc đề bài, phân tích yêu cầu bài toán, viết chương trình trên máy tính.

Bước 2: Thực hiện nhiệm vụ

HS: Nghiên cứu SCD, làm bài trên máy tính. GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận

HS: Chạy chương trình báo cáo kết quả cho GV. GV: Kiểm tra kết quả làm của các nhóm.

Bước 4: Kết luận, nhận định

GV: Đề viết được chương trình trên phải viết trước các hàm của ngăn xếp và lưu vào máy tính. Chạy chương trình, phân tích để HS hiểu hoạt động của chương trình.

Hoạt động 2.2 thực hành nhiệm vụ 2: Viết chương trình kiểm tra các dấu ngoặc trong biểu thức.

- a) Mục tiêu: HS viết được chương trình kiểm tra biểu thức ngoặc đúng.
- b) Nội dung: Viết chương trình nhập vào một biểu thức toán học và kiểm tra các dấu đóng mở ngoặc trong biểu thức có hợp lệ hay không. Biểu thức có thể chứa hai loại dấu đóng mở là dấu “()” và dấu “[]”. Biểu thức là hợp lệ là biểu thức mà trong đó mỗi dấu mở ngoặc cần có các dấu đóng ngoặc tương ứng theo trình tự xuất hiện. Ví dụ: $[(5+4)/(9-3)]$ là hợp lệ, còn $[(5+4)/(9-3)]$ là không hợp lệ.
- c) Sản phẩm: Bài làm của HS trên máy tính.

Sản phẩm dự kiến:

Thuật toán:

Sử dụng cấu trúc ngăn xếp, khi gặp các dấu mở ngoặc thì các dấu được đẩy vào ngăn xếp.

Khi gặp dấu đóng ngoặc thì kiểm tra ngăn xếp. Nếu ngăn xếp rỗng hoặc phần tử tại đỉnh ngăn xếp không phải là dấu mở ngoặc tương ứng thì biểu thức không hợp lệ.

Tiến hành duyệt cho đến hết biểu thức. Sau đó kiểm tra xem ngăn xếp có rỗng không.

Nếu ngăn xếp là rỗng, biểu thức hợp lệ. Nếu không rỗng tức còn dấu ngoặc mở chưa đóng, biểu thức không hợp lệ.

```
from Stack import * def kiemtrabt(bt): hople=True
```

```

S=Stack()
for i in range(0,len(bt)):
    if bt[i] in {"(", "["}: push(S, bt[i]); elif bt[i] in {")", "}"}:
        if isEmptyStack(S): hople=False; break
    else:
        tmp=pop(S)
        if (bt[i]=="") and tmp!="(" or (bt[i]=="") and
            tmp!="["):
            hople=False break
        if not isEmptyStack(S): hople=False return hople
bt=input('Hãy nhập vào một biểu thức:\n') hople=kienmtrabt(bt)
if hople:
    print('Biểu thức hợp lệ') else:
    print('Biểu thức không hợp lệ')

```

d) Tổ chức thực hiện: HS làm bài trên máy tính.

Hoạt động của GV và HS Bước 1: Chuyển giao nhiệm vụ

GV: Yêu cầu HS đọc đề bài, phân tích yêu cầu bài toán.

Phân tích biểu thức hợp lệ không hợp lệ.

Biểu thức hợp lệ nếu số lượng các dấu ngoặc mở và ngoặc đóng phải làm sao? Biểu thức $(())$ có hợp lệ không?

Biểu thức $([])$ có hợp lệ không?

Thuật toán kiểm tra biểu thức hợp lệ như thế nào?

Viết chương trình trên máy tính kiểm tra biểu thức có hợp lệ không.

Bước 2: Thực hiện nhiệm vụ

HS: Nghiên cứu SCD, trả lời câu hỏi. làm bài trên máy tính.

GV: Phân tích thông qua ví dụ minh họa cho HS hiểu thuật toán. Quan sát và trợ giúp HS viết chương trình.

Bước 3: Báo cáo, thảo luận

HS: Chạy chương trình báo cáo kết quả cho GV.

GV: Kiểm tra kết quả làm của các nhóm thông qua chạy chương trình với các bộ dữ liệu $(())$; $[(9-3)*(9+8)]$; $[(4-5)]$.

Bước 4: Kết luận, nhận định

GV: Để viết được chương trình trên phải viết trước các hàm của ngăn xếp và lưu vào máy thư mục chứa bài tập với tên tệp là Stack.py để bài duyệt web sẽ khai thác các hàm trong Stack đã xây dựng.

3. Hoạt động luyện tập

a) Mục tiêu: HS thực sự hiểu được chương trình của Nhiệm vụ 1 và Nhiệm vụ 2 trong SCD.

b) Nội dung: Làm bài tập 1, 2 SCD trang 15 phần luyện tập.

1. Hãy sửa chương trình trong nhiệm vụ 1 để thêm chức năng đi đến trang web kế tiếp (go forward). Sau khi người dùng chọn chức năng trở về trang web trước đó thì có thể

sử dụng chức năng go forward để quay lại trang web vừa duyệt.

2. Sửa chương trình trong nhiệm vụ 2 để in ra màn hình tổng số cặp đóng mở ngoặc của từng loại xuất hiện trong biểu thức.

c) Sản phẩm: Bài làm của HS trên máy tính, vở ghi.

Sản phẩm dự kiến:

Bài 1 (luyện tập trang 15)

Hướng dẫn: Để làm được yêu cầu bài toán, chúng ta cần phải sử dụng thêm 1 ngăn xếp “forward” để lưu trữ địa chỉ website trước thực hiện quay lại trang web cũ. Mỗi khi sử dụng chức năng “backward” chúng ta cần đưa địa chỉ trang web hiện tại vào ngăn xếp forward.

Trong bài toán này, chúng ta sử dụng tùy chọn 1 để nhập địa chỉ mới, tùy chọn 2 để go backward, tùy chọn 3 để go forward và tùy chọn 4 để dừng chương trình. Nhiệm vụ này có thể được cài đặt chương trình như sau:

```
from Stack import * backwark=Stack() forward=Stack() current_web="" option=0
while option!=4:
option=int(input('Hãy nhập vào lựa chọn:\n')) if option==1:
website=input('Hãy nhập vào địa chỉ trang web muốn đi đến\n') if current_web!="":
push(backwark,website)
print('Đang đi đến trang web:'+website) current_web=website

if option==2:
if isEmptyStack(back):
print('Không tồn tại lịch sử duyệt web') else:
push(forward,current_web) current_web=pop(backward)
print('Đang đi đến trang web:'+current_web) if option==3:
if isEmptyStack(forward):
print("Không tồn tại lịch sử duyệt web") else:
push(backward,current_web) current_web=pop(forward)
print('Đang đi đến trang webs:'+current_web)
```

Bài 2 (luyện tập trang 15)

Hướng dẫn: Trong bài toán này, chúng ta chỉ in ra kết quả số lượng các cặp đóng mở ngoặc nếu biểu thức là hợp lệ. Chú ý rằng khi đó số ngoặc mở - được đưa vào ngăn xếp bằng lệnh push sẽ bằng với số ngoặc đóng – được lấy ra khỏi ngăn xếp bằng lệnh pop. Do vậy việc đếm số lượng các cặp ngoặc đóng mở chỉ cần được thực hiện khi chúng ta sử dụng lệnh push. Có hai loại ngoặc nên chúng ta sẽ sử dụng thêm hai biến là dem1 để đếm số lượng ngoặc ‘(’ và dem2 để đếm số lượng ngoặc ‘[’. Chương trình trong Nhiệm vụ 2 có thể được sửa lại như sau:

```
from Stack import *
def kiemtrabt(bt,dem1,dem2): hople=True
S=Stack()
for i in range(0,len(bt)): if bt[i] in {"(", "["}:
push(S,bt[i]);
```

```

if bt[i]=='(': dem1+=1 else: dem2+=1
elif bt[i] in {"", ""}: if isEmptyStack(S):
hople=False break
else:
tmp=pop(S)
if (bt[i]=="") and tmp!="(" or (bt[i]=="") and tmp!="("): hople=False
break if not isEmptyStack(S):
hople=False return hople, dem1, dem2
bt=input('Hãy nhập vào một biểu thức:\n') dem1, dem2=0,0
hople, dem1, dem2=kiemtrabt(bt, dem1, dem2) if hople:
print('Biểu thức hợp lệ')
print('Tổng số cặp ngoặc tròn là:', dem1) print('Tổng số cặp ngoặc vuông là:', dem2)
else:
print('Biểu thức không hợp lệ')

```

d) Tổ chức thực hiện: HS làm bài trên máy tính.

Hoạt động của GV và HS

Bước 1: Chuyển giao nhiệm vụ

GV: 1. Yêu cầu HS chỉnh sửa chương trình trong nhiệm vụ 1, thêm chức năng đi đến trang wel
2. Yêu cầu chỉnh sửa chương trình trong nhiệm vụ 2 để in ra màn hình tổng số cặp ngoặc đóng

Bước 2: Thực hiện nhiệm vụ

HS: Nghiên cứu yêu cầu, phân tích bổ sung câu lệnh vào chương
trình. GV: Phân tích bài 1 cần thêm chức năng khi.

Quan sát và trợ giúp HS viết chương trình.

Bước 3: Báo cáo, thảo luận

HS: Chạy chương trình báo cáo kết quả cho

GV. GV: Kiểm tra kết quả làm của các nhóm.

Bước 4: Kết luận, nhận định

GV: Để viết được chương trình trên phải viết trước các hàm của ngăn xếp và lưu vào
máy thư mục chứa bài tập với tên tệp là Stack.py để bài duyệt web sẽ khai thác các
hàm trong Stack đã xây dựng

4. Hoạt động vận dụng

a) Mục tiêu: HS thực sự hiểu được chương trình của Nhiệm vụ 1 và Nhiệm vụ 2 trong
SCĐ.

b) Nội dung: Làm bài tập 1, 2 SCĐ trang 15 phần vận dụng.

Bài 1: Viết chương trình mô phỏng quá trình sắp xếp và lấy sách ra khỏi ngăn tủ.

Bài 2. Cải tiến nhiệm vụ 2 kiểm tra biểu thức có ba loại dấu đóng mở ngoặc “()”, “[]”, “{ }”.

c) Sản phẩm: Bài làm của HS trên máy tính, vở ghi.

d) Tổ chức thực hiện: HS làm bài ở nhà, tiết sau GV gọi lên bảng chữa bài.

Sản phẩm dự kiến:

Bài 1. Vận dụng SCD trang 15:

Dữ liệu đọc vào từ tệp sach.inp gồm tên các quyền sách.

Dùng kiểu dữ liệu ngăn xếp, với tên mỗi quyền sách đọc vào từ file Sach.inp lưu vào ngăn xếp.

Sau đó cho phép người dùng nhập tên một quyền sách cần lấy. Lần lượt lấy các quyền sách ra khỏi ngăn xếp bằng cách sử dụng hàm *pop* cho đến khi lấy được quyền sách được yêu cầu. Đồng thời tiến hành đếm số quyền sách được lấy ra khỏi ngăn. In ra kết quả số quyền sách được lấy ra trước khi lấy được quyền sách cần lấy.

Tệp sach.inp như sau: Anh Lý Hoá Sinh Văn Sử Địa Toán Tin.

```
from Stack import * fi=open("sach.inp","r",encoding="UTF8") data=fi.read()
S=Stack()
for sach in data.split(): push(S,sach)
fi.close()
sach=input('Nhập vào tên quyền sách cần lấy:\n') ok=0
dem=0
while not isEmptyStack(S): tmp=pop(S)
if tmp==sach: ok=1 break
dem+=1 if ok==0:
print('Không tìm thấy sách cần lấy trong tủ sách') else:
print('Số quyền sách cần lấy ra trước là:',dem)
```

Bài 2. Vận dụng SCD trang 15:

Chương trình kiểm tra tính hợp lệ của biểu thức với ba loại dấu đóng, mở ngoặc “()”, “[]”, “{ }”.

Code chương trình đầy đủ:

```
from Stack import * def kiemtrabt(bt): hople=True
S=Stack()
for i in range(0,len(bt)):
if bt[i] in {"(","[","{"}: push(S,bt[i]);
elif bt[i] in {")",""],""}: if isEmptyStack(S): hople=False
break else:
tmp=pop(S)
if (bt[i]=="(" and tmp!="(") or (bt[i]=="]" and tmp!="]") or (bt[i]=="}" and tmp!="{"):
hople=False break
if not isEmptyStack(S): hople=False
return hople
bt=input('Hãy nhập vào một biểu thức:\n') hople=kiemtrabt(bt)
if hople:
print('Biểu thức hợp lệ') else:
```

print('Biểu thức không hợp lệ')

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phần vận dụng.
- Đọc trước bài 4. Kiểu dữ liệu hàng đợi.

BÀI 4. KIỂU DỮ LIỆU HÀNG ĐỢI

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Cách thiết lập kiểu dữ liệu hàng đợi bằng danh sách (list) trong Python.
- Các thao tác cơ bản với hàng đợi được thực hiện trên kiểu dữ liệu danh sách (list) của Python.

2. Năng lực

Năng lực chung:

- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.

Năng lực Tin học:

- Biểu diễn được kiểu dữ liệu hàng đợi bằng mảng một chiều.
- Giải thích và viết được các chương trình con có sử dụng các hàm cơ bản của kiểu dữ liệu hàng đợi.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành các bài tập theo yêu cầu của GV thông qua hệ thống câu hỏi, phiếu học tập, thông qua sản phẩm.

II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, Máy tính, Máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước Bài 4.

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động khởi động

a) Mục tiêu: Tạo hứng thú học tập cho HS.

b) Nội dung: HS trả lời câu hỏi sau:

Trong bài học trước các em đã viết chương trình sử dụng các hàm cơ bản của hàng đợi được cài đặt bằng kiểu danh sách (kiểu list trong Python). Em hãy trả lời các câu hỏi sau:

- Có thể cài đặt hàng đợi bằng mảng một chiều tương tự như ngăn xếp được không?
- Khi cài đặt hàng đợi bằng mảng một chiều, cần có thông tin nào để thực hiện phép toán thêm vào và lấy ra?

c) Sản phẩm: Câu trả lời của HS.

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm thảo luận và ghi ra giấy câu trả lời của

nhóm. GV gọi đại diện lần lượt 6 nhóm trả lời.

GV: Nhận xét dùng được kiểu dữ liệu danh sách để biểu diễn mô hình dữ liệu hàng đợi. Các thao tác chính như enqueue() và dequeue() sẽ được thiết lập như thế nào? Chúng ta cùng tìm hiểu trong bài học hôm nay.

2. Hoạt động hình thành kiến thức

Hoạt động 2.1: Tìm hiểu biểu diễn hàng đợi bằng mảng một chiều.

a) Mục tiêu: HS biết được cách sử dụng kiểu dữ liệu list trong Python để mô tả hàng đợi thông qua sơ đồ.

b) Nội dung:

+ Ôn tập lại kiểu dữ liệu list của Python và một số hàm, phương thức có liên quan sẽ được sử dụng để cài đặt hàng đợi.

– List có thể hiểu là một dãy các phần tử tuyến tính, đánh chỉ số từ 0.

– Có thể tạo list rỗng bằng lệnh $A = []$ hoặc $A = \text{list}()$.

– List trong Python có thể có không hạn chế các phần tử.

– Có thể bổ sung thêm phần tử vào cuối của danh sách bằng phương thức $\text{append}()$. Ví dụ $A.\text{append}(x)$ sẽ bổ sung x vào cuối dãy A .

– Phần tử cuối của list A có chỉ số $\text{len}(A)-1$ hoặc -1 .

– Phương thức (hàm) $A.\text{pop}(0)$ sẽ xóa phần tử đầu của A và trả về giá trị phần tử này.

+ Sử dụng list để biểu diễn hàng đợi:

- Tạo hàng đợi rỗng.

- Phép toán $\text{enqueue}(Q, x)$: dùng để thêm x vào Q .

0	1	2	3	4	5	chỉ số mảng
5	2	-4	10	-8	11	
↑ đầu (front)			↑ đuôi (back, rear, tail)			

Hình 4.1b. Hàng đợi với phép toán thêm vào

- $\text{dequeue}(Q)$: lấy ra phần tử ở đầu hàng đợi.

0	1	2	3	4		chỉ số mảng
2	-4	10	-8	11		
↑ đầu (front)			↑ đuôi (back, rear, tail)			

Hình 4.1c. Hàng đợi với phép toán lấy ra

- front – đầu của Q : Phần tử đầu tiên của danh sách.

- back, rear – đuôi của Q : Phần tử cuối của danh sách.

Câu 1. Điền vào chỗ trống:

PHIẾU HỌC TẬP SỐ 1

- Khởi tạo danh sách A:.....
- Thêm một phần tử x vào cuối danh sách A:.....
- List trong Python có thể có không hạn chế.....
- Số lượng phần tử có trong danh sách A:.....
- Phần tử đầu tiên trong danh sách A có địa chỉ là:.....
- Phần tử đầu tiên trong danh sách A là phương thức.....

Câu 2. Ghép nối các dòng cột A với cột B cho đúng.

A.	B.
1. $Q=[]$	a. Đỉnh của ngăn xếp là phần tử cuối cùng của danh sách.
2. $enqueue(Q,x)$	b. Xóa phần tử ở đầu hàng đợi.
3. $dequeue(Q)$	c. Khởi tạo hàng đợi Q.
4. front (đầu của hàng đợi)	d. Bổ sung phần tử có giá trị x vào hàng đợi Q bằng lệnh <code>append</code> .
5. back, rearm (Đuôi của hàng đợi)	e. Là phần tử đầu tiên của danh sách.

c) Sản phẩm: Bài làm của HS. Đáp án: Câu 2: 1c, 2d, 3b, 4e, 5a.

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và hoàn thành phiếu học tập số 1.

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ	GV: Phát phiếu học tập số 1.
Thực hiện nhiệm vụ	HS nhận phiếu học tập, nghiên cứu sách giáo khoa, thảo luận để trả lời các câu hỏi trong phiếu học tập số 1. GV: Quan sát các nhóm làm bài.
Báo cáo, thảo luận	GV gọi mỗi nhóm 2 HS lên bảng ghi kết quả bài làm của nhóm mình lên bảng.

	HS: Lên bảng trình bày.
Kết luận, nhận định	GV: Nhận xét kết quả làm của từng nhóm. Chấm điểm khen thưởng các nhóm làm đúng. Chốt lại kiến thức HS ghi nhớ hai nội dung: + Các thao tác cơ bản trên list. + Các phép toán: $enqueue(Q,x)$ và $dequeue(Q)$.
Chuyển giao nhiệm vụ	Yêu cầu cả lớp làm nhanh bài 1, 2 SCD trang 17 phần củng cố kiến thức: 1. tính số phần tử của hàng đợi khi cài đặt kiểu list. 2. giá trị của phần tử đầu và đuôi hàng đợi là bao nhiêu sau khi thực hiện các lệnh: $enqueue(Q,2)$; $enqueue(Q,10)$; $dequeue(Q)$; $enqueue(Q,6)$; $dequeue(Q)$; $enqueue(Q,9)$; $enqueue(Q,1)$.
Thực hiện nhiệm vụ	HS làm bài vào vở.
Báo cáo, thảo luận	GV kiểm tra kết quả làm bài của HS trong vở.

<p>Kết luận, nhận định</p>	<p>GV: Chấm điểm cho 3 HS bất kì, nhận xét bài làm của HS, phân tích kết quả để HS hiểu thao tác enqueue(Q,x) và dequeue(Q). HS: Quan sát chương trình và kết quả chạy chương trình trên máy của GV. 1. Số phần tử của hàng đợi Q chính là giá trị len(Q). 2. Phần tử đầu của hàng đợi Q là: 6, phần tử đuôi là 1.</p>	
<p>Hoạt động 2.2: Tìm hiểu các phép toán của kiểu dữ liệu hàng đợi.</p> <p>a) Mục tiêu: HS hiểu và cài đặt được các hàm cơ bản của hàng đợi bằng list của Python.</p> <p>b) Nội dung: Các phép toán của kiểu dữ liệu hàng đợi.</p> <p>c) Sản phẩm: Bài làm của HS trong vở ghi.</p> <p>d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận, giải thích ý nghĩa các hàm, các câu lệnh trong mỗi hàm, viết các hàm cơ bản của hàng đợi vào vở, sau đó gõ trên máy tính.</p>		
<p>Hoạt động của GV và HS</p>		<p>Sản phẩm dự kiến</p>
<p>Bước 1: Chuyển giao nhiệm vụ GV: Hãy nghiên cứu SCD và viết các phép toán của kiểu dữ liệu hàng đợi. 1. Hàm Stack() khởi tạo hàng đợi rỗng. 2. Hàm push(S,x) dùng để thêm x vào đỉnh (top) của hàng đợi.</p>	<p>2. Các phép toán của kiểu dữ liệu hàng đợi a. Hàm Queue() khởi tạo hàng đợi rỗng def Queue(): return [] b. Hàm enqueue(Q,x) dùng để thêm x vào đuôi của hàng đợi (cuối danh sách)</p>	
<p>3. Hàm isEmptyStack(S) trả về True nếu hàng đợi Q rỗng, ngược lại trả về False. 4. Hàm pop(S) dùng để lấy ra các phần tử tại đỉnh (top) của hàng đợi Q, Nếu hàng đợi rỗng hàm này báo lỗi ngoại lệ ValueError. 5. Hàm top(S) trả về.</p>		
<p>Bước 2: Thực hiện nhiệm vụ HS: Viết 5 hàm ra vở, giải thích ý nghĩa của từng câu lệnh. GV: Quan sát và trợ giúp HS. Bước 3: Báo cáo, thảo luận HS: Mở vở cho GV kiểm tra. GV: Kiểm tra vở ghi của HS, gọi 5 HS lên bảng mỗi HS viết một hàm. Bước 4: Kết luận, nhận định GV nhận xét và gọi một số HS đứng tại vị trí giải thích ý nghĩa của từng câu lệnh trong các hàm trên bảng. Chạy chương trình cho HS quan sát kết quả. HS: Nghe, quan sát và ghi nhớ các hàm. Bước 1: Chuyển giao nhiệm vụ GV: chiếu chương trình lần lượt chương trình ví dụ 1, 2 trong SCD cho HS quan sát. Bước 2: Thực hiện nhiệm vụ HS: Dự đoán kết quả. GV: Chiếu và chạy chương trình cho HS quan sát kết quả và cách gọi, sử dụng hàm.</p>		
<p>Bước 1: Chuyển giao nhiệm vụ GV: Nêu yêu cầu cho HS. 1. Khi sử dụng list để biểu diễn hàng đợi, hãy cho biết chỉ số của các phần tử đầu</p>		

(front) và đuôi (rear). So sánh các chỉ số này với chỉ số của các phần tử ở đỉnh (top) và đáy (bottom) ngăn xếp.

2. Nêu sự giống nhau và khác nhau giữa các hàm của ngăn xếp và hàng đợi được cài đặt bằng kiểu list.

```
def enqueue(Q,x): Q.append(x)
```

c. Hàm isEmptyQueue(Q) trả về True nếu hàng đợi rỗng, ngược lại trả về False

```
def isEmptyQueue(Q): return len(Q)==0
```

d. Hàm dequeue(Q) dùng để lấy ra phần tử tại đầu của hàng đợi Q, Nếu hàng đợi rỗng hàm này báo lỗi ngoại lệ ValueError

```
def dequeue(Q):
```

```
    if isEmptyQueue(Q):
        raise ValueError('Hàng
        đợi rỗng')
```

```
    else: return Q.pop(0)
```

e. Hàm front(Q) trả về phần tử tại đầu của hàng đợi

```
def front(Q):
```

```
    if isEmptyQueue(Q):
        raise ValueError('Hàng
        đợi rỗng')
```

```
    else: return Q[0]
```

ví dụ 1

```
Q=Queue()
```

```
enqueue(Q,10); enqueue(Q,5); enqueue(Q,9); dequeue(Q); ví dụ 2:
```

```
Q=Queue() for x in A:
```

```
    if x>=0: enqueue(Q,x) else: break
```

Củng cố:

1. Nếu hàng đợi Q được biểu diễn bằng mảng (list) trong Python thì:

Chỉ số phần tử đầu hàng đợi: 0.

Chỉ số phần tử cuối hàng đợi: len(Q) – 1.

Bước 2: Thực hiện nhiệm vụ

HS: Thảo luận, trả lời.

Bước 3: Báo cáo, thảo luận

GV: Gọi lần lượt một số HS trong các nhóm trả lời. Gọi một số HS khác mô tả sự giống nhau

Bước 4: Kết luận, nhận định

GV: Nhận xét đánh giá bài làm tốt của một số nhóm HS.

HS: Ghi nhớ.

So sánh với ngăn xếp được biểu

Chỉ số đầu (front) của hàng đợi c

Chỉ số đuôi (back, rear) của hàng c

3. Bảng sau mô tả sự giống nhau và khác nhau giữa các hàm cơ bản giữa ngăn xếp và hàng

Bảng sau mô tả sự giống nhau và khác nhau giữa các hàm cơ bản giữa ngăn xếp và hàng đợi:

STT	Hàm ngăn xếp	Hàm hàng đợi	So sánh
1	S = Stack()	Q = Queue()	Giống nhau hoàn toàn.
2	push(S,x)	enqueue(Q,x)	Giống nhau hoàn toàn.
3	isEmptyStack()	isEmptyQueue(Q)	Giống nhau hoàn toàn.

4	pop(S)	dequeue(Q)	Ngăn xếp: lấy ra phần tử cuối cùng của list. Hàng đợi: lấy ra phần tử đầu tiên của list.
5	top(S)	front(Q)	Ngăn xếp: phần tử cuối cùng. Hàng đợi: phần tử đầu tiên.

3. Hoạt động luyện tập

- a) Mục tiêu: HS hiểu được các hàm cơ bản của hàng đợi.
 b) Nội dung: Làm bài tập 1, 2 phần luyện tập SCD trang 19.
 c) Sản phẩm: Bài làm của HS trong vở ghi hoặc trên máy tính của HS.
 d) Tổ chức thực hiện: 2 HS thành 1 nhóm, các nhóm thảo luận làm vào vở ghi hoặc trên máy tính.

GV: Quan sát HS làm bài. Gọi 4 HS lên bảng, 2 HS làm bài 1, 2 HS làm bài 2. Sau đó lần lượt gọi 2 HS làm bài trên máy tính đem lên cắm máy chiếu chiếu và chạy chương trình cho cả lớp quan sát hàm dequeue(Q) và front(Q) sau khi chỉnh sửa.

Gọi một số HS nhận xét bài trên bảng.

GV: Nhận xét, cho điểm những HS làm bài tốt.

Sản phẩm dự kiến:

1. Sửa lại hàm dequeue(Q) và front(Q) trong chương trình trên như sau:

```
def dequeue(Q):
    if isEmptyQueue(Q):
        raise ValueError('Hàng đợi rỗng không thể thực hiện được')
    else: return Q.pop(0)
def front(Q):
    if isEmptyQueue(Q):
        raise ValueError('Hàng đợi rỗng không thể thực hiện được')
    else: return Q[0]
```

2. Hàm length(Q) có thể viết như sau:

```
def length(Q): return len(Q).
```

4. Hoạt động vận dụng

- a) Mục tiêu: HS vận dụng kiến thức về kiểu dữ liệu hàng đợi viết chương trình đáp ứng yêu cầu của bài.
 b) Nội dung: Làm bài tập 1, 2 SCD trang 19 phần vận dụng.
 c) Sản phẩm: Bài làm của HS trong vở ghi hoặc trong máy tính.
 d) Tổ chức thực hiện: HS làm ở nhà, tiết sau GV gọi 2 HS lên bảng chữa bài.

Sản phẩm dự kiến:

1. Lệnh dequeue(Q) chính là phương thức Q.pop(0) xóa phần tử ở vị trí đầu tiên trong hàng đợi Q, khi đó các phần tử còn lại trong hàng đợi sẽ dịch chuyển sang trái 1 vị trí. Nếu trong Q có n phần tử thì tương đương với đoạn chương trình sau:

```
for i in range(len(Q)-1): Q[i]=Q[i+1] Q.pop()
```

Như vậy thực hiện 1 vòng lặp do vậy có độ phức tạp thời gian O(n).

2. Hãy viết các hàm thiết lập kiểu dữ liệu hàng đợi và các thao tác cơ bản với hàng đợi từ mảng dữ liệu T gồm N phần tử T[0], T[1], T[2],...,T[N-1]:

N=100 T=[0]*N

```

backIdx=-1 def Queue():
global backIdx backIdx=-1 return T
def isfullQueue(Q): return backIdx==N-1
def enqueue(Q,x): global backIdx
if isfullQueue(Q):
raise ValueError('Hàng đợi đầy, không thực hiện được') else:
backIdx+=1 Q[backIdx]=x

def dequeue(Q): global backIdx
if isEmptyStack(Q):
raise ValueError('Hàng đợi rỗng') else:
x=Q[0]
backIdx-=1 Q.pop(0) return x
def front(Q):
if isEmptyQueue(Q):
raise ValueError('Hàng đợi rỗng, không thực hiện được') else: return Q[0]

```

GV: Do mảng T cho trước số phần tử của mảng nên cần thiết lập biến backIdx để mô tả chỉ số của phần tử đuôi của hàng đợi. Cần viết thêm hàm isFullQueue(T) để kiểm tra hàng đợi đã đầy chưa. Hàm trả về giá trị True nếu hàng đợi T đầy (backIdx=N-1), ngược lại trả về False.

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phân vận dụng.
- Đọc trước bài 5. Thực hành kiểu dữ liệu ngăn xếp.

BÀI 5. THỰC HÀNH KIỂU DỮ LIỆU NGĂN XẾP VÀ HÀNG ĐỢI

Thời gian thực hiện: 2 tiết thực hành

I. MỤC TIÊU

1. Kiến thức

Biết cách kết hợp các kiểu dữ liệu ngăn xếp và hàng đợi để biểu diễn các loại dữ liệu khác nhau.

2. Năng lực

Năng lực chung:

- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Thực hiện các nhiệm vụ và làm các bài tập giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.

Năng lực Tin học: Có khả năng phân tích yêu cầu của bài toán để sử dụng kiểu dữ liệu ngăn xếp hoặc hàng đợi một cách phù hợp.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.

- Trách nhiệm: Hoàn thành các nhiệm vụ và bài tập theo yêu cầu của GV thông qua các bài làm, thông qua sản phẩm.

II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, Máy tính, Máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước bài 5.

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động khởi động

- a) Mục tiêu: Tạo hứng thú học tập cho HS.
- b) Nội dung: HS trả lời câu hỏi sau:
Em có thể nêu một ví dụ trong thực tế cần sử dụng cả hai kiểu dữ liệu ngăn xếp và hàng đợi?
- c) Sản phẩm: Câu trả lời của HS.
- d) Tổ chức thực hiện: HS thảo luận và đứng tại vị trí trả lời.

2. Hoạt động thực hành

Nhiệm vụ viết chương trình mô phỏng bếp ăn tập thể.

- a) Mục tiêu: HS cài đặt được chương trình mô phỏng bếp ăn tập thể.
- b) Nội dung: Viết chương trình mô phỏng quá trình hoạt động của bếp ăn. Đầu vào là hai tệp input1.inp thể hiện việc đăng kí suất ăn của mọi người và tệp input2.inp thể hiện các suất ăn đã được chuẩn bị trước và đưa vào ngăn xếp. Hãy cho biết có người nào buộc phải đổi suất ăn của mình hay không. Nếu có thì in ra số ID của những người đó. Biết bếp ăn chỉ có 2 loại cơm gà và cơm bò. Mỗi người khi vào xếp hàng và đăng kí món. Trong đó mỗi hàng tương ứng với lượt đăng kí của một người, số đầu tiên là số định danh (ID của người đăng kí), theo sau là loại suất ăn mà người đó chọn. Căn cứ vào tệp đã đăng kí, người quản lí sẽ cho người lao động xếp thành 2 hàng, một hàng gồm toàn bộ những người chọn cơm gà, hàng còn lại gồm những người chọn cơm bò. Do nhà bếp không biết trước thông tin đăng kí của người lao động nên sẽ chuẩn bị sẵn các suất ăn vào 1 ngăn xếp, tổng số lượng suất ăn bằng tổng số người lao động. Nếu suất ăn lấy ra là cơm gà mà người đăng kí là bò sẽ phải chuyển sang ăn cơm gà.
- c) Sản phẩm: Bài làm của HS trên máy tính, vở ghi.

Sản phẩm dự kiến:

```
from Queue import * from Stack import *
def bepan(dkga,dkbo,san): doimon=[]
while not isEmptyStack(san): tmp=pop(san)
if tmp=="Bò":
if not isEmptyQueue(dkbo): dequeue(dkbo)
else:
ID_doi=dequeue(dkga) doimon.append(ID_doi)
elif tmp=="Gà":
if not isEmptyQueue(dkga): dequeue(dkga)
else:
```

```
ID_doi=dequeue(dkbo) doimon.append(ID_doi)
return doimon
dkga=Queue(); dkbo=Queue(); san=Stack() f1=open("input1.inp","r",encoding="utf8") for line
in f1:
id,dk=line.split() if dk=="Gà":
enqueue(dkga,id) elif dk=="Bò":
enqueue(dkbo,id) f1.close()
f2=open("input2.inp","r",encoding="utf8") data=f2.read()
for x in data.split(): push(san,x)
f2.close() doimon=bepan(dkga,dkbo,san) if len(doimon)==0:
print('Không có người nào phải đổi món')
else: print('danh sách những người phải đổi món là:',doimon)
```

d) Tổ chức thực hiện: HS làm bài trên máy tính.

Hoạt động của GV và HS

Bước 1: Chuyển giao nhiệm vụ

GV: Yêu cầu HS đọc đề bài, phân tích yêu cầu bài toán, viết chương trình trên máy tính.

Bước 2: Thực hiện nhiệm vụ

HS: Nghiên cứu SCĐ, làm bài trên máy tính. GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận

HS: Chạy chương trình báo cáo kết quả cho GV.

GV: Kiểm tra kết quả làm của các nhóm.

Bước 4: Kết luận, nhận định

GV: Để viết được chương trình trên phải viết trước các hàm của ngăn xếp và lưu vào máy thu. Chạy chương trình, phân tích để HS hiểu hoạt động của chương trình.

3. Hoạt động luyện tập

a) Mục tiêu: HS thực sự hiểu được chương trình của nhiệm vụ trong SCĐ trang 20.

b) Nội dung: Làm bài tập 1, 2 SCĐ trang 22 phần luyện tập.

c) Sản phẩm: Bài làm của HS trên máy tính, vở ghi.

Sản phẩm dự kiến:

1. Em có nhận xét gì về vị trí của những người phải đổi món ăn:

Chúng ta có thể thấy những người phải đổi món ăn đều là những người nằm ở cuối danh sách. Điều này xuất phát từ tính chất của hàng đợi, những người ở đầu danh sách sẽ được phục vụ trước và được lấy đúng suất ăn mà mình đã chọn. Những người ở cuối

danh sách là những người ở cuối hàng đợi và có thể sẽ phải đổi món ăn tùy theo số suất mà bếp ăn đã chuẩn bị.

2. Chương trình tính thời gian chờ đợi trung bình của mỗi người để nhận được suất ăn của mình, biết thời gian lấy mỗi suất ăn ra khỏi ngăn xếp và đưa cho người lao động là 1 giây.

```
from Stack import *
suatan=Stack()
fi2=open("input2.inp","r",encoding="utf8") data=fi2.read()
for x in data.split(): push(suatan,x)
fi2.close()
tongtime=0 sosuatan=0
while not isEmptyStack(suatan): tmp=pop(suatan) sosuatan+=1 tongtime+=sosuatan
print("Thời gian chờ đợi trung bình của mỗi người là:",tongtime/sosuatan)
```

d) Tổ chức thực hiện: HS làm bài trên máy tính.

Bước 1: Chuyển giao nhiệm vụ

GV: 1. Yêu cầu HS chỉnh sửa chương trình trong nhiệm vụ để đưa ra thời gian chờ đợi trung bình của mỗi người.

Bước 2: Thực hiện nhiệm vụ

HS: Nghiên cứu yêu cầu, phân tích bổ sung câu lệnh vào chương trình. GV: Phân tích bài 1 cần thêm chức năng khi.

Quan sát và trợ giúp HS viết chương trình.

Bước 3: Báo cáo, thảo luận

HS: Chạy chương trình báo cáo kết quả cho GV.

GV: Kiểm tra kết quả làm của các nhóm, gọi nhóm làm đúng chiếu và chạy chương trình lên máy chiếu cho cả lớp quan sát bài làm trên máy tính.

Bước 4: Kết luận, nhận định

GV: Nhận xét, đánh giá và củng cố các câu lệnh cần bổ sung để đưa ra được thời gian chờ đợi trung bình của mỗi người để nhận được suất ăn của mình.

4. Hoạt động vận dụng

a) Mục tiêu: HS sử dụng được kiểu dữ liệu hàng đợi, ngăn xếp viết được chương trình giải bài toán thực tế.

b) Nội dung: Làm bài tập SCD trang 22 phần vận dụng.

c) Sản phẩm: Bài làm của HS trên máy tính, vở ghi.

d) Tổ chức thực hiện: HS làm bài ở nhà, tiết sau GV gọi một HS lên bảng chữa bài trên bảng, một HS viết bài trên máy tính.

GV: Phân tích: Ở bài toán này, chúng ta vẫn sử dụng kiểu dữ liệu hàng đợi để lưu thông tin về suất ăn mà người lao động đăng kí, sử dụng kiểu dữ liệu ngăn xếp để lưu thông tin về các suất ăn mà bếp ăn đã chuẩn bị.

Mỗi lần lấy một suất ăn ra khỏi bếp ăn, chúng ta sẽ duyệt hàng đợi để tìm người phát suất ăn đó. Chúng ta sẽ duyệt cho đến khi tìm được người đăng kí suất ăn trùng với suất ăn vừa được lấy ra khỏi ngăn xếp. Nếu duyệt đến cuối hàng đợi mà vẫn không thể tìm

được người

để phát suất ăn thì chương trình dừng lại. Lúc này số người không nhận được suất ăn là số người còn lại trong hàng đợi. Bài toán này có thể được thực hiện như sau:

Code chương trình:

```
from Queue import * from Stack import * def bepan(dk,suatan):
while not isEmptyStack(suatan): tmp1=pop(suatan) phat_thanh_cong=0
for i in range(0,len(dk)): tmp2=dequeue(dk)
if tmp1==tmp2: phat_thanh_cong=1 break
else:
enqueue(dk,tmp2) if phat_thanh_cong==0:
break return len(dk)
dk=Queue() suatan=Stack()
f1=open("input1.inp","r",encoding="utf8") data=f1.read()
for x in data.split(' '): enqueue(dk,x)
print(dk) f2=open("input2.inp","r",encoding="utf8") data=f2.read()
for x in data.split(' '): push(suatan,x)
print('Số lượng không nhận được suất ăn là:',bepan(dk,suatan))
```

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành bài tập phần vận dụng.
- Đọc trước bài 6. Cây nhị phân.

CHUYÊN ĐỀ 2. TÌM HIỂU CÂY TÌM KIẾM NHỊ PHÂN TRONG SẮP XẾP VÀ TÌM KIẾM

BÀI 6. CÂY NHỊ PHÂN

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Khái niệm cây và các định nghĩa có liên quan.
- Mô hình cây nhị phân.
- Phân loại cây nhị phân.
- Biểu diễn cây nhị phân hoàn chỉnh bằng mảng một chiều.
- Các thuật toán duyệt trên cây nhị phân.

2. Năng lực

- Nêu được khái niệm cây, cây nhị phân.
- Biểu diễn được cây nhị phân bằng mảng một chiều.
- Trình bày và mô phỏng được các phép toán duyệt trước, duyệt giữa, duyệt sau cây nhị phân bằng biểu diễn trực quan.

3. Phẩm chất

- Hình thành ý thức trách nhiệm, tính cẩn thận khi làm việc nhóm, phẩm chất làm việc chăm chỉ, chuyên cần để hoàn thành một nhiệm vụ.

- Phát triển khả năng phân tích hiểu và giải quyết vấn đề.
- Kỹ năng làm việc nhóm, hợp tác trong học tập.
- Nghiêm túc, tập trung, tích cực chủ động.

II. THIẾT BỊ VÀ HỌC LIỆU

- GV: SCD, Slide máy tính, máy chiếu.
- HS: SCD, vở ghi, máy tính.
- Link code trong bài:

https://drive.google.com/drive/folders/1ItrXUa9ajousstHHATXRyuxBD2_xVgcQ?usp=sha_ring

III. TIẾN TRÌNH DẠY HỌC

A. MỞ ĐẦU

1. Hoạt động khởi động

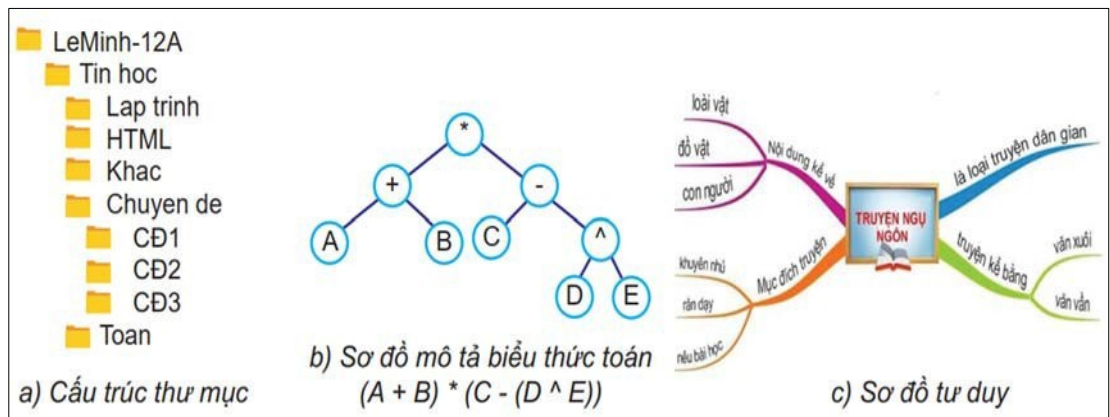
- Mục tiêu: HS nhận ra được một mô hình dữ liệu mới khác với kiểu dữ liệu mảng đã học trong chương trình môn Tin học.
- Nội dung: Quan sát và trả lời Phiếu học tập số 1

Phiếu học tập số 1

Quan sát các sơ đồ biểu diễn thông tin trong hình em có nhận xét gì?

Các sơ đồ này có những điểm gì chung?

Mô hình này có gì khác so với kiểu dữ liệu mảng đã biết? Hãy nêu một số ví dụ thực tế khác c



- Sản phẩm: Trả lời câu hỏi trong phiếu học tập số 1.

- Tổ chức thực hiện:

Chia lớp thành 6 nhóm, mỗi nhóm thảo luận trong thời gian 2 phút và ghi kết quả trả lời ra giấy.

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ	GV yêu cầu HS quan sát và hoàn thành phiếu học tập số 1.
Thực hiện nhiệm vụ	HS tiếp nhận nhiệm vụ, thảo luận nhóm hoàn thành phiếu học tập số 1.

Báo cáo, thảo luận	GV: Thu kết quả trả lời của từng nhóm, cho HS quan sát kết quả của các nhóm và nhận xét.
Kết luận, nhận định	<ul style="list-style-type: none"> - GV nhận xét câu trả lời của các nhóm. - GV chính xác lại các kết quả. <p>1. Quan sát các sơ đồ biểu diễn thông tin trong hình em có nhận xét gì?</p> <p>a) Sơ đồ có dạng cây nằm ngang</p>

	<p>Sơ đồ có dạng cây dốc ngược xuống</p> <p>Sơ đồ có dạng 2 cây ở 2 bên</p> <p>Các sơ đồ này có những điểm gì chung?</p> <p>Đều có hình ảnh của cái cây ở sơ đồ hoặc bộ phận của nó.</p> <p>Mô hình này có gì khác so với kiểu dữ liệu mảng đã biết? Hãy nêu một số ví dụ.</p> <p>Hình ảnh (hình dung về mảng) nó đều nhau còn cây có hình ảnh ngày càng thay đổi.</p> <p>Ví dụ như gia phả dòng họ,...</p>
--	---

B. HÌNH THÀNH KIẾN THỨC MỚI

1. CẤU TRÚC CÂY VÀ CÂY NHỊ PHÂN

Hoạt động 1: Tìm hiểu cấu trúc cây và cây nhị phân

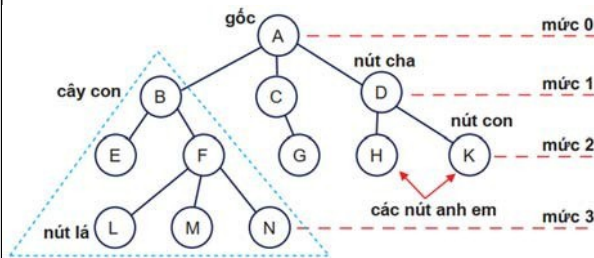
- Mục tiêu: HS nắm được khái niệm và một số định nghĩa của cấu trúc cây tổng quát.
- Nội dung: HS tìm hiểu SCD, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.
- Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
- Tổ chức thực hiện

Sản phẩm dự kiến	Hoạt động của GV và HS
<p>1. CẤU TRÚC CÂY VÀ CÂY NHỊ PHÂN</p> <p>Một số định nghĩa và khái niệm về cây</p> <ul style="list-style-type: none"> - Cây (tree) bao gồm một tập hợp các nút (node) chứa thông tin, có kết nối với nhau gọi là cha – con. - Nút không có nút con được gọi là nút lá (leaf) 	<p>Bước 1: Chuyển giao nhiệm vụ:</p> <p>GV: Chiếu mô hình cây và trình bày và giới thiệu định nghĩa, khái niệm về cây.</p> <p>GV thiết kế phiếu học tập số 2, chia lớp thành 6 nhóm học tập phát phiếu học tập số 2 cho mỗi nhóm và yêu cầu thực hiện.</p>

<p>node).</p> <ul style="list-style-type: none"> - Nút có nút con được gọi là nút trong (inner node) hay nút nhánh (branch node). - Mỗi nút cùng với các nút con bắt đầu từ nút đó tạo thành một cây con (sub tree). - Với mỗi nút của cây, số cạnh cần đi để về tới nút gốc được gọi là mức (level) của nút đó. Mức của nút gốc là 0. - Chiều cao (height) của cây là mức cao nhất của các nút trên cây. 	<p>Phiếu học tập số 2:</p> <p>Em hãy tìm một ví dụ về cây, vẽ nó ra và chỉ ra trên ví dụ đó:</p> <ul style="list-style-type: none"> - Gốc của cây. - Các nút trên cây, nút lá, nút trong. - Quan hệ cha con. - Quan hệ anh em. - Cây con. - Mức của các nút. - Bậc của nút. - Chiều cao của cây.
<p>Sản phẩm dự kiến</p>	<p>Hoạt động của GV và HS</p>

- **Bậc** (degree) của một nút là số các nút con của nó.

- **Cây nhị phân** (binary tree) là cây mà mọi nút có tối đa hai nút con là nút con trái và nút con phải.



Mô hình cây

Ghi nhớ

- Cấu trúc cây bao gồm các nút có quan hệ cha – con. Cây có một nút gốc. Một nút có thể có nhiều nút con. Nút gốc không có nút cha, mỗi nút còn lại chỉ có một nút cha.
- Cấu trúc cây có nhiều ứng dụng trên thực tế và trong Khoa học máy tính.
- Cây nhị phân là cây mà mỗi nút có nhiều nhất hai nút con, được gọi là nút con trái và nút con phải.

Câu hỏi củng cố kiến thức

1. Tìm thêm các ví dụ cấu trúc cây.
2. Vẽ sơ đồ cây cho các biểu thức toán học sau:
 - a) $(x + y) * (x - (y + z)/t)$.
 - b) $x + (y + (z + t)/(u - v))$.
3. Tính chiều cao của các cây trong Trả lời:
 1. Một số ví dụ khác của cây có thể là:
 - Mô hình gia phả dòng họ.
 - Mô hình tổ chức của cơ quan.
 - Mô hình các chương, mục, bài học của một cuốn sách.
 2. Sơ đồ cây tương ứng như sau.

HS: Lắng nghe, thực hiện theo yêu cầu của GV.

Bước 2: Thực hiện nhiệm vụ:

HS: Tìm câu trả lời từ hiểu biết hoặc từ SCD và thảo luận.

GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận:

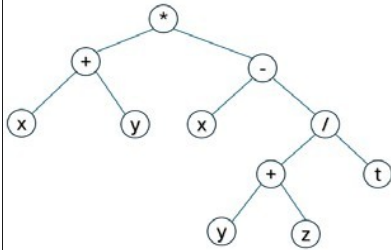
GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau.

HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.

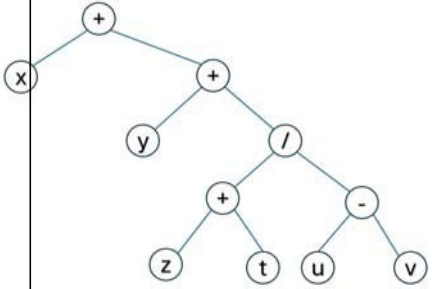
Bước 4: Kết luận, nhận định:

GV chính xác lại các câu trả lời và chính xác lại đáp án.

GV cho HS ghi nhớ các nội dung quan trọng.



a) $(x + y) * (x - (y + z)/t)$



b) $x + (y + (z + t)/(u - v))$

3. Chiều cao của cây: a) 4; b) 5.

2. BIỂU DIỄN CÂY NHỊ PHÂN BẰNG MẢNG MỘT CHIỀU

Hoạt động 2: Tìm hiểu cây nhị phân và cách biểu diễn cây nhị phân

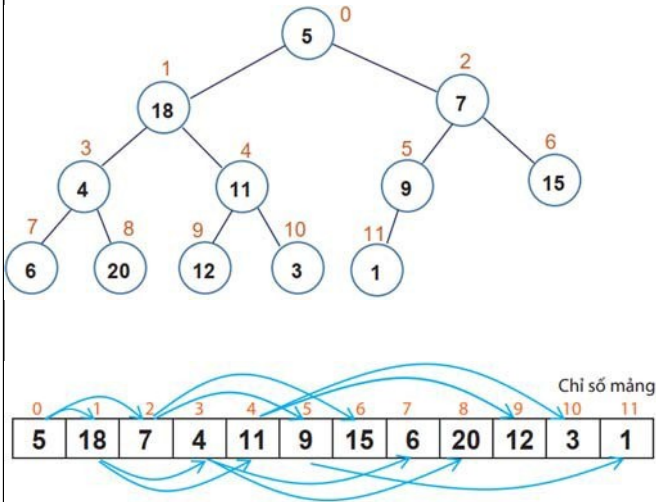
- Mục tiêu: HS phân biệt được định nghĩa cây nhị phân hoàn hảo, hoàn chỉnh và cách biểu diễn cây nhị phân hoàn chỉnh bằng mảng một chiều.
- Nội dung: HS tìm hiểu SCĐ, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.
- Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
- Tổ chức thực hiện

Sản phẩm dự kiến	Hoạt động của GV và HS
<p>2. BIỂU DIỄN CÂY NHỊ PHÂN BẰNG MẢNG MỘT CHIỀU</p> <p>- Phân loại, phân biệt cây nhị phân hoàn hảo và hoàn chỉnh.</p> <p>Cây nhị phân hoàn chỉnh (complete) là cây nhị phân mà có thể đi một lượt tất cả các nút của cây lần lượt theo các mức, bắt đầu từ gốc, mỗi mức duyệt từ trái sang phải mà không bị đứt quãng.</p>	<p>Bước 1: Chuyển giao nhiệm vụ:</p> <p>GV: Yêu cầu HS phân biệt cây nhị phân hoàn chỉnh và cây nhị phân hoàn hảo?</p> <p>GV trình bày cách biểu diễn một cây nhị phân hoàn chỉnh sang mảng một chiều và ngược lại (có thể lấy ví dụ trong SCĐ).</p>

Sản phẩm dự kiến

Hoạt động của GV và HS

- Biểu diễn cây nhị phân hoàn chỉnh sang mảng một chiều và ngược lại, có thể thấy nó đó là một tương ứng 1 - 1



Ghi nhớ

Cây nhị phân hoàn chỉnh có thể được biểu diễn bằng mảng một chiều có số phần tử bằng số nút của cây. Ngược lại, mảng một chiều có thể biểu diễn cây nhị phân hoàn chỉnh.

Câu hỏi củng cố kiến thức

1. Cho mảng $A = [2, 1, 8, 10, 0, 5, 9]$, biểu diễn cây nhị phân hoàn chỉnh. Hãy chỉ ra dãy các nút đi từ nút lá 9 về nút gốc 2.

2. Cho mảng A có 14 phần tử, biểu diễn cây nhị phân hoàn chỉnh. Tính chiều cao của cây nhị phân này.

Trả lời:

1. Cây nhị phân hoàn chỉnh tương ứng

$A = [2, 1, 8, 10, 0, 5, 9]$

có dạng như sau:

Dãy nút đi từ nút lá 9 về gốc là: 9, 8, 2.

GV thiết kế phiếu học tập số 3, chia lớp thành 6 nhóm học tập phát phiếu học tập số 3 cho mỗi nhóm và yêu cầu thực hiện.

Phiếu học tập số 3:

- 1) Lấy một mảng số nguyên và vẽ thành cây nhị phân hoàn chỉnh?
- 2) Vẽ một cây nhị phân hoàn chỉnh và biểu diễn dưới dạng mảng một chiều?

HS: Lắng nghe, thực hiện theo yêu cầu của GV.

Bước 2: Thực hiện nhiệm vụ:

HS: Tìm câu trả lời từ hiểu biết hoặc từ SCD và thảo luận.

GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận:

GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau.

HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.

Bước 4: Kết luận, nhận định:

GV chính xác lại các câu trả lời và chính xác lại đáp án.

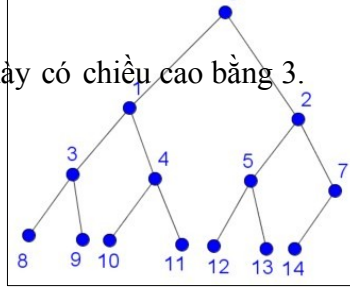
GV cho HS ghi nhớ các nội dung quan trọng.

Sản phẩm dự kiến

Hoạt động của GV và HS

2. Cây nhị phân hoàn chỉnh có 14 nút sẽ có dạng như sau:

Cây nhị phân này có chiều cao bằng 3.



3. CÁC THUẬT TOÁN DUYỆT CÂY NHỊ PHÂN

Hoạt động 3: Tìm hiểu một số thuật toán duyệt cây nhị phân

- Mục tiêu: HS biết và thực hiện được thủ công và bằng lập trình các thuật toán duyệt trước, duyệt giữa và duyệt sau trên cây nhị phân hoàn chỉnh.
- Nội dung: HS tìm hiểu SCD, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.
- Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
- Tổ chức thực hiện

Sản phẩm dự kiến

3. CÁC THUẬT TOÁN DUYỆT CÂY NHỊ PHÂN

a) Duyệt trước (preorder traversal)

Ý tưởng

Hoạt động của GV và HS

Bước 1: Chuyển giao nhiệm vụ:

GV: Thiết kế phiếu học tập số 4, chia lớp thành 6 nhóm học tập phát phiếu học tập số 4 cho mỗi nhóm và yêu cầu thực hiện.

Phiếu học tập số 4:

Mỗi nhóm sẽ đọc sách, trao đổi và trình bày lại một thuật toán duyệt cây nhị phân. Mỗi nhóm sẽ trình bày theo trình tự sau:

- Ý tưởng duyệt.
- Mô tả cách duyệt trên một ví dụ cụ thể.
- Viết đoạn chương trình mô tả thuật toán duyệt.

HS: Lắng nghe, thực hiện theo yêu cầu của GV.

preorder(cây v) # Duyệt trước (gốc-trái-phải) cây v
Nếu cây v khác rỗng
Duyệt nút v # gốc
preorder(cây con trái của nút v) # trái
preorder(cây con phải của nút v) # phải

Ví dụ

```
preorder.py
1 def left(i):
2     return 2*i + 1
3 def right(i):
4     return 2*i + 2
5 def preorder(A, k):
6     if k < len(A) and A[k] != None:
7         print(A[k], end = " ")
8         preorder(A, left(k))
9         preorder(A, right(k))
10 # Chương trình
11 A=[4, 1, 7, 0, 2, 6, 8]
12 preorder(A, 0)
```

Shell %
Python 3.12.2 (C:\Users\Admin\AppData\Local\Programs\Python\Python312\python.exe)
>>> %cd 'C:\Users\Admin\Desktop\soanKHBD1'
>>> %Run preorder.py
4 1 0 2 7 6 8

Sản phẩm dự kiến

Hoạt động của GV và HS

b) Duyệt sau (postorder traversal)

Ý tưởng

```

1 postorder(cây v) # Duyệt sau (trái-phải-gốc) cây v
2     Nếu cây v khác rỗng
3         postorder(cây con trái của nút v) # trái
4         postorder(cây con phải của nút v) # phải
5         Duyệt nút v # gốc
    
```

Ví dụ

```

postorder.py %
1 def left(i):
2     return 2*i + 1
3 def right(i):
4     return 2*i + 2
5 def postorder(A, k):
6     if k < len(A) and A[k] != None:
7         postorder(A, left(k))
8         postorder(A, right(k))
9         print(A[k], end = " ")
10 # Chương trình
11 A=[4, 1, 7, 0, 2, 6, 8]
12 postorder(A, 0)

Shell %
Python 3.12.2 (C:\Users\Admin\AppData\Local\
>>> %Run postorder.py
0 2 1 6 8 7 4
    
```

c) Duyệt giữa (inorder traversal)

```

1 inorder(cây v) # Duyệt giữa (trái-gốc-phải) cây v
2     Nếu cây v khác rỗng
3         inorder(cây con trái của nút v) # trái
4         Duyệt nút v # gốc
5         inorder(cây con phải của nút v) # phải
    
```

Ví dụ

```

inorder.py %
1 def left(i):
2     return 2*i + 1
3 def right(i):
4     return 2*i + 2
5 def inorder(A, k):
6     if k < len(A) and A[k] != None:
7         inorder(A, left(k))
8         print(A[k], end = " ")
9         inorder(A, right(k))
10 # Chương trình
11 A=[4, 1, 7, 0, 2, 6, 8]
12 inorder(A, 0)

Shell %
Python 3.12.2 (C:\Users\Admin\AppData\Local\
>>> %Run inorder.py
0 1 2 4 6 7 8
    
```

Bước 2: Thực hiện nhiệm vụ:

HS: Tìm câu trả lời từ hiểu biết hoặc từ SCD và thảo luận.

GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận:

GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau.

HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.

Bước 4: Kết luận, nhận định:

GV chính xác lại các câu trả lời và chính xác lại đáp án.

GV cho HS ghi nhớ các nội dung quan trọng.

Ghi nhớ

Các thuật toán duyệt cây nhị phân bao gồm duyệt trước, duyệt sau và duyệt giữa.

Câu hỏi củng cố kiến thức

1. Cho mảng [A, B, C, D, E, F, G, H, I, J] biểu diễn một cây nhị phân. Em hãy cho biết thứ tự duyệt các nút của cây này theo phép duyệt trước (gốc-trái-phải).

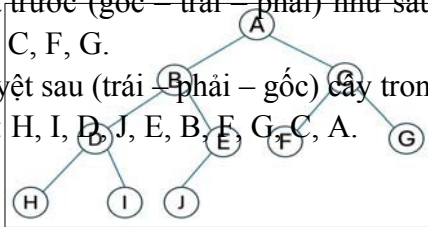
2. Với mảng dữ liệu ở Câu 1, thứ tự duyệt các phần tử sẽ như thế nào nếu thực hiện thuật toán duyệt sau?

Trả lời:

1. Cây đã cho có dạng sau:

Thứ tự duyệt trước (gốc – trái – phải) như sau: A, B, D, H, I, E, J, C, F, G.

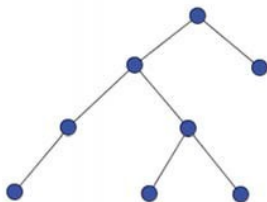
2. Thứ tự duyệt sau (trái – phải – gốc) cây trong ví dụ trên như sau: H, I, D, J, E, B, F, G, C, A.

**C. LUYỆN TẬP – VẬN DỤNG****Hoạt động luyện tập**

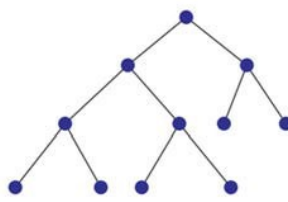
- Mục tiêu: Giúp HS luyện tập khái niệm cây nhị phân hoàn chỉnh, cây nhị phân hoàn hảo.
- Nội dung: GV giao nhiệm vụ cho HS, HS tìm hiểu xem lại và trả lời câu hỏi.
- Sản phẩm: HS hoàn thiện các kĩ năng nhận biết cây nhị phân hoàn hảo, hoàn chỉnh.
- Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập GV đặt câu hỏi cho HS,

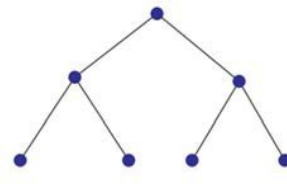
1. Cây nào sau đây là hoàn hảo? Cây nào là hoàn chỉnh? Cây nào không là hoàn hảo và hoàn chỉnh?



a)



b)



c)

2. Cây nhị phân gọi là đầy đủ nếu mỗi nút của nó hoặc là nút lá hoặc có đúng hai nút con. Khẳng định "Cây nhị phân đầy đủ sẽ luôn là hoàn chỉnh hoặc hoàn hảo" là đúng hay sai?

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời. GV quan sát qua trình

HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

GV chính xác hoá lại các nội dung trả lời của HS.

Gợi ý câu trả lời

1. Đáp án như sau:

a) Không hoàn hảo và không hoàn chỉnh. b) Hoàn chỉnh. c) Hoàn hảo.

2. Mệnh đề sai.

Hoạt động vận dụng

a) Mục tiêu: Giúp HS tăng cường kỹ năng cài đặt duyệt cây nhị phân hoàn chỉnh.

b) Nội dung: HS tìm hiểu, thảo luận trả lời theo yêu cầu của GV.

c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.

d) Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập

GV đặt câu hỏi cho HS,

1. Cho mảng một chiều A biểu diễn cây nhị phân hoàn chỉnh T. Viết hàm level(k) trả về mức của nút tương ứng với phần tử A[k] của cây T.

2. Cho cây nhị phân T được biểu diễn bởi mảng một chiều A. Viết các hàm duyệt trước, duyệt giữa và duyệt sau trên cây T.

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời. GV quan sát qua trình HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

GV chính xác hoá lại các nội dung trả lời của HS.

Gợi ý đáp án

1. Hàm level(k) tính mức của nút A[k] có thể viết đơn giản như sau:

```
1 def level(k):
```

```
2     lev = 0
```

```
3     while k > 0:
```

```
4         lev = lev + 1
```

```
5         k = (k-1)//2
```

```
6     return lev
```

2. Chương trình như sau, chú ý các hàm Duyệt_truoc(), Duyệt_giua() và Duyệt_sau() sẽ thực hiện các thuật toán duyệt tương ứng trên cây T.

Binary_tree.py

```
1 def left(i):
```

```

2     return 2*i + 1
3 def right(i):
4     return 2*i+2
5 def parent(i):
6     return (i - 1)//2
7 def preorder(A,k):
8     if k < len(A):
9         print(A[k], end = " ")
10        preorder(A,left(k))
11        preorder(A,right(k))
12 def postorder(A,k):
13     if k < len(A):
14         postorder(A,left(k))
15         postorder(A,right(k))
16         print(A[k], end = " ")
17 def inorder(A,k):
18     if k < len(A):
19         inorder(A,left(k))
20         print(A[k], end = " ")
21         inorder(A,right(k))
22 def Duyet_truoc(A):
23     preorder(A,0)
24 def Duyet_sau(A):
25     postorder(A,0)
26 def Duyet_giua(A):
27     inorder(A,0)

```

D. THÔNG TIN BỔ SUNG

1. Cấu trúc dữ liệu cây nhị phân được phát triển vào đầu những năm 1960 và nhanh chóng trở nên phổ biến bởi sự đơn giản và hiệu quả so với các cấu trúc dữ liệu khác thời bấy giờ. Ban đầu, cây nhị phân được sử dụng trong các ứng dụng như:

- Tìm kiếm và sắp xếp: Cây nhị phân được sử dụng để tìm kiếm và sắp xếp dữ liệu hiệu quả hơn so với các phương pháp tuyến tính như tìm kiếm tuần tự. Thuật toán tìm kiếm nhị phân có độ phức tạp thời gian trung bình là $O(\log n)$, giúp tăng tốc độ tìm kiếm đáng kể khi lượng dữ liệu tăng lên.
- Lưu trữ dữ liệu: Cây nhị phân được sử dụng để lưu trữ dữ liệu trong các cơ sở dữ liệu. Cấu trúc này cho phép truy cập dữ liệu nhanh chóng và hiệu quả thông qua việc so sánh các khoá của các nút.
- Biểu diễn cấu trúc dữ liệu: Cây nhị phân được sử dụng để biểu diễn các cấu trúc dữ liệu phức tạp khác như biểu thức toán học và đồ thị.

Ngày nay, cây nhị phân được sử dụng rộng rãi trong nhiều ứng dụng hiện đại, bao gồm:

- Hệ điều hành: Cây nhị phân được sử dụng để quản lý bộ nhớ và các tài nguyên hệ thống khác. Ví dụ, thuật toán Buddy Allocation sử dụng cây nhị phân để phân bổ hiệu quả bộ nhớ.
- Mạng máy tính: Cây nhị phân được sử dụng để định tuyến dữ liệu trong mạng. Cây định tuyến (routing tree) giúp xác định đường dẫn tối ưu để truyền dữ liệu giữa các thiết bị trong mạng.

- Trí tuệ nhân tạo: Cây nhị phân được sử dụng để biểu diễn tri thức và học máy. Ví dụ, thuật toán ID3 sử dụng cây nhị phân để xây dựng cây quyết định cho các bài toán phân loại.

- Cơ sở dữ liệu: Cây nhị phân được sử dụng để lập chỉ mục dữ liệu trong cơ sở dữ liệu. Cấu trúc B-tree là một biến thể của cây nhị phân được sử dụng phổ biến để lập chỉ mục trong các cơ sở dữ liệu lớn.

2. So sánh cây nhị phân với danh sách liên kết, ngăn xếp và hàng đợi

Cấu trúc dữ liệu	Ưu điểm	Nhược điểm
Cây nhị phân	<ul style="list-style-type: none"> - Tìm kiếm hiệu quả: $O(\log n)$ - Sắp xếp hiệu quả: $O(n \log n)$ - Lưu trữ dữ liệu có thứ tự - Hỗ trợ biểu diễn cấu trúc dữ liệu phức tạp 	<ul style="list-style-type: none"> - Phức tạp hơn danh sách liên kết - Chèn và xoá dữ liệu phức tạp hơn - Chiếm nhiều bộ nhớ hơn danh sách liên kết
Danh sách liên kết	<ul style="list-style-type: none"> - Đơn giản - Chèn và xoá dữ liệu dễ dàng - Tiết kiệm bộ nhớ 	<ul style="list-style-type: none"> - Tìm kiếm chậm: $O(n)$ - Sắp xếp chậm: $O(n^2)$ - Không lưu trữ dữ liệu có thứ tự
Ngăn xếp	<ul style="list-style-type: none"> - Cấu trúc LIFO - Dễ dàng thực hiện các thao tác push và pop - Hiệu quả cho các ứng dụng cần truy cập dữ liệu theo thứ tự ngược 	<ul style="list-style-type: none"> - Không hỗ trợ truy cập ngẫu nhiên - Không hiệu quả cho các ứng dụng cần truy cập dữ liệu theo thứ tự
Hàng đợi	<ul style="list-style-type: none"> - Cấu trúc FIFO - Dễ dàng thực hiện các thao tác enqueue và dequeue 	<ul style="list-style-type: none"> - Không hỗ trợ truy cập ngẫu nhiên
	Hiệu quả cho các ứng dụng cần truy cập dữ liệu theo thứ tự	- Không hiệu quả cho các ứng dụng cần truy cập dữ liệu theo thứ tự ngược

Mỗi cấu trúc dữ liệu có ưu và nhược điểm riêng. Cần lựa chọn cấu trúc dữ liệu phù hợp với yêu cầu cụ thể của từng ứng dụng. Ví dụ:

Cây nhị phân phù hợp cho các ứng dụng cần tìm kiếm và sắp xếp dữ liệu hiệu quả, ví dụ như danh bạ điện thoại.

Danh sách liên kết phù hợp cho các ứng dụng cần chèn và xoá dữ liệu thường xuyên, ví dụ như danh sách các bản nhạc.

Ngăn xếp phù hợp cho các ứng dụng cần truy cập dữ liệu theo thứ tự ngược, ví dụ như undo/redo trong các chương trình soạn thảo văn bản.

Hàng đợi phù hợp cho các ứng dụng cần truy cập dữ liệu theo thứ tự, ví dụ như hàng đợi chờ in ấn.

BÀI 7. CÂY TÌM KIẾM NHỊ PHÂN

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Khái niệm cây tìm kiếm nhị phân.
- Các thuật toán tạo cây tìm kiếm nhị phân, chèn thêm nút vào cây tìm kiếm nhị phân.
- Thuật toán tìm kiếm khoá trên cây tìm kiếm nhị phân.

2. Năng lực

- Trình bày được khái niệm cây tìm kiếm nhị phân.
- Mô phỏng được thuật toán tạo cây tìm kiếm nhị phân từ một tập hợp các số cho trước.
- Biết và thực hiện được thuật toán tìm kiếm một giá trị của cây tìm kiếm nhị phân.

3. Phẩm chất

- Hình thành ý thức trách nhiệm, tính cẩn thận khi làm việc nhóm, phẩm chất làm việc chăm chỉ, chuyên cần để hoàn thành một nhiệm vụ.
- Phát triển khả năng phân tích hiểu và giải quyết vấn đề.
- Kỹ năng làm việc nhóm, hợp tác trong học tập.
- Nghiêm túc, tập trung, tích cực chủ động.

II. THIẾT BỊ VÀ HỌC LIỆU

- GV: SCĐ, Slide máy tính, máy chiếu.
- HS: SCĐ, vở ghi, máy tính.
- Link code trong bài:

https://drive.google.com/drive/folders/1j8BUBf3e6WVSIykIvnFbjF7pj961UdAx?usp=sha_ring

III. TIẾN TRÌNH DẠY HỌC

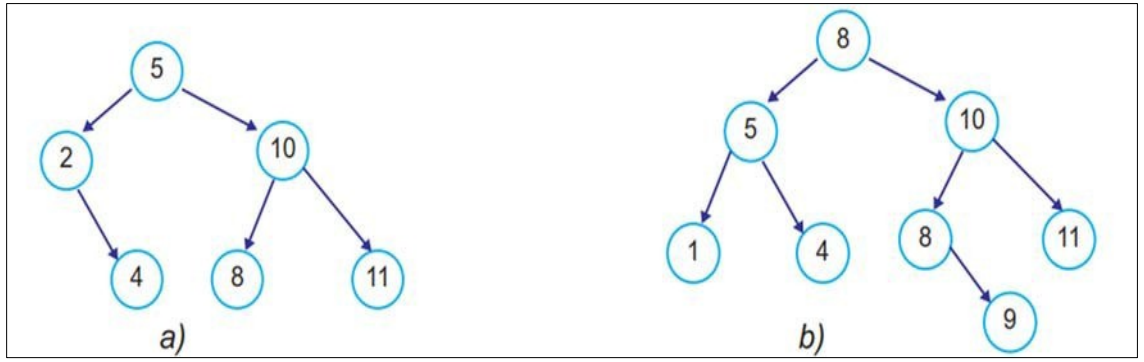
A. MỞ ĐẦU

1. Hoạt động khởi động

- Mục tiêu: HS nhận biết được một dạng đặc biệt của cây nhị phân: Cây tìm kiếm nhị phân (BST).
- Nội dung: Quan sát và trả lời Phiếu học tập số 1

Phiếu học tập số 1

Quan sát các cây nhị phân sau, em có nhận xét gì về giá trị của các nút trên cây?



c) Sản phẩm: Trả lời câu hỏi trong phiếu học tập số 1.

d) Tổ chức thực hiện:

Chia lớp thành 6 nhóm, mỗi nhóm thảo luận trong thời gian 2 phút và ghi kết quả trả lời ra giấy.

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ	GV yêu cầu HS quan sát và hoàn thành phiếu học tập số 1.
Thực hiện nhiệm vụ	HS tiếp nhận nhiệm vụ, thảo luận nhóm hoàn thành phiếu học tập số 1.
Báo cáo, thảo luận	GV: Thu kết quả trả lời của từng nhóm, cho HS quan sát kết quả các nhóm và nhận xét.
Kết luận, nhận định	<ul style="list-style-type: none"> - GV nhận xét câu trả lời của các nhóm. - GV chính xác lại các kết quả. Giá trị của nút con trái luôn nhỏ hơn giá trị nút con phải hay trên một mức (cùng một độ cao) thì giá trị tăng từ trái sang phải.

B. HÌNH THÀNH KIẾN THỨC MỚI

1. CÂY TÌM KIẾM NHỊ PHÂN

Hoạt động 1: Tìm hiểu cấu trúc cây tìm kiếm nhị phân

a) Mục tiêu: HS biết được định nghĩa chính xác của cây tìm kiếm nhị phân và biết được một số cách biểu diễn cây nhị phân tìm kiếm trên máy tính.

b) Nội dung: HS tìm hiểu SCD, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.

c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.

d) Tổ chức thực hiện

Sản phẩm dự kiến	Hoạt động của GV và HS
------------------	------------------------

1. CÂY TÌM KIẾM NHỊ PHÂN

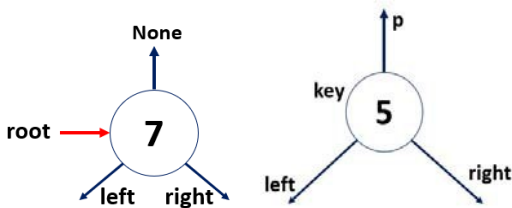
Cây tìm kiếm nhị phân (BST – Binary Search Tree) được tạo ra với mục đích hỗ trợ thuận tiện cho các bài toán tìm kiếm, chèn, xoá, sắp xếp.

a) Mô hình dữ liệu cây nhị phân

Biểu diễn cây nhị phân bằng cấu trúc nút liên kết. Hai cấu trúc sử dụng:

+) Cấu trúc **Node** để thể hiện thông tin từng nút (node) của cây nhị phân.

+) Cấu trúc **Tree** chỉ có thuộc tính **root** sẽ chỉ vào nút gốc của cây nhị phân.



Biểu diễn cây nhị phân bằng mảng một chiều

+) Cây nhị phân tổng quát cần được bổ sung thêm các nút giả (có giá trị None) để tạo thành cây nhị phân hoàn chỉnh đã biến đổi, sau đó sử dụng cách biểu diễn cây hoàn chỉnh.

+) Ví dụ:

Bước 1: Chuyển giao nhiệm vụ:

GV: Thiết kế phiếu học tập số 2. Chia lớp thành 6 nhóm học tập phát phiếu học tập số 2 cho mỗi nhóm và yêu cầu các nhóm thảo luận, tìm hiểu trong SCD thực hiện:

Phiếu học tập số 2:

- 1) Trình bày cách biểu diễn cây nhị phân tổng quát.
- 2) Định nghĩa cây nhị phân tìm kiếm và biểu diễn cây nhị phân tìm kiếm?

HS: Lắng nghe, thực hiện theo yêu cầu của GV.

Bước 2: Thực hiện nhiệm vụ:

HS: Tìm câu trả lời từ hiểu biết hoặc từ SCD và thảo luận.

GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận:

GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau.

Sản phẩm dự kiến

		[5,3,7,None, None,6]
Cây nhị phân tổng quát	Bổ sung nút giả None để thành cây nhị phân hoàn chỉnh	Biểu diễn bằng mảng một chiều

Hoạt động của GV và HS

HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.

Bước 4: Kết luận, nhận định:

GV chính xác lại các câu trả lời và chính xác lại đáp án.

GV cho HS ghi nhớ các nội dung quan trọng.

b) Cây tìm kiếm nhị phân

Cây tìm kiếm nhị phân là cây nhị phân, có hai tính chất quan trọng:

- Khoá của mỗi nút của cây lớn hơn khoá của tất cả các nút thuộc cây con trái và nhỏ hơn khoá của tất cả các nút thuộc cây con phải của nó.

- Hai nút khác nhau có khoá khác nhau.

+ **Cây cân bằng** là cây tìm kiếm nhị phân mà tại mọi nút thì chiều cao của cây con trái và của cây con phải lệch nhau nhiều nhất là 1.

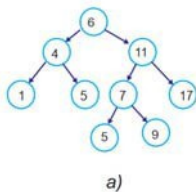
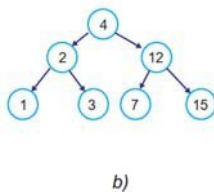
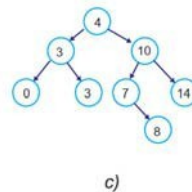
+ **Cây suy biến** là cây tìm kiếm nhị phân có chiều cao lớn nhất, mỗi nút chỉ có tối đa một nút con.

Ghi nhớ

Cây nhị phân tổng quát có thể được cài đặt bằng cấu trúc nút liên kết hoặc bằng mảng một chiều. Cây tìm kiếm nhị phân là cây nhị phân mà tại mọi nút, khoá của nút này lớn hơn khoá của các nút con thuộc cây con trái và nhỏ hơn khoá của các nút con thuộc cây con phải. Khoá của các nút là duy nhất, nghĩa là hai nút khác nhau có khoá khác nhau.

Câu hỏi củng cố kiến thức

1. Trong hình, em hãy cho biết cây nào là cây tìm kiếm nhị phân.



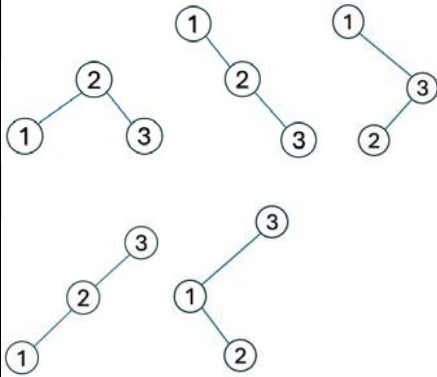
Sản phẩm dự kiến

Hoạt động của GV và HS

2. Từ các khoá 1, 2, 3 có thể tạo ra được bao nhiêu cây tìm kiếm nhị phân? Hãy vẽ sơ đồ mô tả các cây này.

Trả lời:

1. Cây tìm kiếm nhị phân duy nhất là trường hợp b).
2. Có 5 cách thiết lập cây tìm kiếm nhị phân từ các khoá 1, 2, 3.



2. CHÈN MỘT KHOÁ VÀO CÂY TÌM KIẾM NHỊ PHÂN

Hoạt động 2: Thuật toán chèn khoá mới vào cây tìm kiếm nhị phân

- a) Mục tiêu: HS phân biệt được định nghĩa cây nhị phân hoàn hảo, hoàn chỉnh và cách biểu diễn cây nhị phân hoàn chỉnh bằng mảng một chiều.
- b) Nội dung: HS tìm hiểu SCD, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.
- c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
- d) Tổ chức thực hiện

Sản phẩm dự kiến	Hoạt động của GV và HS
<p>2. CHÈN MỘT KHOÁ VÀO CÂY TÌM KIẾM NHỊ PHÂN</p> <p>Ví dụ chèn khoá $v = 7$ vào cây tìm kiếm nhị phân T</p> <p>Cây T</p>	<p>Bước 1: Chuyển giao nhiệm vụ: GV: Trình bày ví dụ trong SCD. Sau đó GV thiết kế phiếu học tập số 3, chia lớp thành 4 nhóm học tập phát phiếu học tập số 3 cho mỗi nhóm và yêu cầu thực hiện. Phiếu học tập số 3: 1. Thực hiện chèn node x $x = 2, 6, 12, 13$</p>

Sản phẩm dự kiến

Hoạt động của GV và HS

Biểu diễn dưới dạng mảng T A = [5, 3, 10, 1, 4, 8, 11, 9]

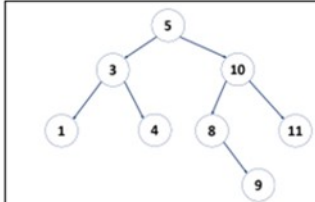
hoatDong2Chen.py ✕

```
1 # Chỉ số con tr
2 def left(k):
3     return 2*k
4 def right(k):
5     return 2*k+1
6 def parent(k):
7     return (k-1)//2
8 # Tạo cây rỗng
9 def Tree():
10     return []
11 # Hàm chèn n
12 def Tree_Insert
13     k = 0
14     while k <
15         if v <
16             elif v
17             k
18         else:
19             re
20             if k >= le
21             T.este
22             T[k] = v
23 # Chương trình
24 A = [5, 3, 10,
25 T = Tree()
26 for x in A:
27     Tree_Insert
28 print(A)
29 x = 7
30 Tree_Insert(T,
31 print(T)
32
```

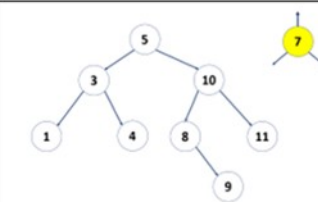
Shell ✕

>> %Run hoatDong2C

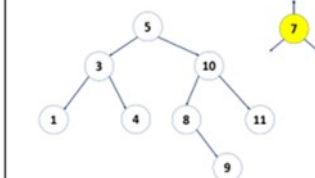
[5, 3, 10, 1, 4, 8, 11, 9]
[5, 3, 10, 1, 4, 8, 11, None, None, None, None, 7, 9]



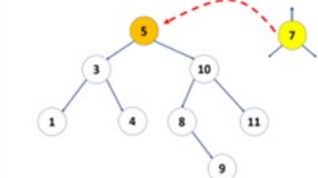
Đây là cây tìm kiếm nhị phân gốc.



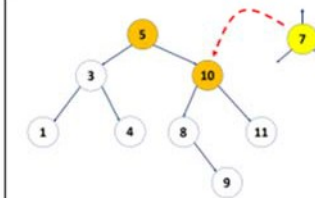
Giả sử cần chèn vào nút mới với khoá 7.



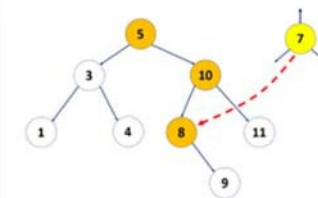
B1. Thiết lập cấu trúc nút mới với khoá 7 sẵn sàng chèn vào cây tìm kiếm nhị phân T.



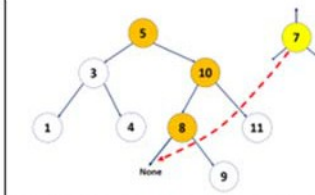
B2. Công việc chèn bắt đầu bằng việc tìm kiếm vị trí để chèn nút mới. Luôn bắt đầu từ nút gốc.



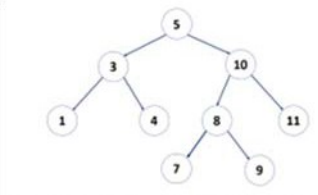
B3. Vì nút gốc có khoá < 7, nên sẽ chuyển tìm tiếp sang nút con phải của nút gốc (với khoá 10).



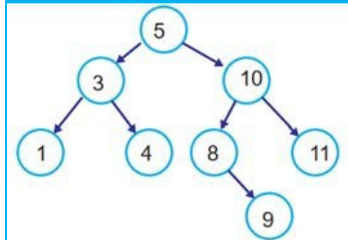
B4. Nút hiện thời có khoá 10 > 7 nên sẽ chuyển tìm tiếp sang nút con trái của nút hiện thời (với khoá 8).



B5. Nút hiện thời là None. Đây chính là vị trí cần chèn nút mới.



B6. Hoàn thành công việc chèn nút mới vào cây T mà vẫn bảo toàn tính chất của cây BST.



2. Cài đặt hàm chèn x vào cây bằng Python

HS: Lắng nghe, thực hiện theo yêu cầu của GV.

Bước 2: Thực hiện nhiệm vụ:

HS: Tìm câu trả lời từ hiểu biết hoặc từ SCD và thảo luận.

GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận:

GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau.

HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.

Bước 4: Kết luận, nhận định:

GV chính xác lại các câu trả lời và chính xác lại đáp án.

GV cho HS ghi nhớ các nội dung quan trọng.

Cài đặt

Ghi nhớ

Quá trình chèn một khoá v vào cây tìm kiếm nhị phân T gồm hai bước:

Bước 1. Tìm vị trí chính xác cần chèn. Nếu gặp khoá v thì dừng chương trình.

Bước 2. Thực hiện thao tác chèn.

Câu hỏi củng cố kiến thức

1. Cho trước dãy các số $A = [10, 1, 2, 11, 8, 15, 20, 9, 0]$. Hãy mô tả và vẽ sơ đồ cây nhị phân biểu diễn dãy số

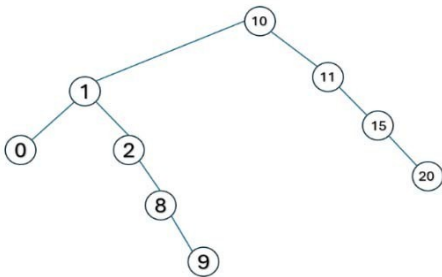
Sản phẩm dự kiến	Hoạt động của GV và HS
------------------	------------------------

trên sau khi thực hiện thao tác chèn như đã mô tả trong hoạt động.

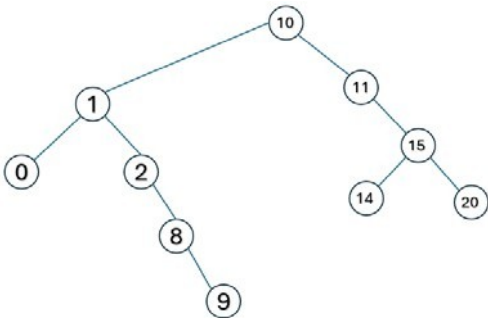
2. Với cây nhị phân đã có ở Câu 1, em hãy vẽ sơ đồ cây sau khi chèn khoá 14 và cho biết vị trí của khoá này ở trong cây.

Trả lời

1. Kết quả cây tìm kiếm nhị phân thu được như sau



2.



3. THUẬT TOÁN TÌM KIẾM TRÊN CÂY TÌM KIẾM NHỊ PHÂN

Hoạt động 3: Tìm hiểu thuật toán tìm kiếm trên cây tìm kiếm nhị phân

- a) Mục tiêu: HS hiểu được thuật toán tìm kiếm khoá trên cây tìm kiếm nhị phân, hiểu được tính ưu việt của việc tìm kiếm rất nhanh trên cây BST.
- b) Nội dung: HS tìm hiểu SCD, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.
- c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
- d) Tổ chức thực hiện

Sản phẩm dự kiến	Hoạt động của GV và HS
3. THUẬT TOÁN TÌM KIẾM TRÊN CÂY TÌM KIẾM NHỊ PHÂN Ý tưởng	Bước 1: Chuyển giao nhiệm vụ: GV: Thiết kế phiếu học tập số 4, chia lớp thành 6 nhóm học tập phát phiếu
Sản phẩm dự kiến	Hoạt động của GV và HS

<div data-bbox="0 177 723 1290"> <p>TimKiemNhiPhanKhuDeQuy.py ✕</p> <pre> 3 return 2 * k + 1 4 def right(i): 5 return 2 * i + 2 6 # Tạo cây rỗng 7 def Tree(): 8 return [] 9 # Hàm chèn khóa vào cây 10 def Tree_Insert(T, v): 11 k = 0 12 while k < len(T) and T[k] != None and T[k] != v: 13 if v < T[k]: 14 k = left(k) 15 elif v > T[k]: 16 k = right(k) 17 else: 18 return 19 if k >= len(T): 20 T.extend([None]*(k - len(T) + 1)) 21 T[k] = v 22 # Hàm tìm kiếm 23 def search(T, k, v): 24 while k < len(T) and T[k] != None and T[k] != v: 25 print(T[k], end = ' ') 26 if v < T[k]: 27 k = left(k) 28 else: 29 k = right(k) 30 if k >= len(T) or T[k] == None: 31 return -1 32 else: 33 return k 34 # Chương trình 35 A = [11, 4, 20, 1, 7, 15, 21, 6, 16] 36 T = Tree() 37 for x in A: 38 Tree_Insert(T, x) 39 print(A) 40 v = 18 41 k = search(T, 0, v) 42 if k >= 0: 43 print("Tìm thấy nút có khoá", T[k]) 44 else: 45 print("Không tìm thấy khoá", v) </pre> </div> <div data-bbox="0 1290 723 1467"> <p>Shell ✕</p> <pre> >> %Run TimKiemNhiPhanKhuDeQuy.py [11, 4, 20, 1, 7, 15, 21, 6, 16] 11 20 15 16 Không tìm thấy khoá 18 </pre> </div> <div data-bbox="0 1467 723 1669"> <p>Shell ✕</p> <pre> python 3.12.2 (C:\Users\Admin\AppData\Local\Programs\Python\Python312\python.exe) >> %Run TimKiemNhiPhanDeQuy.py [11, 4, 20, 1, 7, 15, 21, 6, 16] 11 4 Tìm thấy nút có khoá 7 </pre> </div>	<div data-bbox="723 110 1408 1669"> <p>Tim kiếm nút v từ nút có chỉ số k trên cây tìm kiếm nhị phân T.</p> <p>Nếu tìm thấy thì hàm trả về chỉ số của nút có giá trị v, ngược lại trả về -1.</p> <p>Bắt đầu tìm kiếm từ nút có chỉ số k.</p> <p>Việc tìm kiếm được thực hiện như sau:</p> <ul style="list-style-type: none"> Nếu k nằm ngoài khoảng chỉ số của T hoặc T[k] = None thì trả về -1. Nếu T[k] = v thì dừng tìm kiếm và trả về k. Nếu T[k] ≠ v thì sẽ tìm tiếp trong cây con trái nếu v < T[k], ngược lại tìm tiếp trong cây con phải nếu T[k] < v. <p>HS: Lắng nghe, thực hiện theo yêu cầu của GV.</p> <p>Bước 2: Thực hiện nhiệm vụ: HS: Tìm câu trả lời từ hiểu biết hoặc từ SCĐ và thảo luận. GV: Quan sát và trợ giúp HS.</p> <p>Bước 3: Báo cáo, thảo luận: GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau. HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.</p> <p>Bước 4: Kết luận, nhận định: GV chính xác lại các câu trả lời và chính xác lại đáp án. GV cho HS ghi nhớ các nội dung quan trọng.</p> </div>
<div data-bbox="0 1669 723 1826"> <p>Cài đặt đệ quy</p> <p>Cài đặt không đệ quy</p> </div>	

Ghi nhớ

Thuật toán tìm kiếm khoá K trong cây tìm kiếm nhị phân có độ phức tạp thời gian $O(h)$, với h là chiều cao của cây, hay tương đương trong trường hợp trung bình là $O(\log n)$ với n là số nút của cây.

Sản phẩm dự kiến	Hoạt động của GV và HS
------------------	------------------------

Câu hỏi củng cố kiến thức

1. Khi nào việc tìm kiếm trên cây tìm kiếm nhị phân là: a) nhanh nhất? b) chậm nhất?

2. Cây tìm kiếm nhị phân T được thiết lập bằng cách chèn lần lượt các phần tử 3, 1, 6, 5, 0, 2, 4. Dùng sơ đồ mô tả các bước tìm kiếm giá trị khoá là: a) 4. b) 10. c) 0.

Trả lời:

1. a) Nhanh nhất khi giá trị tìm kiếm trùng với khoá của nút gốc.

b) Chậm nhất khi cây tìm kiếm nhị phân suy biến chỉ có một nhánh, giá trị tìm kiếm trùng với khoá tại nút cuối cùng của nhánh này.

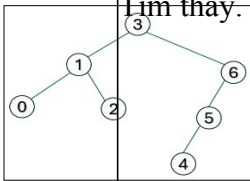
2. Cây tìm kiếm nhị phân được xây dựng có dạng sau:

Các bước tìm kiếm sẽ là: a) $3 \rightarrow 6 \rightarrow 5 \rightarrow 4$

Tìm thấy

b) $3 \rightarrow 6 \rightarrow \text{None}$, Không tìm thấy. c) $3 \rightarrow 1 \rightarrow 0$

Tìm thấy.



C. LUYỆN TẬP – VẬN DỤNG

Hoạt động luyện tập

a) Mục tiêu: Giúp HS luyện tập các tính huống với cây nhị phân, cây tìm kiếm nhị phân.

b) Nội dung: GV giao nhiệm vụ cho HS, HS tìm hiểu xem lại và trả lời câu hỏi.

c) Sản phẩm: HS hoàn thiện các kỹ năng nhận biết cây nhị phân hoàn hảo, hoàn chỉnh.

d) Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập

GV đặt câu hỏi cho HS,

1. Thay đổi thứ tự chèn các phần tử vào cây nhị phân có tạo ra các cây tìm kiếm nhị phân khác nhau hay không? Cho ví dụ minh hoạ.

2. Nếu dãy số được đưa vào cây tìm kiếm nhị phân là tăng dần (hoặc giảm dần) thì cây tìm kiếm nhị phân tương ứng có dạng như thế nào?

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời. GV quan sát qua trình HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

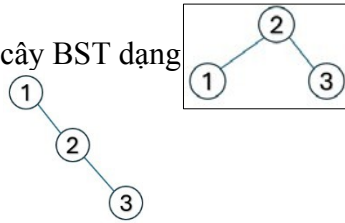
GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

GV chính xác hoá lại các nội dung trả lời của HS.

Gợi ý câu trả lời

1. Có. Ví dụ: Thứ tự chèn 2, 1, 3 sẽ cho ta cây BST dạng



Thứ tự chèn 1, 2, 3 cho ta cây BST dạng:

2. Nếu dãy số được đưa vào cây tìm kiếm nhị phân là tăng dần (hoặc giảm dần) thì cây BST thu được sẽ có dạng suy biến chỉ có một nhánh con duy nhất tính từ gốc.

Hoạt động vận dụng

a) Mục tiêu: Giúp HS tăng cường kỹ năng cài đặt với kỹ thuật đệ quy, kiểm tra cây một cây là cây tìm kiếm nhị phân.

b) Nội dung: HS tìm hiểu, thảo luận trả lời theo yêu cầu của GV.

c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.

d) Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập

GV đặt câu hỏi cho HS,

1. Dữ liệu đầu vào là danh sách học sinh trong lớp và điểm trung bình được cho trong tệp văn bản có dạng như bảng bên.

Data.inp	
Nguyễn Văn Năm	9.3
Bùi Văn Hai	9.0
Trần Quang Bảy	8.8

Viết chương trình đọc tệp dữ liệu đầu vào trên và liên tục thực hiện

a) Nhập thêm vào danh sách học sinh và điểm trung bình.

b) Tìm kiếm với yêu cầu nhập họ tên học sinh và đưa ra kết quả họ tên học sinh, điểm trung bình hoặc thông báo "không tìm thấy".

Chương trình kết thúc khi nhập vào một chuỗi rỗng. Yêu cầu giải bài này bằng cây tìm kiếm nhị phân.

2. Viết hàm chèn khoá v vào cây tìm kiếm nhị phân T sử dụng kỹ thuật đệ quy.

3. Cho trước dãy A bao gồm các số nguyên và các giá trị None. Viết chương trình kiểm tra xem A có phải là biểu diễn của một cây nhị phân hoàn chỉnh đã biến đổi hay không?

Ví dụ:

Dãy [10, 7, 0, 5, None, 3] là biểu diễn của cây nhị phân hoàn chỉnh đã biến đổi. Dãy [1, 6, None, 2, 3, None, 4] không là biểu diễn của cây nhị phân tổng quát nào.

4. Cho trước dãy A bao gồm các số nguyên và các giá trị None. Viết chương trình kiểm tra xem A có phải là biểu diễn của một cây tìm kiếm nhị phân hay không.

Ví dụ:

Dãy [5, 3, 6, None, 4, None, 10] là biểu diễn của cây tìm kiếm nhị phân.

Dãy [2, 1, 5, None, 3, 4, 10] không là biểu diễn của cây tìm kiếm nhị phân (mặc dù dãy này là biểu diễn của cây nhị phân hoàn chỉnh đã biến đổi).

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời. GV quan sát qua trình HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

GV chính xác hoá lại các nội dung trả lời của HS.

Gợi ý đáp án

1. Chương trình hoàn chỉnh của bài vận dụng này như sau:

qlHS.py

```

1     fname = "Data.inp"
2     def NhapDL(fname):
3         f = open(fname,encoding = "UTF-8")
4         A = []
5         for st in f:
6             line = st.split()
7             diem = float(line[-1])
8             hoten = " ".join(line[ : -1])
9             A.append((hoten,diem))
10        f.close()
11        return A
12    # Các hàm mô tả cây BST
13    def left(k):
14        return 2*k+1
15    def right(k):
16        return 2*k+2
17    def parent(k):
18        return (k-1)//2
19    def Tree():
20        return []
21    def Tree_Insert(T,v,d):
22        k = 0
23
24        while k < len(T) and T[k] != None:
25            if v < T[k][0]:
26                k = left(k)
27            elif v > T[k][0]:
28                k = right(k)
29            else:
30                return
31        if k >= len(T):
32            T.extend([None]*(k - len(T) + 1))
33        T[k] = (v,d)
34    def search(T,k,v):
35        if k >= len(T):
36            return -1
37        elif T[k] == None:
38            return -1
39        else:
40            if v == T[k][0]:
41                return k
42            elif v < T[k][0]:
43                return search(T,left(k),v)
44            else:

```

```

44         return search(T, right(k), v)
45 # Chương trình chính
46 def Menu():
47     print("Chọn chức năng:\n1. Nhập thêm \n2. Tìm kiếm \n0. Thoát ")
48     ch = int(input("Chọn chức năng (1,2,0): "))
49     return ch
50
51 A = NhapDL(fname)
52 T = Tree()
53 for (ht, diem) in A:
54     Tree_Insert(T, ht, diem)
55 while True:
56     ch = Menu()
57     if ch == 1:
58         ht = input("Nhập thêm họ tên HS: ")
59         diem = float(input("Nhập điểm HS trên: "))
60         Tree_Insert(T, ht, diem)
61     if ch == 2:
62         ht = input("Nhập họ tên HS cần tìm điểm: ")
63         k = search(T, 0, ht)
64         if k >= 0:
65             print("Học sinh", T[k][0], "có điểm số:", T[k][1])
66         else:
67             print("Không tìm thấy!")
68     if ch == 0:
69         break

```

2. Hàm đệ quy insert(T,v,k) sẽ thực hiện chèn khoá v vào cây tìm kiếm nhị phân T tính từ nút có chỉ số k. Để gọi lệnh chèn khoá v vào cây T thì gọi lệnh insert(T,v,0).

```

1 def insert(T,v,k):
2     if k >= len(T):
3         T.extend([None]*(k - len(T) + 1))
4     if T[k] == None:
5         T[k] = v
6     else:
7         if v < T[k]:
8             insert(T,v,left(k))
9         elif v > T[k]:
10            insert(T,v,right(k))
11        else:
12            return

```

3. Để dãy A có thể là biểu diễn của một cây nhị phân hoàn chỉnh mở rộng thì các điều kiện sau phải thoả mãn:

- Phần tử đầu và cuối của A phải khác None.
- Với mọi chỉ số k, nếu k là nút thật của A và không là gốc thì phải thoả mãn thêm điều kiện: $A[(k - 1)/2]$ khác None.

```

1 def isComplete(A):

```

```

2         """ kiểm tra xem A có phải là dãy tương ứng cây nhị phân hoàn chỉnh mở rộng
hay không """
3         n = len(A)
4         if A[0] == None or A[-1] == None:
5             return False
6         for i in range(1,n):
7             if A[i] != None:
8                 if A[(i-1)//2] == None:
9                     return False
10        return True

```

4. Để dãy A có thể là biểu diễn của một cây tìm kiếm nhị phân hay không thì A phải thoả mãn các điều kiện sau:

(i) A phải là biểu diễn của cây nhị phân hoàn chỉnh đã biến đổi, tức là A phải thoả mãn các điều kiện của bài trên.

(ii) Cây nhị phân được thiết lập bởi A phải là cây tìm kiếm nhị phân.

Điều kiện (i) ở trên chính là yêu cầu của bài vận dụng 3 tương ứng với hàm

isComplete(A). Để dãy A thoả mãn điều kiện (ii) chúng ta cần kiểm tra các điều kiện sau:

(iia) Với mọi nút k của A thì ta phải có: $A[\text{left}(k)] < A[k] < A[\text{right}(k)]$

(iia) Với mọi nút k của A, gọi leftmost(k) là chỉ số của nút nằm ở cây con trái gần nhất với k và gọi rightmost(k) là chỉ số của nút ở cây con phải gần nhất với k, thì phải thoả mãn:

$A[\text{leftmost}(k)] < A[k] < A[\text{rightmost}(k)]$ Hàm leftmost(A,k) thiết lập như sau:

```

1     def leftmost(T,k): # phân tử gần nhất bên trái k...
2         if left(k) < len(T) and T[left(k)] != None:
3
4             Idx = left(k)
5             while right(Idk) < len(T) and T[right(Idk)] != None:
6                 Idx = right(Idk)
7             return Idx
8         else:
9             return -1

```

Hàm rightmost(A,k) xác định như sau:

```

1     def rightmost(T,k): # phân tử gần nhất bên phải k
2         if right(k) < len(T) and T[right(k)] != None:
3             Idx = right(k)
4             while left(Idk) < len(T) and T[left(Idk)] != None:
5                 Idx = left(Idk)
6             return Idx
7         else:
8             return -1

```

Hàm kiểm tra xem dãy A có phải là biểu diễn của cây tìm kiếm nhị phân được thiết lập

như sau:

```

1  def isBST(A):
2      n = len(A)
3      if not isComplete(A):
4          return False
5      for k in range(len(A)):
6          if A[k] != None:
7              if left(k) < n and A[left(k)] != None:
8                  if A[left(k)] >= A[k]:
9                      return False
10             if right(k) < n and A[right(k)] != None:
11                 if A[right(k)] <= A[k]:
12                     return False
13             if leftmost(A,k) >= 0:
14                 if A[leftmost(A,k)] >= A[k]:
15                     return False
16             if rightmost(A,k) >= 0:
17                 if A[rightmost(A,k)] <= A[k]:
18                     return False
19         return True

```

D. THÔNG TIN BỔ SUNG

1. Cấu trúc cây BST sử dụng các nút liên kết đầy đủ có thể thiết lập như sau: Cấu trúc Node mô tả các nút.

```

1  class Node:
2      def init (self,key):
3          self.key = key
4          self.left = None
5          self.right = None
6          self.p = None

```

Cấu trúc cây Tree được định nghĩa như sau:

```

1  class Tree:
2      def init (self):
3          self.root = None

```

2. Phân tích độ phức tạp thời gian của thuật toán chèn nút mới vào cây tìm kiếm nhị phân trong hoạt động 2, trang 33, 34. Cần phân tích kỹ hơn dòng lệnh 10, 11 khi cần bổ sung thêm các phần tử None vào cuối dãy A.

- Trong trường hợp trung bình, giả sử các nút của cây BST phân bố đều trong vùng hình tam giác cân có đỉnh là gốc của cây. Khi đó độ dài cạnh đáy tam giác là $O(n)$. Khi chèn thêm nút mới, số lượng các nút giả None cần chèn tỉ lệ với độ dài cạnh đáy tam giác, suy ra độ phức tạp thời gian trong trường hợp này là $O(n)$.
- Trường hợp xấu nhất, khi cây tìm kiếm nhị phân suy biến thành cây chỉ có một nhánh,

khi đó n = chiều cao của cây. Do vậy cạnh đáy của cây sẽ là 2^{n-1} , do vậy nếu bổ sung thêm phần tử thì số lượng nút giả cần bổ sung sẽ xấp xỉ 2^{n-1} , do đó độ phức tạp thời gian của chương trình sẽ là $O(2^n)$.

Chú ý rằng nếu cây BST được tổ chức theo mô hình nút liên kết thì độ phức tạp thời gian của thao tác chèn nút vào cây sẽ là $O(\log n)$ trong trường hợp trung bình, và bằng $O(n)$ trong trường hợp xấu nhất. Do vậy tổ chức dữ liệu của cây tìm kiếm nhị phân theo mảng một chiều là không tối ưu.

3. Mô hình BST và cách cài đặt kiểu dữ liệu BST trên các ngôn ngữ lập trình khác nhau.

Khái niệm	Định nghĩa và phân loại			
Định nghĩa chung BST.	Tại mọi nút x của cây, tất cả các khoá của các nút thuộc cây con trái của x phải có khoá nhỏ hơn hoặc bằng khoá của x , tất cả các khoá của các nút thuộc cây con phải của x phải có khoá lớn hơn hoặc bằng khoá của nút x . $y.key \leq x.key \leq z.key$			
Phân loại theo yêu cầu khoá trong BST.	Cho phép khoá trùng nhau		Các khoá phải khác nhau từng đôi một	
Phân loại theo kiểu cài đặt cây BST.	Nút liên kết	Mảng một chiều	Nút liên kết	Mảng một chiều
Phân loại chi tiết cho từng kiểu cài đặt.	Tại mọi nút x , mọi nút y trong cây con trái nút z trong cây con phải của x ta có: $y.key < x.key \leq z.key$ hoặc: $y.key \leq x.key < z.key$	$T[i] < T[k] \leq T[j]$ hoặc: $T[i] \leq T[k] < T[j]$	Với mọi nút x , mọi nút y trong cây con trái nút z trong cây con phải của x ta có:	$T[i] < T[k] < T[j]$

Khái niệm	Định nghĩa và phân loại					
				$y.key < x.key < z.key$		
Kiểu nút liên kết	Nút có 3 thuộc tính (left, right, p)	Nút có 2 thuộc tính (left, right)		Nút có 3 thuộc tính (left, right, p)	Nút có 2 thuộc tính (left, right)	
Phân loại	(1)	(2)	(3)	(4)	(5)	(6)

BÀI 8. THỰC HÀNH CÂY TÌM KIẾM NHỊ PHÂN

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Phương pháp cài đặt cây tìm kiếm nhị phân để mỗi nút chứa thông tin của một đối tượng gồm nhiều thuộc tính.

2. Năng lực

- Vận dụng cây tìm kiếm nhị phân vào các bài toán cụ thể để tìm kiếm dữ liệu, đặc biệt trường hợp dữ liệu của các đối tượng gồm nhiều thuộc tính.
- Mô phỏng được thuật toán tạo cây tìm kiếm nhị phân, tìm kiếm một nút trên cây theo giá trị của khoá và cập nhật các thuộc tính của nút đó.

3. Phẩm chất

- Hình thành ý thức trách nhiệm, tính cẩn thận khi làm việc nhóm, phẩm chất làm việc chăm chỉ, chuyên cần để hoàn thành một nhiệm vụ.
- Phát triển khả năng phân tích hiểu và giải quyết vấn đề.
- Kỹ năng làm việc nhóm, hợp tác trong học tập.
- Nghiêm túc, tập trung, tích cực chủ động.

II. THIẾT BỊ VÀ HỌC LIỆU

- GV: SCD, Slide máy tính, máy chiếu.
- HS: SCD, vở ghi, máy tính.
- Link code trong bài:

<https://drive.google.com/drive/folders/1IIIV59nYTt9R8PUrnEwPjYFJAS2NQJI2?usp=sharing>

III. TIẾN TRÌNH DẠY HỌC

A. MỞ ĐẦU

1. Hoạt động khởi động

- Mục tiêu: HS ôn lại các kiến thức đã biết từ hai bài học trước. HS cùng trao đổi và trả lời các câu hỏi của GV đưa ra.
- Nội dung: HS dựa vào hiểu biết để trả lời các câu hỏi.
- Sản phẩm: Từ yêu cầu HS vận dụng sự hiểu biết để trả lời câu hỏi GV đưa ra.
- Tổ chức thực hiện:

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ	<ul style="list-style-type: none"> - GV đặt các câu hỏi, yêu cầu HS trả lời - GV yêu cầu HS đặt câu hỏi nếu có vấn đề cần hỏi <p>Câu 1: Cây nhị phân hoàn hảo có n nút thì chiều cao của nó là bao nhiêu?</p> <p>Câu 2: Tại sao khi sử dụng cây tìm kiếm nhị phân để tìm kiếm dữ liệu, nên cố gắng để cây trở nên cân bằng?</p> <p>Câu 3: Em hãy đề xuất thuật toán xoá một phần tử của cây nhị phân tìm kiếm được cài đặt bằng mảng.</p>
Thực hiện nhiệm vụ	HS tiếp nhận nhiệm vụ, nhớ lại các kiến thức để trả lời câu hỏi.
Báo cáo, thảo luận	GV cho HS nhận xét câu trả lời của nhau, cho HS đưa ra những ý kiến.

Kết luận, nhận định	<ul style="list-style-type: none"> - GV nhận xét câu trả lời của các nhóm. - GV chính xác lại các kết quả. <p>Câu 1: Chiều cao của nó là $\log_2(n+1)$.</p> <p>Câu 2: Trong số các cây nhị phân tìm kiếm thì cây nhị phân tìm kiếm cân bằng có thể đảm bảo số bước tìm kiếm (thường gọi là thời gian tìm kiếm trong thuật toán) ít nhất. Cụ thể với n phần tử dữ liệu khác nhau, thời gian tìm kiếm không lớn hơn $\log(n)$.</p> <p>Câu 3: Để xoá một phần tử của cây nhị phân tìm kiếm, đầu tiên sử dụng thao tác tìm kiếm vị trí phần tử v (đã giới thiệu ở mục 3 Bài 7) sẽ nhận được giá trị chỉ số k trên mảng T, sau đó cập nhật phần tử đó của thành None bằng lệnh $T[k] = \text{None}$.</p>
---------------------	--

B. HÌNH THÀNH KIẾN THỨC MỚI

NHIỆM VỤ: VIẾT CHƯƠNG TRÌNH QUẢN LÝ THỰC ĐƠN

Hoạt động thực hành: Viết chương trình quản lý thực đơn

- Mục tiêu: HS được luyện tập thiết lập và sử dụng cây tìm kiếm nhị phân, áp dụng cho dữ liệu cụ thể, đặc biệt trường hợp quản lý các đối tượng có các thuộc tính khác ngoài khoá.
- Nội dung: HS tìm hiểu SCD, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.
- Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
- Tổ chức thực hiện

Sản phẩm dự kiến	Hoạt động của GV và HS
------------------	------------------------

VIẾT CHƯƠNG TRÌNH QUẢN LÝ THỰC ĐƠN

Em có nhiệm vụ quản lý thực đơn các món ăn hoặc uống (gọi chung là món) của một nhà hàng. Mỗi món đều có tên (không trùng nhau) và giá tiền. Dữ liệu được nhập từ tệp văn bản `menu.inp`, mỗi dòng ứng với một món, có tên và giá tiền cách nhau bởi dấu phẩy.

Em hãy viết chương trình nhập thực đơn từ tệp `menu.inp` và lưu trữ vào cây tìm kiếm nhị phân được cài đặt bằng mảng, sau đó cho phép người dùng:

a) Tra cứu giá theo tên món, ví dụ khi nhập *Bún chả* thì chương trình thông báo giá 60 000.

b) Bổ sung thêm món hoặc cập nhật giá tiền. Ví dụ nhập *Cà phê đen, 35 000* thì chương trình hiểu là cập nhật lại giá *Cà phê đen* thành 35 000, nếu nhập *Nước chanh, 20 000* thì chương trình sẽ bổ sung thêm món *Nước chanh*.

`menu.inp`

```
Cơm suất cá thu sốt, 50000
Cơm suất cá trắm, 40000
Cơm sườn cốt lết, 85000
Phở xào bò, 35000
Phở tái chín, 40000
Bún chả, 60000
Cà phê đen, 30000
Cà phê nâu, 35000
Nước ngọt, 15000
```

Phân tích bài toán

Bước 1. Cài đặt cây tìm kiếm nhị phân.

Bước 1: Chuyển giao nhiệm vụ:

GV: Chia lớp thành 6 nhóm học tập yêu cầu các nhóm tìm hiểu nhiệm vụ, thảo luận đề xuất ý tưởng, triển khai thành mã nguồn, kiểm tra với dữ liệu sách đã cho và minh họa trước lớp.

HS: Lắng nghe, thực hiện theo yêu cầu của GV.

Bước 2: Thực hiện nhiệm vụ:

HS: Tìm câu trả lời từ hiểu biết hoặc từ SCD và thảo luận.

GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận:

GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau.

HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.

Bước 4: Kết luận, nhận định:

GV chính xác lại các câu trả lời và chính xác lại đáp án.

GV cho HS ghi nhớ các nội dung quan trọng.

Sản phẩm dự kiến

Hoạt động của GV và HS

- Hàm `Tree_Insert(T,name,price)` dùng để thêm món (name, price) vào cây tìm kiếm nhị phân T.
 - Hàm `search(T,k,name)` dùng để tìm chỉ số của nút có tên là name, bắt đầu từ chỉ số k trong mảng T.
 - Hàm `Tree_Insert_Update(T,name,price)` gọi hàm `search` để kiểm tra xem món ăn đó đã tồn tại chưa, nếu đã tồn tại thì cập nhật giá, nếu chưa thì gọi hàm `Tree_Insert` để thuận tiện cho việc bổ sung món hoặc cập nhật giá tiền.

Tập `BST_menu.py` gồm các hàm `left(k)`, `right(k)`, `parent(k)` Hàm `Tree()`

`Tree_Insert(T, name, price)` Hàm `search(T, k, name)`

`Tree_Insert_Update(T, name, price)` Tạo tệp `menu.inp` và nhập dữ liệu

Bước 2. Xây dựng chương trình hoàn chỉnh.

Chương trình có bảng chọn ba chức năng tương ứng với thoát chương trình; tra cứu giá theo tên món; cập nhật giá tiền hoặc bổ sung món mới như sau:

0: Thoát chương trình

1: Tra cứu giá

2: Cập nhật hoặc thêm món

Toàn bộ chương trình được đặt trong thư mục

`NVquanlythucdon` bao gồm các tệp

`menu.inp` `BST_menu.py` `quanlythucdon.py`

Kết quả chạy chương trình

```
>>> %Run quanlythucdon.py
0 - Thoát chương trình
1 - Tra cứu giá
2 - Cập nhật hoặc thêm món
Chọn chức năng: |
```

C. LUYỆN TẬP – VẬN DỤNG

Hoạt động luyện tập

- Mục tiêu: Giúp HS rèn luyện các dạng câu hỏi trên cây tìm kiếm nhị phân.
- Nội dung: GV giao nhiệm vụ cho HS, HS tìm hiểu xem lại và trả lời câu hỏi.

c) Sản phẩm: HS hoàn thiện các kỹ năng nhận biết cây nhị phân hoàn hảo, hoàn chỉnh.

d) Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập

GV đặt câu hỏi cho HS,

1. Vẽ cây tìm kiếm nhị phân ứng với tệp menu.inp trong nhiệm vụ thực hành, lưu ý mỗi nút gồm hai thuộc tính *name* và *price*.

2. Mô tả quá trình tra cứu giá tiền món *Bún chả* thực hiện trên cây tìm kiếm nhị phân đã vẽ ở Luyện tập 1.

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời. GV quan sát qua trình HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

GV chính xác hoá lại các nội dung trả lời của HS.

Gợi ý câu trả lời

1. Để vẽ lại cây tìm kiếm nhị phân ứng với tệp *menu.inp*, đầu tiên cần sắp xếp tên các món ăn theo thứ tự từ điển để dễ so sánh các khoá khi chèn các nút vào cây. Thứ tự như sau:

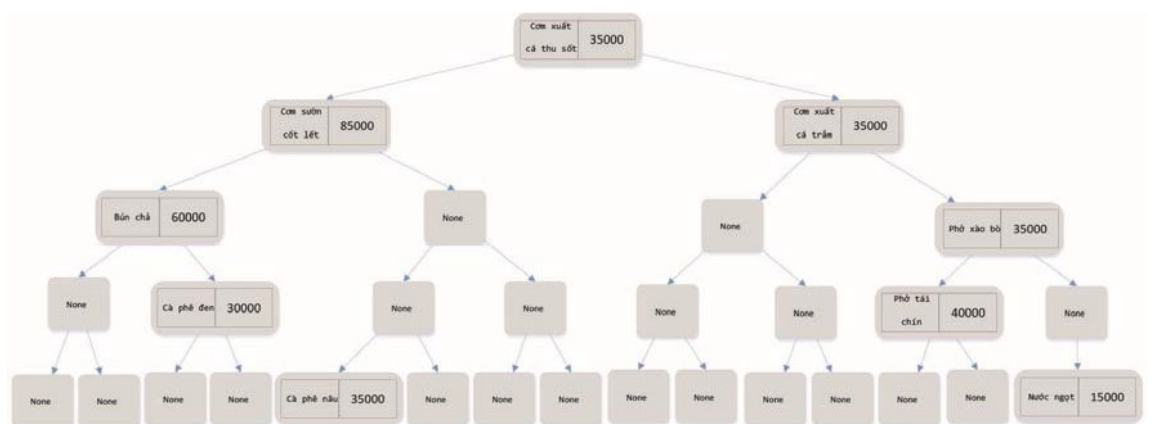
Bún chả, Cà phê đen Cà phê nâu

Cơm sườn cốt lết Cơm xuất cá thu sốt Cơm xuất cá trắm Nước ngọt

Phở tái chín Phở xào bò

Lưu ý: Do cây được cài đặt bằng mảng nên cây nhị phân thu được có thể có nhiều nút giả None và thực tế không dữ liệu con trỏ left và right từ mỗi nút tới các nút con của nó.

Thực hiện thuật toán chèn sẽ thu được cây như sau:



2. Quá trình tìm kiếm bắt đầu bằng lời gọi hàm $\text{search}(T, k = 0, \text{name} = \text{"Bún chả"})$. Vì $\text{"Bún chả"} < T[0][0] = \text{"Cơm xuất cá thu sốt"}$ nên theo thuật toán cần đi tìm theo nhánh trái. Thuật toán gọi hàm $\text{left}(0)$ trả về giá trị 1 ứng với nút $\text{"Cơm sườn cốt lết"}$.

Ở lần gọi hàm search thứ hai: $\text{search}(T, k = 1, \text{name} = \text{"Bún chả"})$ do $\text{"Bún chả"} < T[1][0] = \text{"Cơm sườn cốt lết"}$ nên lại đi theo nhánh bên trái, $\text{left}(1) = 3$ ứng với nút "Bún chả" .

Ở lần gọi hàm search thứ ba, $\text{search}(T, k = 3, \text{name} = \text{"Bún chả"})$ thì giá trị cần tìm Bún

chả đã bằng với khoá của nút hiện tại nên giá trị $k = 3$ là chỉ số của nút cần tìm trong mảng T. Nút này có dữ liệu giá tiền $T[3][1]$ là 60000.

Hoạt động vận dụng

a) Mục tiêu: Giúp HS tăng cường kỹ năng cài đặt với các câu hỏi trên cây nhị phân tìm kiếm.

b) Nội dung: HS tìm hiểu, thảo luận trả lời theo yêu cầu của GV.

c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.

d) Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập

GV đặt câu hỏi cho HS,

1. Sử dụng cây tìm kiếm nhị phân để viết ứng dụng quản lý tài khoản ngân hàng. Mỗi một tài khoản gồm mã tài khoản (duy nhất) và số dư tài khoản. Ứng dụng cho phép thêm tài khoản, sửa số dư tài khoản, tìm kiếm tài khoản theo mã tài khoản.

2. Sử dụng cây tìm kiếm nhị phân để viết chương trình quản lý danh sách học sinh của một trường trung học phổ thông. Mỗi một học sinh gồm các thông tin: mã học sinh (duy nhất), họ tên, ngày sinh. Chương trình cho phép thêm một học sinh vào danh sách với các trường thông tin kể trên, tìm kiếm học sinh theo mã và sửa đổi họ tên và ngày sinh ứng với một mã học sinh.

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời.

GV quan sát qua trình HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

GV chính xác hoá lại các nội dung trả lời của HS.

Gợi ý đáp án

1. Chương trình hoàn chỉnh của bài vận dụng này như sau:

BST_account.py

```

1  # Thiết lập cây BST menu bằng mảng
2  # Cây rỗng = mảng rỗng
3  # Phương án khoá không trùng nhau
4
5  def left(k):
6      return 2*k+1
7  def right(k):
8      return 2*k+2
9  def parent(k):
10     return (k-1)//2
11 def Tree():
12     return []
13
14 def Tree_Insert(T, account, balance):
15     k = 0
16     while k < len(T) and T[k] != None:
17         if account < T[k][0]:

```

```

18         k = left(k)
19     elif account > T[k][0]:
20         k = right(k)
21     else:
22         return
23     if k >= len(T):
24         T.extend([None]*(k - len(T) + 1))
25     T[k] = [account, balance] #Chèn thông tin tài khoản dưới dạng một mảng vào phần
từ k
26
27 def search(T, k, account):
28     if k >= len(T) or T[k] == None:
29         return -1
30     else:
31         if account == T[k][0]:
32             return k
33         elif account < T[k][0]:
34             return search(T, left(k), account)
35         else:
36             return search(T, right(k), account)
37
38 def Tree_Insert_Update(T, account, balance):
39     k = search(T, 0, account)
40     if k == -1:
41         Tree_Insert(T, account, balance)
42         return False #Thể hiện đã thêm tài khoản mới
43     else:
44         T[k][1] = balance #Cập nhật số dư tài khoản
45         return True #Thể hiện đã cập nhật

```

Account_management.py

```

1     from BST_account import *           #Khai báo thư viện cây nhị phân tìm kiếm
2
3     #Khởi tạo cây tìm kiếm
4     T = Tree()
5     options = ["Thoát chương trình", "Tra cứu số dư theo mã tài khoản", "Cập nhật hoặc
thêm tài khoản"]
6     selected = -1
7     while selected != 0:
8         for choice in range(len(options)):
9             print (choice, "-", options[choice] )
10        selected = int(input("Chọn chức năng: "))
11        if selected < 0 or selected > 2:
12            print("Chức năng không hợp lệ, hãy nhập số 0 hoặc 1,2")
13            continue
14        #Kiểm tra lựa chọn và xử lý
15        if selected == 1:
16            item = input("Nhập mã tài khoản: ")
17            k = search(T, 0, item)

```

```

18         if k != -1:
19             print("Số dư ", item, ":", T[k][1], sep="") #T[k][1] là số dư của
tài khoản k
20         else:
21             print("Tài khoản", item, "không có trong thực đơn")
22     elif selected == 2:
23         key = input("Nhập mã tài khoản:")
24         price = int(input("Nhập số dư (số nguyên dương):"))
25         if Tree_Insert_Update(T, key, price):
26             print("Đã cập nhật tài khoản", key)
27         else:
28             print("Đã thêm tài khoản", key, "vào danh mục")

```

2. Chương trình này tương tự chương trình ở bài 1 nhưng sử dụng thêm hàm inorder để in danh sách các học sinh theo thứ tự mã từ nhỏ đến lớn

BST_pupil.py

```

1     # Thiết lập cây BST menu bằng mảng
2     # Cây rỗng = mảng rỗng
3     # Phương án khoá không trùng nhau
4
5     def left(k):
6         return 2*k+1
7     def right(k):
8         return 2*k+2
9     def parent(k):
10         return (k-1)//2
11     def Tree():
12         return []
13
14     def Tree_Insert(T, code, name, birthdate):
15         k = 0
16         while k < len(T) and T[k] != None:
17             if name < T[k][0]:
18                 k = left(k)
19             elif name > T[k][0]:
20                 k = right(k)
21             else:
22                 return
23         if k >= len(T):
24             T.extend([None]*(k - len(T) + 1))
25         T[k] = [code, name, birthdate] #Chèn thông tin học sinh dưới dạng một mảng vào
phần tử k
26
27     def search(T, k, code):
28         if k >= len(T) or T[k] == None:
29             return -1
30         else:
31             if code == T[k][0]:
32                 return k

```

```

33         elif code < T[k][0]:
34             return search(T, left(k), code)
35         else:
36             return search(T, right(k), code)
37
38     def Tree_Insert_Update(T, code, name, birthdate):
39         k = search(T, 0, code)
40         if k == -1:
41             Tree_Insert(T, code, name, birthdate)
42             return False #Thẻ hiện đã thêm học sinh
43         else:
44             T[k][1] = name #Cập nhật tên
45             T[k][2] = birthdate #Cập nhật ngày sinh
46             return True #Thẻ hiện đã cập nhật

```

Pupil_management.py

```

1     from BST_pupil import *           #Khai báo thư viện cây nhị phân tìm kiếm
2     from datetime import date
3
4     #Khởi tạo cây tìm kiếm
5     T = Tree()
6
7     options = ["Thoát chương trình", "Tìm học sinh theo mã", "Cập nhật hoặc thêm học
sinh", "In danh sách học sinh theo mã"]
8     selected = -1
9     while selected != 0:
10         for choice in range(len(options)):
11             print (choice, "-", options[choice] )
12             selected = int(input("Chọn chức năng: "))
13             if selected <0 or selected>3:
14                 print("Chức năng không hợp lệ, hãy nhập số 0 hoặc 1,2,3")
15                 continue
16             #Kiểm tra lựa chọn và xử lý
17             if selected == 1:
18                 code = input("Nhập mã học sinh: ")
19                 k = search(T, 0, code)
20                 if k != -1:
21                     print("Họ tên:", T[k][1], "Ngày sinh:", T[k][2].strftime("%d-%m-
%Y"))
22                 else:
23                     print("Mã", code, "không có trong danh sách")
24             elif selected == 2:
25                 code = input("Nhập mã học sinh:")
26                 name = input("Nhập họ tên:")
27                 birthdate = None
28                 try:
29                     date_components = input('Nhập ngày tháng năm sinh dạng DD- MM-
YYYY: ').split('-')
30                     day, month, year = [int(item) for item in date_components]
31                     birthdate = date(year, month, day)

```

```

31         except:
32             print("Ngày tháng không hợp lệ")
33             continue
34         if Tree_Insert_Update(T, code, name, birthdate):
35             print("Đã cập nhật học sinh mã ", code)
36         else:
37             print("Đã thêm học sinh mã", code, "vào danh sách")

```

D. THÔNG TIN BỔ SUNG

Cây nhị phân (và cây nhị phân tìm kiếm) được cài đặt bằng mảng trong nhiều trường hợp xuất hiện nút giả None dẫn đến lãng phí bộ nhớ. Sau đây là mã nguồn tham khảo thuật toán cây nhị phân tìm kiếm cho dữ liệu thực đơn món ăn sử dụng cấu trúc nút liên kết (tương tự ý tưởng dùng con trỏ trong các ngôn ngữ có con trỏ như C, C++).

BST_pupil.py

```

1     class Node:
2         def  __init__(self, name, price):
3             self.name = name
4             self.price = price
5             self.p = None
6
7             self.left = None
8             self.right = None
9     class Tree:
10         def  __init__(self, name = None, price = None):
11             if name is None:
12                 self.root = None
13             else:
14                 self.root = Node(name, price)
15         def search(x, name):
16             while x is not None and x.name != name:
17                 if name < x.name:
18                     x = x.left
19                 else:
20                     x = x.right
21             return x
22         def Tree_Insert(T,z):
23             y = None
24             x = T.root
25             while x is not None:
26                 y = x
27                 if z.name < x.name:
28                     x = x.left
29                 else:
30                     x = x.right
31             z.p = y
32             if y == None: #cây T rỗng
33                 T.root = z
34             else:

```

```

34         if z.name < y.name:
35             y.left = z
36         else:
37             y.right = z
38     def Tree_Insert_Update(T, name, price):
39         n = search(T.root, name)
40         if n is not None:
41             n.price = price
42             return 1 #Thể hiện đã cập nhật giá
43         else:
44             newNode = Node(name, price)
45             Tree_Insert(T, newNode)
46             return 0 #Thể hiện đã thêm món mới

```

BÀI 9. CÁC THUẬT TOÁN DUYỆT TRÊN CÂY TÌM KIẾM NHỊ PHÂN

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Các thuật toán duyệt trên cây tìm kiếm nhị phân.
- Thuật toán sắp xếp dãy số sử dụng cây tìm kiếm nhị phân.

2. Năng lực

- Thực hiện được các thuật toán duyệt trên cây tìm kiếm nhị phân.
- Tìm kiếm và sắp xếp dãy số dựa trên cây tìm kiếm nhị phân.

3. Phẩm chất

- Hình thành ý thức trách nhiệm, tính cẩn thận khi làm việc nhóm, phẩm chất làm việc chăm chỉ, chuyên cần để hoàn thành một nhiệm vụ.
- Phát triển khả năng phân tích hiểu và giải quyết vấn đề.
- Kỹ năng làm việc nhóm, hợp tác trong học tập.
- Nghiêm túc, tập trung, tích cực chủ động.

II. THIẾT BỊ VÀ HỌC LIỆU

- GV: SCĐ, Slide máy tính, máy chiếu.
- HS: SCĐ, vở ghi, máy tính.
- Link code trong bài:

https://drive.google.com/drive/folders/1o127DERSkCl0mWKakYcqVPbK3hr_pwe_?usp=s_haring

III. TIẾN TRÌNH DẠY HỌC

A. MỞ ĐẦU

1. Hoạt động khởi động

a) Mục tiêu: HS nhớ lại các thuật toán duyệt cây nhị phân đã học trong bài 6, áp dụng thử cho cây tìm kiếm nhị phân. Thông qua trao đổi, GV đưa các câu hỏi để gợi ý học sinh trả lời và nhận ra được các điểm mới khi áp dụng các thuật toán duyệt đã học

cho mô hình cây BST.

b) Nội dung: Quan sát và trả lời Phiếu học tập số 1

Phiếu học tập số 1

Quan sát cây tìm kiếm nhị phân trong Hình, cùng trao đổi, thảo luận các câu hỏi sau:

Nếu thực hiện thuật toán duyệt giữa (trái – gốc – phải) thì nút đầu tiên được duyệt là nút nào?

Nút cuối cùng được duyệt là nút nào?

Thứ tự các nút được duyệt theo thuật toán duyệt giữa sẽ theo thứ tự nào?

Em có nhận xét gì về kết quả đạt được? Giải thích vì sao.

c) Sản phẩm: Trả lời câu hỏi trong phiếu học tập số 1.

d) Tổ chức thực hiện:

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ	GV yêu cầu HS quan sát và hoàn thành phiếu học tập số 1.
Thực hiện nhiệm vụ	HS tiếp nhận nhiệm vụ, thảo luận hoàn thành phiếu học tập số 1.
Báo cáo, thảo luận	GV: Cho HS trả đứng trả lời, nhận xét câu trả lời của nhau.
Kết luận, nhận định	<ul style="list-style-type: none"> - GV nhận xét câu trả lời của các nhóm. - GV chính xác lại các kết quả. <p>a) Node 1 b) Node 19 c) 1 – 3 – 4 – 7 – 8 – 9 – 10 – 12 – 14 – 15 – 19 Giá trị các node được duyệt tăng dần</p>

B. HÌNH THÀNH KIẾN THỨC MỚI

1. CÁC THUẬT TOÁN DUYỆT CÂY TÌM KIẾM NHỊ PHÂN

Hoạt động 1: Tìm hiểu các thuật toán duyệt cây tìm kiếm nhị phân

a) Mục tiêu: HS biết và hiểu được cách thiết lập các thuật toán duyệt cây tìm kiếm nhị phân, biết được sự khác biệt của các thuật toán này so với các thuật toán duyệt đã biết trong bài học 6, biết được ý nghĩa thuật toán duyệt giữa và duyệt ngược trên cây tìm kiếm nhị phân.

b) Nội dung: HS tìm hiểu SCD, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.

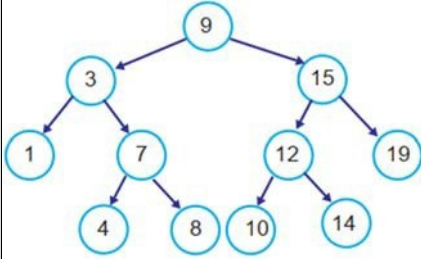
c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.

d) Tổ chức thực hiện

Sản phẩm dự kiến	Hoạt động của GV và HS
------------------	------------------------

1. CÁC THUẬT TOÁN DUYỆT CÂY TÌM KIẾM NHỊ PHÂN

a) Thuật toán duyệt trước



Cây T Cây T biểu diễn bằng mảng T là

$T = [9, 3, 15, 1, 7, 12, 19, \text{None}, \text{None}, 4, 8, 10, 14]$

Thuật toán duyệt trước bắt đầu từ nút k:

```

1 def preorder(T, k):
2     if k < len(T) and T[k] != None:
3         print(T[k], end = " ")
4         preorder(T, left(k))
5         preorder(T, right(k))
  
```

Lệnh duyệt trước toàn bộ cây tìm kiếm nhị phân T là:

```
preorder(T, 0)
```

b) Thuật toán duyệt sau

Thuật toán duyệt sau bắt đầu từ nút k:

```

1 def postorder(T, k):
2     if k < len(T) and T[k] != None:
3         postorder(T, left(k))
4         postorder(T, right(k))
5         print(T[k], end = " ")
  
```

Lệnh duyệt sau toàn bộ cây tìm kiếm nhị phân T là:

```
postorder(T, 0)
```

c) Thuật toán duyệt giữa

Bước 1: Chuyển giao nhiệm vụ:

GV: Thiết kế phiếu học tập số 2. Chia lớp thành 6 nhóm học tập phát phiếu học tập số 2 cho mỗi nhóm và yêu cầu các nhóm thảo luận, tìm hiểu trong SCD thực hiện:

Phiếu học tập số 2:

Câu 1: Phân biệt sự khác nhau giữa các thuật toán duyệt cây nhị phân đã học trong bài 6 và các thuật toán duyệt trong bài học này.
 Câu 2: Duyệt theo thứ tự nào thì duyệt giá trị các nút tăng dần và giảm dần? Hãy cài đặt các thuật toán duyệt với T là cây để các nút tăng dần và giảm dần

HS: Lắng nghe, thực hiện theo yêu cầu của GV.

Bước 2: Thực hiện nhiệm vụ:

HS: Tìm câu trả lời từ hiểu biết hoặc từ SCD và thảo luận.

GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận:

GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau.

<p>Thuật toán duyệt giữa bắt đầu từ nút k:</p> <hr/> <pre> 1 def inorder(T,k): 2 if k < len(T) and T[k] != None: 3 inorder(T,left(k)) 4 print(T[k], end = " ") 5 inorder(T,right(k)) </pre> <hr/> <p>Lệnh duyệt toàn bộ cây tìm kiếm nhị phân T là:</p> <hr/> <pre> inorder(T,0) </pre> <hr/> <p><i>d) Thuật toán duyệt ngược</i></p> <p>Thuật toán duyệt ngược bắt đầu từ nút k:</p> <hr/> <pre> 1 def reverseorder(T,k): 2 if k < len(T) and T[k] != None: 3 reverseorder(T,right(k)) 4 print(T[k], end = " ") 5 reverseorder(T,left(k)) </pre> <hr/> <p>Lệnh duyệt toàn bộ cây tìm kiếm nhị phân T là:</p> <hr/> <pre> reverseorder(T,0) </pre> <hr/> <p>Câu 1:</p> <p>- Trong bài học 6, mô hình cây được học là cây nhị phân hoàn chỉnh, còn trong bài học này thì cây nhị phân tìm kiếm đã được bổ sung thêm các phần tử giả None để trở thành cây nhị phân hoàn chỉnh, do vậy trong quá trình duyệt luôn phải kiểm tra xem nút đang duyệt có phải là nút giả None hay không.</p> <p>Câu 2:</p> <p>- Khi thực hiện các thuật toán duyệt trên cây nhị phân cần chú ý đến hai thuật toán duyệt giữa và duyệt ngược, hai thuật toán này sẽ duyệt lần lượt các phần tử của cây theo thứ tự tăng dần và giảm dần của khoá.</p> <p>- Duyệt theo thứ tự giữa</p>	<p>HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.</p> <p>Bước 4: Kết luận, nhận định: GV chính xác lại các câu trả lời và chính xác lại đáp án. GV cho HS ghi nhớ các nội dung quan trọng.</p>
<p>Sản phẩm dự kiến</p>	<p>Hoạt động của GV và HS</p>

```

inorder.py x
1 def left(i):
2     return 2*i + 1
3 def right(i):
4     return 2*i + 2
5 def inorder(A, k):
6     if k < len(A) and A[k] != None:
7         inorder(A, left(k))
8         print(A[k], end = " ")
9         inorder(A, right(k))
10 # Chương trình
11 T=[9, 3, 15, 1, 7, 12, 19, None, None, 4, 8, 10, 14]
12 inorder(T, 0)

```

```

Shell x
Python 3.12.2 (C:\Users\Admin\AppData\Local\Programs\Python
>>> %Run inorder.py
1 3 4 7 8 9 10 12 14 15 19

```

- Duyệt ngược

```

reverseorder.py x
1 def left(i):
2     return 2*i + 1
3 def right(i):
4     return 2*i + 2
5 def reverseorder(T, k):
6     if k < len(T) and T[k] != None:
7         reverseorder(T, right(k))
8         print(T[k], end = " ")
9         reverseorder(T, left(k))
10 # Chương trình
11 T=[9, 3, 15, 1, 7, 12, 19, None, None, 4, 8, 10, 14]
12 reverseorder(T, 0)

```

```

Shell x
Python 3.12.2 (C:\Users\Admin\AppData\Local\Programs\Python
>>> %Run reverseorder.py
19 15 14 12 10 9 8 7 4 3 1

```

Ghi nhớ

Các thuật toán duyệt chính trên cây tìm kiếm nhị phân bao gồm duyệt trước, duyệt giữa, duyệt sau và duyệt ngược. Thuật toán duyệt giữa sẽ duyệt các nút của cây theo thứ tự tăng dần của khoá. Thuật toán duyệt ngược sẽ duyệt các nút của cây theo thứ tự giảm dần của khoá.

Câu hỏi củng cố kiến thức

1. Cho trước dãy số A = [2,1,9,0,2,1,5]. Tạo cây tìm kiếm nhị phân T từ dãy A và thực hiện thuật toán duyệt giữa trên cây T. Em hãy cho biết kết quả duyệt là dãy các khoá có thứ tự như thế nào.
2. Với cây T như Câu 1, nếu thực hiện thuật toán duyệt ngược thì thứ tự các khoá thể hiện trên màn hình như thế nào?

Trả lời: 1. Các khoá được duyệt theo thuật toán duyệt giữa là: 0, 1, 2, 5, 9. 2. Các khoá được duyệt theo thuật toán duyệt ngược là: 9, 5, 2, 1, 0.	
---	--

2. SẮP XẾP DÃY SỐ BẰNG CÂY TÌM KIẾM NHỊ PHÂN

Hoạt động 2: Tìm hiểu thuật toán sắp xếp dãy số bằng cây tìm kiếm nhị phân

- a) Mục tiêu: HS hiểu được ý tưởng và cách thực hiện thuật toán sắp xếp dãy số mới dựa trên cây nhị phân tìm kiếm, và so sánh với các thuật toán sắp xếp đã biết.
 b) Nội dung: HS tìm hiểu SCĐ, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.
 c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
 d) Tổ chức thực hiện

Sản phẩm dự kiến	Hoạt động của GV và HS
2. SẮP XẾP DÃY SỐ BẰNG CÂY TÌM KIẾM NHỊ PHÂN Ý tưởng sắp xếp sử dụng cây tìm kiếm nhị phân <pre> 1 BSTSort(A) 2 Thiết lập cây tìm kiếm nhị phân T từ dãy A. 3 Thực hiện thuật toán duyệt giữa (inorder) để in các nút theo thứ tự tăng dần, cập nhật các thay đổi vào dãy A. </pre> Hai cách cài đặt Cách 1. Cây tìm kiếm nhị phân với yêu cầu các khoá phải khác nhau từng đôi một. Cách 2. Cây tìm kiếm nhị phân cho phép các khoá trùng nhau. Cài đặt theo cách 2 + Tạo thư mục hoatdong2 , trong thư mục gồm 2 tệp - Tệp module: BST.py	Bước 1: Chuyển giao nhiệm vụ: GV: Thiết kế phiếu học tập số 3, chia lớp thành 6 nhóm học tập phát phiếu học tập số 3 cho mỗi nhóm và yêu cầu thực hiện. Phiếu học tập số 3: - Tìm ý tưởng thiết kế thuật toán sắp xếp dãy dựa trên thuật toán duyệt giữa của cây tìm kiếm nhị phân. - Thực hành cài đặt và triển khai ý tưởng trên dãy A, $A = [5, 2, 1, 4, 3, 9, -1]$ HS: Lắng nghe, thực hiện theo yêu cầu của GV. Bước 2: Thực hiện nhiệm vụ: HS: Tìm câu trả lời từ hiểu biết hoặc từ SCĐ và thảo luận. GV: Quan sát và trợ giúp HS.
Sản phẩm dự kiến	Hoạt động của GV và HS

```

BST.py
1 def left(i):
2     return 2*i + 1
3 def right(i):
4     return 2*i + 2
5 # Hàm chèn khoá v vào cây tìm kiếm nhị phân
6 def Tree_Insert(T, v):
7     k = 0
8     while k < len(T) and T[k] != None:
9         if v < T[k]:
10             k = left(k)
11         else:
12             k = right(k)
13     if k >= len(T):
14         T.extend([None]*(k - len(T) + 1))
15     T[k] = v
16 # Hàm sắp xếp mảng A (duyệt theo thứ tự giữa và đẩy vào A)
17 def new_inorder(T, k, A):
18     if k < len(T) and T[k] != None:
19         new_inorder(T, left(k), A)
20         A.append(T[k])
21         new_inorder(T, right(k), A)

```

- Tên chương trình: sortBST.py

```

BST.py sortBST.py
1 # Chương trình
2 from BST import *
3 def BSTSort(A):
4     T = []
5     for x in A:
6         Tree_Insert(T, x)
7     A.clear()
8     new_inorder(T, 0, A)
9 # Chương trình
10 A = [5, 2, 1, 4, 3, 9, -1]
11 BSTSort(A)
12 print(A)

```

Ghi nhớ

Có thể thiết lập thuật toán sắp xếp danh sách theo kỹ thuật sử dụng cây tìm kiếm nhị phân bằng cách duyệt giữa trên cây tìm kiếm nhị phân được tạo bởi danh sách.

Câu hỏi củng cố kiến thức

- Viết lại hàm **BSTSort(A)** thực hiện sắp xếp dãy số A theo thứ tự tăng dần nhưng kết quả không cập nhật vào A. Hàm trả lại dãy số mới là dãy vừa được sắp xếp (gồm các phần tử của dãy A).
- Nếu không cần cập nhật vào dãy A mà chỉ cần in ra màn hình các phần tử của A theo thứ tự tăng dần thì cần sửa lại chương trình sắp xếp trên như thế nào?

Trả lời

Bước 3: Báo cáo, thảo luận:

GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau.

HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.

Bước 4: Kết luận, nhận định:

GV chính xác lại các câu trả lời và chính xác lại đáp án.

GV cho HS ghi nhớ các nội dung quan trọng.

Sản phẩm dự kiến

Hoạt động của GV và HS

1. Hàm BSTSort(A) thực hiện sắp xếp dãy A nhưng không cập nhật vào A, hàm sẽ trả lại dãy mới là sắp xếp theo thứ tự tăng dần của A.

```

1 def BSTSort(A):
2     B = A.copy()
3     T = Tree()
4     for x in B:
5         insert(T,x,0)
6     B.clear()
7     new_inorder(T,0,B)
8     return B

```

2. Cách làm là sử dụng thuật toán duyệt giữa như sau:

```

1 def BSTSort(A):
2     T = Tree()
3     for x in A:

```

C. LUYỆN TẬP – VẬN DỤNG

Hoạt động luyện tập

- Mục tiêu: Giúp HS luyện tập cài đặt chương trình sắp xếp sử dụng cây nhị phân.
- Nội dung: GV giao nhiệm vụ cho HS, HS tìm hiểu xem lại và trả lời câu hỏi.
- Sản phẩm: HS hoàn thiện các kỹ năng nhận biết cây nhị phân hoàn hảo, hoàn chỉnh.
- Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập

GV đặt câu hỏi cho HS,

- Dựa trên hàm **BSTSort(A)** đã biết, viết chương trình sắp xếp dãy số giảm dần theo kỹ thuật sử dụng cây tìm kiếm nhị phân.
- Thuật toán sắp xếp dãy sử dụng cây tìm kiếm nhị phân có độ phức tạp thời gian là bao nhiêu?

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời. GV quan sát qua trình HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

GV chính xác hoá lại các nội dung trả lời của HS.

Gợi ý câu trả lời

- Sử dụng thuật toán duyệt ngược. Cách làm như sau:

```

1 def new_reverseorder(T,k,A):
2     if k < len(T) and T[k] != None:
3         new_reverseorder(T,left(k),A)
4         A.append(T[k])

```

```

5         new_reverseorder(T,right(k),A)
6     def BSTSort(A):
7         T = Tree()
8         for x in A:
9             insert(T,x,0)
10        A.clear()
11        new_reverseorder(T,0,A)

```

2. Độ phức tạp thời gian của thuật toán sắp xếp dãy A sử dụng cây tìm kiếm nhị phân trong trường hợp trung bình là $O(n^2)$.

Hoạt động vận dụng

- a) Mục tiêu: Giúp HS tăng cường kỹ năng cài đặt tìm max, min trên cây nhị phân tìm kiếm.
- b) Nội dung: HS tìm hiểu, thảo luận trả lời theo yêu cầu của GV.
- c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
- d) Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập

GV đặt câu hỏi cho HS,

1. Dựa trên tính chất của cây tìm kiếm nhị phân, hãy viết hàm **minimum**(T) và **maximum**(T) tính giá trị khoá nhỏ nhất và lớn nhất của cây tìm kiếm nhị phân T.

2. Viết hàm **height**(T) tính chiều cao của cây tìm kiếm nhị phân T.

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời. GV quan sát qua trình HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

GV chính xác hoá lại các nội dung trả lời của HS. [Gợi ý đáp án](#),

1. Các hàm tính minimum và maximum của cây tìm kiếm nhị phân có thể thiết lập như sau: Hàm **minimum**(T,k) sẽ tính chỉ số của phần tử minimum của cây con với gốc là nút k.

```

1     def minimum(T,k):
2
3         while left(k) < len(T) and T[left(k)] != None:
4             k = left(k)
5         return k

```

Khi đó lệnh sau sẽ tính được giá trị minimum của cây T. **minimum**(T,0)

Hàm **maximum**(T,k) sẽ tính chỉ số của phần tử lớn nhất của cây con với gốc là nút k.

```

1     def maximum(T,k):
2         while right(k) < len(T) and T[right(k)] != None:

```

```

3         k = right(k)
4     return k

```

Khi đó lệnh sau sẽ tính được giá trị maximum của cây T. `maximum(T,0)`

2. Thiết lập hàm đệ quy `height(T,k)` tính chiều cao của cây con bắt đầu từ gốc là nút k.

```

1     def height(T,k):
2         if k >= len(T) or T[k] == None:
3             return 0
4         else:
5             return 1 + max(height(T,left(k)),height(T,right(k)))

```

Khi đó muốn tính chiều cao của cây T chỉ cần thực hiện lệnh:

`height(T,0)`

D. THÔNG TIN BỔ SUNG

1. Sau đây là mô tả thuật toán sắp xếp dãy dựa trên cây tìm kiếm nhị phân không cho phép trùng khoá. Bên cạnh mảng T, cần thêm một mảng C dùng để lưu trữ số lần lặp của khoá tương ứng. Như vậy trong mô hình này $C[k] = \text{số lần lặp của khoá } T[k]$. Trong mô hình mới này để thiết lập cây rỗng cần thiết lập $T = []$, $C = []$.

Toàn bộ chương trình mô tả cây và thuật toán sắp xếp dãy như sau.

BST-array.py

```

1     def left(k):
2         return 2*k+1
3     def right(k):
4         return 2*k+2
5     def parent(k):
6         return (k-1)//2
7     def Tree():
8         return [],[]
9
10    def Tree_Insert(T,C,v):
11        k = 0
12        while k < len(T) and T[k] != None:
13            if v < T[k]:
14                k = left(k)
15
16            elif v > T[k]:
17                k = right(k)
18            else:
19                C[k] = C[k] + 1
20        if k >= len(T):
21            T.extend([None]*(k - len(T) + 1))
22            C.extend([None]*(k - len(C) + 1))
23            T[k] = v

```

```

23         C[k] = 1
24
25     def new_inorder(T,C,k,A):
26         if k < len(T) and T[k] != None:
27             new_inorder(T,C,left(k),A)
28             A.extend([T[k]]*C[k])
29             new_inorder(T,C,right(k),A)
30
31     def BSTSort(A):
32         T,C = Tree()
33         for x in A:
34             Tree_Insert(T,C,x)
35         A.clear()
36         new_inorder(T,C,0,A)

```

2. Trong câu hai phần Luyện tập, ta đã biết thuật toán sắp xếp dãy sử dụng cây tìm kiếm nhị phân sử dụng mạng sẽ có độ phức tạp thời gian trung bình là $O(n^2)$. Trong trường hợp xấu nhất thì độ phức tạp thời gian là $O(2^n)$.

3. Nếu cây tìm kiếm nhị phân được cài đặt bằng các nút liên kết thì độ phức tạp thời gian của thuật toán sắp xếp sẽ nhanh hơn nhiều: Trường hợp trung bình là $O(n \log n)$, trường hợp xấu nhất là $O(n^2)$.

4. Bảng sau so sánh thêm các thuật toán cơ bản chính trên các mô hình cây tìm kiếm nhị phân khác nhau.

Mô hình BST và cách cài đặt kiểu dữ liệu BST trên các ngôn ngữ lập trình khác nhau.

Khái niệm	Định nghĩa và phân loại			
Định nghĩa chung BST.	Tại mọi nút x của cây, tất cả các khoá của các nút thuộc cây con trái của x phải có khoá nhỏ hơn hoặc bằng khoá của x, tất cả các khoá của các nút thuộc cây con phải của x phải có khoá lớn hơn hoặc bằng khoá của nút x. $y.key \leq x.key \leq z.key$			
Phân loại theo yêu cầu khoá trong BST.	Cho phép khoá trùng nhau		Các khoá phải khác nhau từng đôi một	
Phân loại theo kiểu	Nút liên kết	Mảng một chiều	Nút liên kết	Mảng một chiều

Khái niệm	Định nghĩa và phân loại			
cài đặt cây BST.				

Phân loại chi tiết cho từng kiểu cài đặt.	Tại mọi nút x, mọi nút y trong cây con trái nút z trong cây con phải của x ta có: $y.key < x.key \leq z.key$ hoặc: $y.key \leq x.key < z.key$	$T[i] < T[k] \leq T[j]$ hoặc: $T[i] \leq T[k] < T[j]$	Với mọi nút x, mọi nút y trong cây con trái nút z trong cây con phải của x ta có: $y.key < x.key < z.key$	$T[i] < T[k] < T[j]$		
Kiểu nút liên kết	Nút có 3 thuộc tính (left, right, p)	Nút có 2 thuộc tính (left, right)		Nút có 3 thuộc tính (left, right, p)	Nút có 2 thuộc tính (left, right)	
Phân loại	(1)	(2)	(3)	(4)	(5)	(6)
Bộ nhớ sử dụng	$O(n)$	$O(n)$	$O(2^n)$	$O(n)$	$O(n)$	$O(2^n)$
Chèn nút mới	$O(\log n)$	$O(\log n)$	$O(n)$, xấu nhất: $O(2^n)$	$O(\log n)$	$O(\log n)$	$O(n)$, xấu nhất: $O(2^n)$
Thiết lập cây từ dãy.	$O(n \log n)$	$O(n \log n)$	$O(n^2)$, xấu nhất: $O(2^n)$	$O(n \log n)$	$O(n \log n)$	$O(n^2)$, xấu nhất: $O(2^n)$
Duyệt cây	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Sắp xếp	$O(n \log n)$	$O(n \log n)$	$O(n^2)$, xấu nhất: $O(2^n)$	$O(n \log n)$	$O(n \log n)$	$O(n^2)$, xấu nhất: $O(2^n)$

BÀI 10. THỰC HÀNH TÌM KIẾM CÂY NHỊ PHÂN

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Ôn luyện kiến thức tổng hợp về cây tìm kiếm nhị phân gồm các thao tác: thêm, tìm kiếm, và duyệt nút gồm nhiều trường thông tin.

2. Năng lực

- Vận dụng cây tìm kiếm nhị phân vào các bài toán cụ thể.
- Sử dụng các phương thức duyệt cây tìm kiếm nhị phân để in danh sách các đối tượng theo thứ tự tăng dần hoặc giảm dần của khoá.

3. Phẩm chất

- Hình thành ý thức trách nhiệm, tính cẩn thận khi làm việc nhóm, phẩm chất làm việc chăm chỉ, chuyên cần để hoàn thành một nhiệm vụ.
- Phát triển khả năng phân tích hiểu và giải quyết vấn đề.

- Kỹ năng làm việc nhóm, hợp tác trong học tập.
- Nghiêm túc, tập trung, tích cực chủ động.

II. THIẾT BỊ VÀ HỌC LIỆU

- GV: SCĐ, Slide máy tính, máy chiếu.
- HS: SCĐ, vở ghi, máy tính.
- Link code trong bài:

https://drive.google.com/drive/folders/1Fe4qU2w3--euN4brTYdb_wkDvLFIBYx1?usp=sharing

III. TIẾN TRÌNH DẠY HỌC

A. MỞ ĐẦU

1. Hoạt động khởi động

- Mục tiêu: HS ôn lại các kiến thức đã biết từ hai bài học trước. HS cùng trao đổi và trả lời các câu hỏi của GV đưa ra.
- Nội dung: HS dựa vào hiểu biết để trả lời các câu hỏi.
- Sản phẩm: Từ yêu cầu HS vận dụng sự hiểu biết để trả lời câu hỏi GV đưa ra.
- Tổ chức thực hiện:

Nhiệm vụ	Cách thức tổ chức
Chuyển giao nhiệm vụ	<ul style="list-style-type: none"> - GV đặt các câu hỏi, yêu cầu HS trả lời - GV yêu cầu HS đặt câu hỏi nếu có vấn đề cần hỏi <p>Với bài toán thực tế quản lý danh bạ điện thoại, làm thế nào để sử dụng các thao tác đó vào cây tìm kiếm nhị phân để thêm, tìm kiếm, hiển thị toàn bộ các liên hệ theo thứ tự sắp xếp của tên liên hệ trong danh bạ?</p>
Thực hiện nhiệm vụ	HS tiếp nhận nhiệm vụ, nhớ lại các kiến thức để trả lời câu hỏi.
Báo cáo, thảo luận	GV cho HS nhận xét câu trả lời của nhau, cho HS đưa ra những ý kiến.

Kết luận, nhận định	GV nhận xét câu trả lời của các nhóm.
	GV chính xác lại các kết quả.
	Mỗi danh bạ là một nút có khóa của cây nhị phân, để có thể làm key

B. HÌNH THÀNH KIẾN THỨC MỚI

NHIỆM VỤ: VIẾT CHƯƠNG TRÌNH QUẢN LÝ DANH BẠ ĐIỆN THOẠI

Hoạt động thực hành: Viết chương trình quản lý danh bạ

- Mục tiêu: HS luyện tập kiến thức tổng hợp về cây tìm kiếm nhị phân để thêm, sửa, tìm kiếm, hiển thị danh sách đối tượng theo thứ tự sắp xếp từ nhỏ đến lớn.
- Nội dung: HS tìm hiểu SCĐ, thảo luận tìm hiểu nội dung kiến thức theo yêu cầu của GV.

- c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
d) Tổ chức thực hiện

Sản phẩm dự kiến	Hoạt động của GV và HS
<p>VIẾT CHƯƠNG TRÌNH QUẢN LÝ DANH BẠ ĐIỆN THOẠI</p> <p>Bài toán</p> <p>Các thiết bị máy tính, điện thoại hiện nay đều tích hợp ứng dụng quản lý danh bạ. Em hãy sử dụng cấu trúc dữ liệu cây tìm kiếm nhị phân để viết ứng dụng quản lý danh bạ đơn giản.</p> <p>Mỗi liên hệ trong danh bạ gồm các thông tin: tên liên hệ (duy nhất), tên đầy đủ, số điện thoại. Khi chạy chương trình, dữ liệu được đọc từ tệp contacts.inp với mỗi dòng ứng với một liên hệ có dạng như hình trên.</p> <p>Các chức năng chính của chương trình:</p> <p>a) Hiển thị danh sách liên hệ theo thứ tự sắp xếp tên theo thứ tự từ điển.</p> <p>b) Tìm kiếm liên hệ theo tên.</p> <p>c) Thêm, sửa các liên hệ.</p>	<p>Bước 1: Chuyển giao nhiệm vụ: GV: Chia lớp thành 6 nhóm học tập yêu cầu các nhóm tìm hiểu nhiệm vụ, thảo luận đề xuất ý tưởng, triển khai thành mã nguồn, kiểm tra với dữ liệu sách đã cho và minh họa trước lớp.</p> <p>HS: Lắng nghe, thực hiện theo yêu cầu của GV.</p> <p>Bước 2: Thực hiện nhiệm vụ: HS: Tìm câu trả lời từ hiểu biết hoặc từ SCD và thảo luận. GV: Quan sát và trợ giúp HS.</p> <p>Bước 3: Báo cáo, thảo luận: GV: Điều khiển hoạt động của các HS, cho HS phát biểu, cho HS nhận xét câu trả lời của nhau. HS: Phát biểu trả lời câu hỏi, đặt câu hỏi nếu chưa rõ.</p>
Sản phẩm dự kiến	Hoạt động của GV và HS

<div data-bbox="258 137 796 358" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> contacts.inp Anh An, Nguyễn Văn An, 0901.000.159 Bố, Nguyễn Văn Hoàng, 0983 000 131 Mẹ, Hoàng Thị Hồng, 0962 000 481 ICTLab Station, Số cố định tại ICTLab, 024 124 000 313 </div> <p>Phân tích bài toán</p> <p>Xây dựng chương trình gồm hai bước:</p> <p>(1) Cài đặt cây tìm kiếm nhị phân bằng mảng để lưu thông tin của các liên hệ, mỗi liên hệ gồm ba thông tin là tên liên hệ, tên đầy đủ, số điện thoại.</p> <p>(2) Xây dựng chương trình hoàn chỉnh đọc dữ liệu từ tệp contacts.inp, lưu dữ liệu vào cây tìm kiếm nhị phân, cho phép người dùng tra cứu, bổ sung, cập nhật danh bạ và in danh bạ theo thứ tự từ điển.</p> <p>Bước 1. Cài đặt cây tìm kiếm nhị phân</p> <p>Tạo tệp: contacts.inp và nhập dữ liệu như mẫu. Xây dựng các hàm:</p> <p>Hàm left(k), right(k), parent(k) Hàm Tree()</p> <p>Hàm Tree_Insert(T, key, fullName, phoneNumber)</p> <p>Hàm search(T, k, name)</p> <p>Hàm Tree_Insert_Update(T, key, fullName, phoneNumber):</p> <p>Hàm inorder(T, k)</p> <p>đặt trong tệp BST_phonebook.py</p> <p>Bước 2. Xây dựng chương trình hoàn chỉnh có menu</p> <p>0: "Thoát chương trình"</p> <p>1: "Hiển thị các liên hệ theo thứ tự từ điển"</p> <p>2: "Tra cứu liên hệ"</p> <p>3: "Cập nhật hoặc thêm một liên hệ"</p> <p>Toàn bộ chương trình được đặt trong thư mục NVquanlydanhba</p> <p>bao gồm các tệp contacts.inp</p>	<p>Bước 4: Kết luận, nhận định:</p> <p>GV chính xác lại các câu trả lời và chính xác lại đáp án.</p> <p>GV cho HS ghi nhớ các nội dung quan trọng.</p>
---	---

BST_phonebook.py quanlydanhba.py

Kết quả chạy chương trình

```
>>> %Run quanlydanhba.py
0 - Thoát chương trình
1 - Hiển thị các liên hệ theo thứ tự từ điển
2 - Tra cứu liên hệ
3 - Cập nhật hoặc thêm một liên hệ
Chọn chức năng: |
```

C. LUYỆN TẬP – VẬN DỤNG

Hoạt động luyện tập

- Mục tiêu: Giúp HS rèn luyện các dạng câu hỏi với cây tìm kiếm nhị phân.
- Nội dung: GV giao nhiệm vụ cho HS, HS tìm hiểu xem lại và trả lời câu hỏi.
- Sản phẩm: HS hoàn thiện các kĩ năng cài đặt chương trình.
- Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập

GV đặt câu hỏi cho HS,

1. Hãy vẽ cây tìm kiếm nhị phân ứng với:

- Dữ liệu tệp [contacts.inp](#) ở trong phần thực hành.
- Từ cây nhận được ở ý a, thêm liên hệ “Anh, Nguyễn Văn Tùng, 0982 000 134”.

2. Tiếp tục với ứng dụng quản lí danh bạ, chức năng hiển thị danh sách liên hệ theo thứ tự từ điển. Do hạn chế của màn hình, mỗi trang chỉ hiển thị được 20 liên hệ. Hãy thêm tính năng in các liên hệ ở trang n bất kì do người dùng nhập vào, điều kiện n nguyên, lớn hơn 0 và nhỏ hơn hoặc bằng tổng số trang có thể hiển thị.

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời. GV quan sát qua trình HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

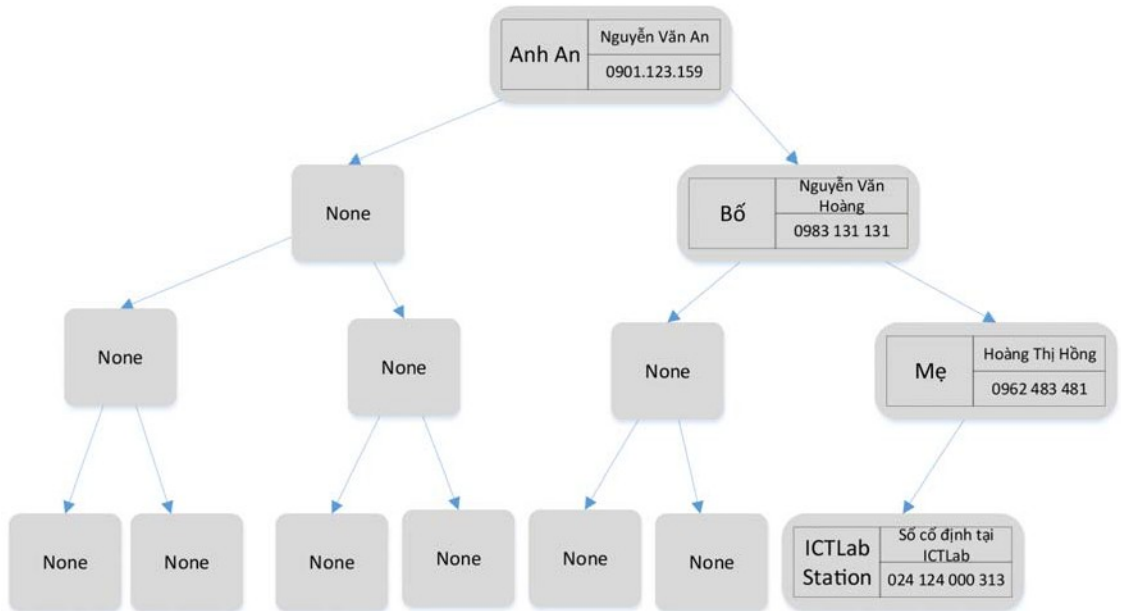
GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

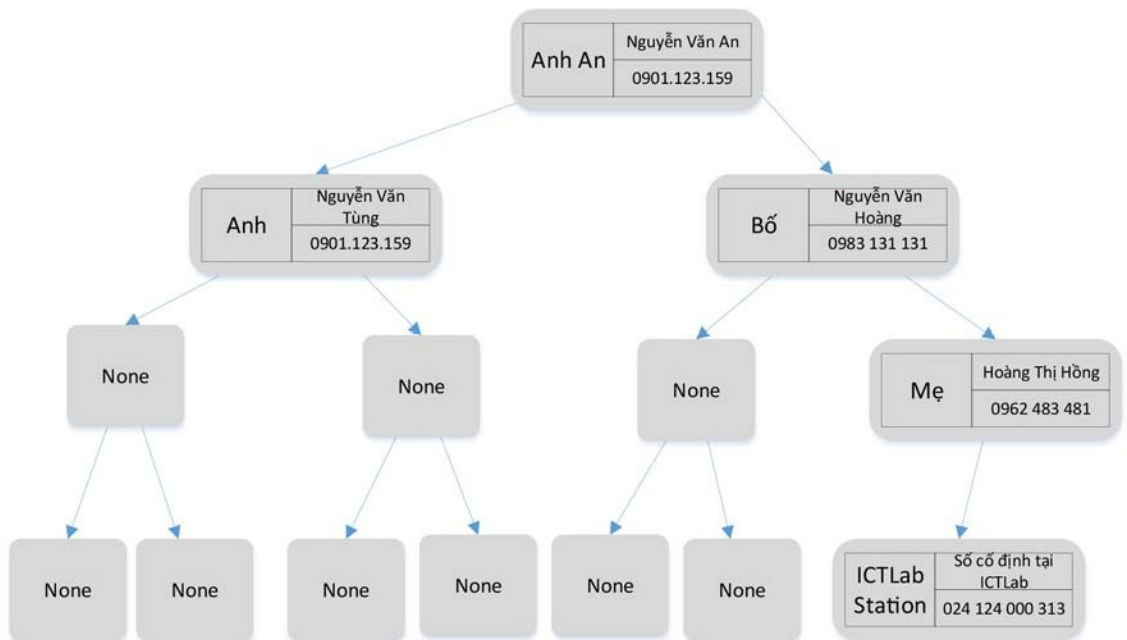
GV chính xác hoá lại các nội dung trả lời của HS.

Gợi ý câu trả lời

1. a) Để vẽ lại cây tìm kiếm nhị phân ứng với tệp [contacts.inp](#) hãy thực hiện dần các bước của thuật toán chèn cho từng liên hệ trong tệp [contacts.inp](#). Do dữ liệu chỉ gồm 4 liên hệ trong đó 3 liên hệ đầu tiên được chèn vào lại đúng với thứ tự khoá của chúng, nút “Bố” và “Mẹ” đều nằm ở nhánh phải của cây. Vì “ICTLab Station” < “Mẹ” nên “ICTLab Station” là nút con trái của “Mẹ”. Cây nhị phân tìm kiếm thu được, không may thay, có dạng “cây suy biến” như hình sau:



b) Khi thêm liên hệ mới có khoá “Anh” vào cây, do nút này có khoá nhỏ hơn khoá ở nút gốc “Anh An” nên liên hệ này được thay vào nút con trái None của nút gốc và ta thu được cây như hình sau:



Để in được các danh bạ ở một trang, cần phải đếm thứ tự các liên hệ khi duyệt cây. Giáo viên có thể gợi ý học sinh sử dụng biến toàn cục count để đếm khi duyệt bằng hàm inorder,

và các biến starting, ending để đánh dấu vị trí đầu tiên và cuối cùng của trang cần tìm.

Mã nguồn tham khảo như sau:

phonebook_paging.py

```
1 from BST_phonebook import * #Khai báo thư viện cây nhị phân tìm kiếm
```

```

2
3 #Khởi tạo cây tìm kiếm và đọc dữ liệu từ tệp
4 T = Tree()
5 f = open("contacts_paging.inp", "r", encoding="utf8") #File chứa nhiều liên hệ
6 lines = f.readlines()
7 for line in lines:
8     l = line.strip()
9     l = l.split(",", maxsplit=3)
10    Tree_Insert(T, l[0], l[1], l[2])
11 f.close()
12
13 count = 0
14 starting = 0
15 ending = 0
16 def inorder_paging(T, k):
17     global count
18     global starting
19     global ending
20     if(count>ending):
21         return
22     if k < len(T) and T[k] != None:
23         inorder_paging(T,left(k))
24         count += 1
25         if(count>= starting):
26             print(T[k][0], "|", T[k][1], "|", T[k][2])
27             inorder_paging(T,right(k))
28
29 # Hiển thị các chức năng chương trình và xử lý theo lựa chọn
30 options = ["Thoát chương trình", "Hiển thị các liên hệ theo thứ tự alphabet", "Tra cứu liên
hệ", "Cập nhật hoặc thêm một liên hệ"]
31 selected = -1 #Chức năng người dùng chọn lựa từ menu, được khởi tạo bằng
-1
32 while selected != 0:
33     for choice in range(len(options)):
34         print (choice, "-", options[choice] )
35     selected = int(input("Chọn chức năng: "))
36     if selected < 0 or selected > 4:
37         print("Chức năng không hợp lệ, hãy nhập số 0 đến 4")
38         continue
39     if selected == 1: #In danh bạ
40         page = int(input("In danh bạ theo thứ tự alphabet. Hãy nhập trang muốn in: "))
41         page_size = 20 #Một trang có 20 liên hệ


---


42         starting = (page - 1)*page_size + 1
43         ending = starting + page_size
44         count = 0
45         inorder_paging(T, 0)
46     elif selected == 2: #Tìm liên hệ
47         item = input("Nhập tên liên hệ: ")
48         k = search(T, 0, item)
49         if k >= 0:
50             print("Tên đầy đủ:", T[k][1], "| Số điện thoại:", T[k][2])

```

```

51         else:
52             print("Tên liên hệ", item, "không có trong danh bạ")
53     elif selected == 3: #Thêm hoặc cập nhật liên hệ
54         key = input("Nhập tên liên hệ: ")
55         fullName = input("Nhập tên đầy đủ: ")
56         phoneNumber = input("Nhập số điện thoại: ")
57         if Tree_Insert_Update(T, key, fullName, phoneNumber):
58             print("Đã cập nhật liên hệ", key)
59         else:
60             print("Đã thêm liên hệ", key, "vào danh bạ")

```

Hoạt động vận dụng

- a) Mục tiêu: Giúp HS tăng cường kỹ năng cài đặt với các câu hỏi trên cây nhị phân tìm kiếm.
- b) Nội dung: HS tìm hiểu, thảo luận trả lời theo yêu cầu của GV.
- c) Sản phẩm: HS hoàn thành tìm hiểu kiến thức.
- d) Tổ chức thực hiện

Bước 1: Chuyển giao nhiệm vụ học tập

GV đặt câu hỏi cho HS,

1. Sử dụng cây tìm kiếm nhị phân để viết chương trình quản lý danh sách học sinh của một lớp. Thông tin mỗi học sinh gồm mã (duy nhất), tên đầy đủ, ngày sinh. Chương trình cho phép thêm mới thông tin các học sinh, in danh sách sắp xếp theo mã từ nhỏ đến lớn và từ lớn đến nhỏ, tìm kiếm học sinh theo mã.

2. Sử dụng cây tìm kiếm nhị phân để hiển thị các món trong tệp menu.inp ở Bài 8 theo thứ tự giá tiền tăng dần. Mỗi dòng in ra gồm tên món và giá tiền. Nếu có hai hoặc nhiều món cùng giá tiền thì các món đó được hiển thị theo thứ tự xuất hiện trong tệp menu.inp.

Bước 2: HS thực hiện nhiệm vụ học tập

HS tiếp nhận, thực hiện nhiệm vụ, thảo luận, đưa ra câu trả lời. GV quan sát qua trình HS thảo luận, hỗ trợ khi HS cần.

Bước 3: Báo cáo kết quả hoạt động và thảo luận

GV cho HS trả lời.

Bước 4: Đánh giá kết quả, thực hiện nhiệm vụ học tập

GV chính xác hoá lại các nội dung trả lời của HS.

Gợi ý đáp án

1. Bài tập này luyện kỹ năng duyệt cây nhị phân tìm kiếm theo một thứ tự. Để duyệt các nút theo khoá từ nhỏ đến lớn, cần sử dụng thuật toán duyệt giữa (cây con trái trước, nút gốc, cây con phải). Để duyệt các nút theo khoá từ lớn đến nhỏ, cũng sử dụng thuật toán duyệt giữa nhưng với cây con phải trước, nút gốc, rồi mới đến cây con trái. Sửa chương trình, thêm vào mã nguồn câu 2 vận dụng Bài 8 các đoạn mã sau:

BST_pupil.py

```

46 # Tiếp đoạn mã nguồn file BST_pupil ở bài 8
47 def inorder_leftright(T,k):
48     if k < len(T) and T[k] != None:

```

```

49         inorder_lef2right(T, left(k))
50         print("Mã", T[k][0], "Họ tên:", T[k][1], "Ngày sinh:", T[k][2].strftime("%d-
%m-%Y"))
51         inorder_lef2right(T, right(k))
52
53     def inorder_right2left(T,k):
54         if k < len(T) and T[k] != None:
55             inorder_right2left(T, right(k))
56             print("Mã", T[k][0], "Họ tên:", T[k][1], "Ngày sinh:", T[k][2].strftime("%d-
%m-%Y"))
57             inorder_right2left(T, left(k))

```

Pupil_management.py

```

1     from BST_pupil import *           #Khai báo thư viện cây nhị phân tìm kiếm
2     from datetime import date
3
4     # Khởi tạo cây tìm kiếm
5     T = Tree()
6     options = ["Thoát chương trình", "Tìm học sinh theo mã", "Cập nhật hoặc thêm học sinh",
7               "In danh sách học sinh theo mã từ nhỏ đến lớn", "In danh
sách học sinh theo mã từ lớn đến nhỏ"]
8     selected = -1
9     while selected != 0:
10         for choice in range(len(options)):
11             print (choice, "-", options[choice] )
12             selected = int(input("Chọn chức năng: "))
13             if selected <0 or selected>4:
14                 print("Chức năng không hợp lệ, hãy nhập số 0 hoặc 1,2,3")
15                 continue
16             #Kiểm tra lựa chọn và xử lý
17             if selected == 1:
18                 code = input("Nhập mã học sinh: ")
19                 k = search(T, 0, code)
20                 if k != -1:
21                     print("Họ tên:", T[k][1], "Ngày sinh:", T[k][2].strftime("%d-%m-%Y"))
22
23             else:
24                 print("Mã", code, "không có trong danh sách")
25             elif selected == 2:
26                 code = input("Nhập mã học sinh:")
27                 name = input("Nhập họ tên:")
28                 birthdate = None
29                 try:
30                     date_components = input('Nhập ngày tháng năm sinh dạng DD- MM-
YYYY: ').split('-')
31                     day, month, year = [int(item) for item in date_components]
32                     birthdate = date(year, month, day)
33                 except:
34                     print("Ngày tháng không hợp lệ")
35                     continue

```

```

35         if Tree_Insert_Update(T, code, name, birthdate):
36             print("Đã cập nhật học sinh mã ", code)
37         else:
38             print("Đã thêm học sinh mã", code, "vào danh sách")
39     elif selected == 3:
40         inorder_leftright(T,0)
41     elif selected == 4:
42         inorder_rightleft(T,0)

```

2. Do yêu cầu của đề bài là sắp xếp các món ăn theo giá sử dụng cây tìm kiếm nhị phân nên cần sử dụng giá là khoá của cây. Tuy nhiên do giá không duy nhất nên cần sửa hàm `Tree_Insert` để cho phép chèn các khoá giống nhau, nút nào chèn sau nằm ở bên phải nút chèn trước. Mã nguồn tham khảo như sau:

BST_menu.py

```

1     def left(k):
2         return 2*k+1
3     def right(k):
4         return 2*k+2
5     def parent(k):
6         return (k-1)//2
7     def Tree():
8         return []
9     def Tree_Insert(T, name, price):
10        k = 0
11        while k < len(T) and T[k] != None:
12            if name < T[k][0]:
13                k = left(k)
14            elif name >= T[k][0]:
15                k = right(k)
16        if k >= len(T):
17            T.extend([None]*(k - len(T) + 1))
18            T[k] = [name, price] #Chèn thông tin món ăn
19
20 def inorder(T, k):

```

```

21     if k < len(T) and T[k] != None:
22         inorder(T,left(k))
23         print("Món", T[k][0], " giá:", T[k][1])
24         inorder(T,right(k))

```

menu_management.py

```

1     from BST_menu import * #Khai báo thư viện cây nhị phân tìm kiếm
2
3     #Khởi tạo cây tìm kiếm và đọc dữ liệu từ tệp
4     T = Tree()
5     f = open("menu.inp", "r", encoding="utf8")

```

```

6 lines = f.readlines()
7 for line in lines:
8     l = line.strip()
9     l = l.split(",", maxsplit=2)
10    Tree_Insert(T, l[1], l[0])
11 f.close()
12 inorder(T, 0) #Duyệt cây để in các món ăn xếp theo giá tăng dần

```

D. THÔNG TIN BỔ SUNG

Cây tìm kiếm nhị phân cho dữ liệu danh bạ được cài đặt bằng cấu trúc nút liên kết thay vì mảng tránh lãng phí bộ nhớ như sau:

```

1 class Node:
2     def init (self, key, fullName, phoneNumber, p = None, left = None, right = None):
3         self.key = key
4         self.fullName = fullName
5         self.phoneNumber = phoneNumber
6         self.p = p
7         self.left = left
8         self.right = right
9 class Tree:
10    def init (self, key = None, fullName = None, phoneNumber = None):
11        if key == None:
12            self.root = None
13        else:
14            self.root = Node(key, fullName, phoneNumber)
15    def search(x, key):
16        while x != None and x.key != key:
17            if key < x.key:
18                x = x.left
19            else:
20                x = x.right
21        return x
22    def Tree_Insert(T,z):
23        y = None
24        x = T.root
25        while x != None:
26            y = x
27            if z.key.upper() < x.key.upper():
28                x = x.left
29            else:
30                x = x.right
31            z.p = y
32        if y == None: #cây T rỗng
33            T.root = z
34        else:
35            if z.key.upper() < y.key.upper():
36                y.left = z
37            else:

```

```

38             y.right = z
39     def Tree_Insert_Update(T, key, fullName, phoneNumber):
40         n = search(T.root, key)
41         if n != None:
42             n.fullName = fullName
43             n.phoneNumber = phoneNumber
44             return 1
45         else:
46             newNode = Node(key, fullName, phoneNumber)
47             Tree_Insert(T, newNode)
48             return 0
49     def inorder(x):
50         if x != None:
51             inorder(x.left)
52             print(x.key, " ", x.fullName, " ", x.phoneNumber, sep="", end = "")
53             inorder(x.right)

```

CHUYÊN ĐỀ 3. TÌM HIỂU KỸ THUẬT DUYỆT ĐỒ THỊ VÀ ỨNG DỤNG

BÀI 11. KHÁI NIỆM ĐỒ THỊ

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Biết và trình bày được khái niệm đồ thị và các định nghĩa có liên quan.
- Mô tả được cấu trúc và ý nghĩa của ma trận kề và danh sách kề.

2. Năng lực

- Năng lực chung:
- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.
- Năng lực Tin học:
- Biết và trình bày được khái niệm đồ thị và các định nghĩa có liên quan.
- Mô tả được cấu trúc và ý nghĩa của ma trận kề và danh sách kề.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành các bài tập theo yêu cầu của GV thông qua hệ thống câu hỏi, phiếu học tập, thông qua sản phẩm.

II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, máy tính, máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước Bài 11. Khái niệm đồ thị.

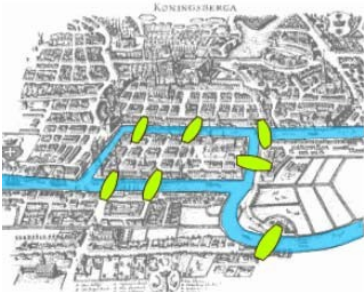
III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động 1: Khởi động

a) Mục tiêu: HS làm quen với một bài toán mới, qua đó gợi ý tò mò muốn biết về một mô hình dữ liệu mới. Tạo hứng thú học tập cho HS.

b) Nội dung: HS trả lời hai câu hỏi sau:

1. Năm 1736, nhà bác học Euler đưa ra bài toán, được gọi là bài toán 7 cây cầu ở Königsberg. Tại thành phố cổ Königsberg của nước Phổ cũ (nay thuộc nước Nga) có dòng sông Pregel vắt ngang qua, chia thành phố thành các vùng riêng biệt. Bài toán Euler đặt ra là làm sao đi qua tất cả 7 cây cầu này, mỗi cầu chỉ được phép đi qua đúng một lần. Em hãy giải bài toán trên. Em hãy quan sát Hình 11.1 SCD trang 49 và cho biết làm sao đi qua được tất cả 7 cây cầu này, mỗi cây cầu chỉ được phép đi qua đúng 1 lần? Có thể dùng mô hình dữ liệu nào để mô phỏng bài toán này?



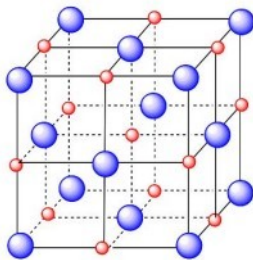
Hình 11.1 Sơ đồ các cây cầu trong thành phố cổ Königsberg

2. Em hãy nêu một số bài toán thực tế khác có thể biểu diễn được bằng đồ thị.

c) Sản phẩm: Câu trả lời của HS.

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm thảo luận và ghi ra giấy câu trả lời của nhóm GV gọi đại diện lần lượt 6 nhóm trả lời.

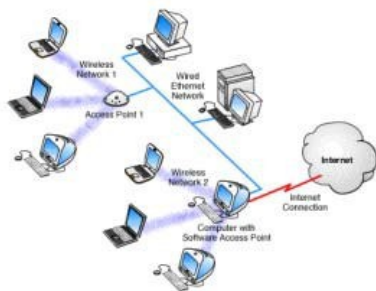
GV: Nhận xét mô hình đồ thị, để giải được bài toán trên chúng ta cùng tìm hiểu các khái niệm đồ thị trong bài học hôm nay. GV chiếu một số hình ảnh của các bài toán thực tế có thể biểu diễn được bằng đồ thị.



a) Cấu trúc tinh thể liên kết ion của muối ăn



b) Sơ đồ các tuyến xe buýt trong một thành phố



c) Mạng Internet kết nối máy tính toàn cầu



d) Mô hình liên kết siêu văn bản của các trang web trên Internet

2. Hoạt động 2: Hình thành kiến thức

Hoạt động 2.1: Tìm hiểu khái niệm đồ thị

- Mục tiêu: HS biết được khái niệm đồ thị và một số định nghĩa, khái niệm trực tiếp có liên quan đến đồ thị.
- Nội dung: HS hoàn thành phiếu học tập số 1.

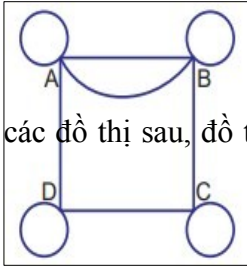
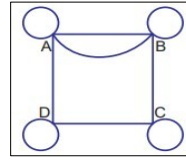
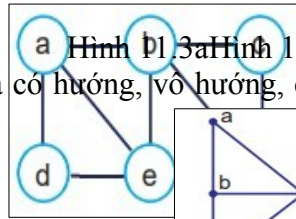
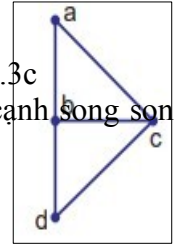
PHIẾU HỌC TẬP SỐ 1

1. Cho đồ thị $G = (V, E)$, trong đó V, E Giả sử có n đỉnh, m cạnh

-Tập các đỉnh:.....

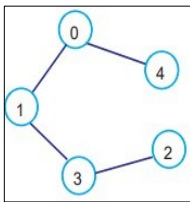
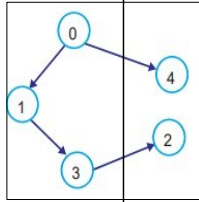
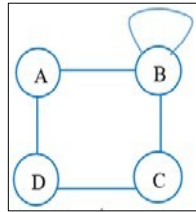
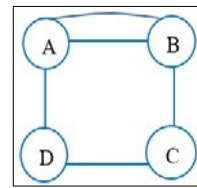
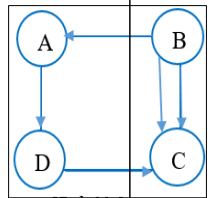
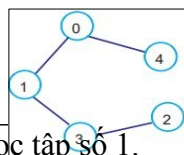
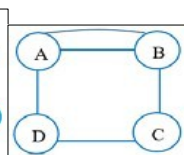
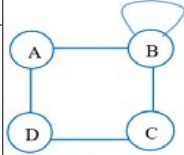
-Tập các cạnh:..... Trong đó các cạnh $e = (v_i, v_j)$

Hãy nêu các tập đỉnh và tập cạnh của của các đồ thị sau:



Trong các đồ thị sau, đồ thị nào là có hướng, vô hướng, có khuyên, cạnh song song, cạnh song song có hướng?

Hình 11.5 a Hình 11.5 b Hình 11.5 c Hình 11.5 d Hình 11.5 e
Thế nào là đơn đồ thị?



c) Sản phẩm: Hoàn thành phiếu học tập số 1.

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và hoàn thành phiếu học tập số 1.

Hoạt động của GV và HS

Bước 1: Chuyển giao nhiệm vụ

GV: Chiếu phiếu học tập số 1 lên bảng cho các nhóm quan sát.

Bước 2: Thực hiện nhiệm vụ

Sản phẩm dự kiến

1. Khái niệm đồ thị

a) Khái niệm đồ thị

Đồ thị $G = (V, E)$ là tập hợp hữu hạn các đỉnh V và tập hợp các cạnh E nối các đỉnh V với nhau.

HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt trả lời các câu hỏi trong phiếu học tập số 1.

GV: Quan sát và trợ giúp các nhóm HS.

Bước 3: Báo cáo, thảo luận

HS: Các nhóm HS đại diện trả lời.

Các nhóm còn lại nghe và nhận xét, bổ sung cho nhau.

GV: Điều khiển hoạt động của của các nhóm HS.

Bước 4: Kết luận, nhận định

GV nhận xét câu trả lời của các nhóm. Cùng cố kiến thức cần ghi nhớ.

HS: Ghi nhớ kiến thức.

Giả sử đồ thị $G = (V, E)$ có n đỉnh và m cạnh, ta có các kí hiệu sau:

- Tập hợp các đỉnh $V = \{v_1, v_2, v_3, \dots, v_n\}$;
- Tập hợp các cạnh $E = \{e_1, e_2, e_3, \dots, e_m\}$.

Trong đó, các cạnh $e_k = (v_i, v_j)$ nối hai đỉnh v_i và v_j .

Ví dụ 1: Xét đồ thị:

Tập đỉnh $V = \{A, B, C, D\}$.

Tập cạnh $E = \{(A, B), (A, D), (D, C), (B, C), (B, A)\}$.

Ví dụ 2: Xét đồ thị:

Tập đỉnh $V = \{a, b, c, d\}$.

Tập cạnh $E = \{(a, b), (a, c), (b, c), (b, d), (d, c)\}$.

b) Đồ thị vô hướng và đồ thị có hướng

Đồ thị vô hướng có các cạnh nối không phân biệt hướng, có thể đi được hai chiều, các cạnh được biểu diễn bằng các đoạn thẳng.

Ví dụ: Hình 11.5a, 11.5c, 11.5d là các đồ thị vô hướng:

Đồ thị có hướng có mũi tên chỉ hướng trên các cạnh, chỉ đi theo hướng có mũi tên:

c) Đơn đồ thị

Nếu đồ thị có cạnh $e = (v, v)$, tức là xuất phát và kết thúc tại một đỉnh, thì được gọi là có **khuyên**.

Nếu giữa hai đỉnh u, v có nhiều hơn một cạnh nối thì được gọi là có cạnh **song song**.

Bước 1: Chuyển giao nhiệm vụ

GV: Nêu câu hỏi củng cố kiến thức.

1. Cây, cây nhị phân và cây tìm kiếm nhị phân có là mô hình đồ thị không?

2. Vẽ đồ thị vô hướng $G = (V, E)$ sau: $V = [0, 1, 2, 3, 4]$ $E = [\{0, 1\}, \{0, 4\}, \{1, 2\}, \{1, 3\}, \{2, 4\}]$

3. Mô tả tập hợp đỉnh V và tập hợp cạnh E của đồ thị vô hướng trong Hình 11.7.

Bước 2: Thực hiện nhiệm vụ

HS: Trả lời các câu hỏi trên vào vở ghi.

GV: Quan sát HS làm.

Bước 3: Báo cáo, thảo luận

GV: Gọi 2 HS lên bảng làm câu 2, 3, một HS đứng tại vị trí trả lời câu 1.

Bước 4: Kết luận, nhận định

GV gọi HS nhận xét câu trả lời của các bạn.

GV: Đánh giá bài làm chấm điểm. Cùng cố kiến thức.

Đồ thị $G = (V, E)$ được gọi là **đơn đồ thị** nếu đồ thị **không có khuyên** và **không có cạnh song song**. Với đơn đồ thị, giữa hai đỉnh bất kì của đồ thị có nhiều nhất một cạnh nối.

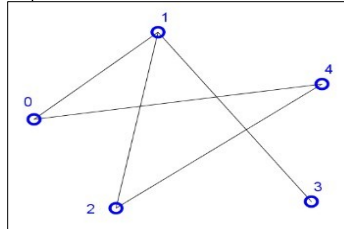
Củng cố kiến thức

1. Cây nhị phân và cây tìm kiếm nhị phân đúng là mô hình đồ thị.

2. Đồ thị này có thể vẽ như sau:

$V = [0, 1, 2, 3, 4]$;

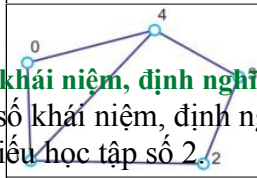
$E = [\{0, 1\}, \{0, 4\}, \{1, 2\}, \{1, 3\}, \{2, 4\}]$.



3.

$V = [0, 1, 2, 3, 4]$

$E = [\{0, 1\}, \{0, 4\}, \{1, 2\}, \{1, 4\}, \{2, 3\}, \{3, 4\}]$



Hoạt động 2.2: Tìm hiểu một số khái niệm, định nghĩa liên quan đến đồ thị

a) Mục tiêu: HS biết được một số khái niệm, định nghĩa quan trọng liên quan đến đồ thị

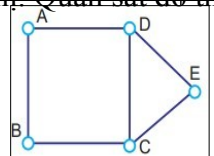
b) Nội dung: HS hoàn thành phiếu học tập số 2.

PHIẾU HỌC TẬP SỐ 2

Đọc, trao đổi và thảo luận các khái niệm, định nghĩa liên quan đến đồ thị. Quan sát đồ thị ở Hình 1.

Kể tên các đỉnh kề với D.

Hãy cho biết bậc của đỉnh A, C.

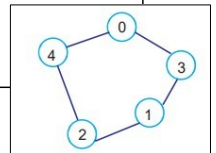


Liệt kê các đường đi từ đỉnh A đến đỉnh E.

Liệt kê 3 chu trình trong đồ thị.

Đồ thị có liên thông không? Có mấy thành phần liên thông?

Biểu diễn đồ thị dưới dạng ma trận kề, danh sách kề của đồ thị bên.



c) Sản phẩm: Hoàn thành phiếu học tập số 2.

1. Kề với đỉnh D là các đỉnh: A, C, E. 2. $\deg(A) = 2$; $\deg(C) = 3$.

3. Đường đi từ $A \rightarrow E$:

$\{(A, D), (D, E)\}$; $\{(A, B), (B, C), (C, E)\}$; $\{(A, B), (B, C), (C, D), (D, E)\}$.

4. $A \rightarrow B \rightarrow C \rightarrow D \rightarrow A$; $D \rightarrow E \rightarrow C \rightarrow D$; $A \rightarrow D \rightarrow E \rightarrow C \rightarrow B \rightarrow A$.

5. Đồ thị liên thông, có 1 thành phần liên thông.

6. Biểu diễn ma trận kề, danh sách kề:

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 \rightarrow 3, 4 \\ 1 \rightarrow 2, 3 \\ 2 \rightarrow 1, 4 \\ 3 \rightarrow 0, 1 \\ 4 \rightarrow 0, 2 \end{pmatrix}$$

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và hoàn thành phiếu

học tập số 2.

Hoạt động của GV và HS	Sản phẩm dự kiến
<p>Bước 1: Chuyển giao nhiệm vụ GV: Chiếu phiếu học tập số 2 lên bảng cho các nhóm quan sát trả lời câu hỏi.</p> <p>Bước 2: Thực hiện nhiệm vụ HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt trả lời các câu hỏi trong phiếu học tập số 2. GV: Quan sát và trợ giúp các nhóm HS.</p> <p>Bước 3: Báo cáo, thảo luận HS: Các nhóm HS đại diện trả lời. Các nhóm còn lại nghe và nhận xét, bổ sung cho nhau. GV: Điều khiển hoạt động của các nhóm HS.</p> <p>Bước 4: Kết luận, nhận định</p>	<p>2. Một số khái niệm liên quan đến đồ thị Cho đơn đồ thị $G = (V, E)$, có thể vô hướng hoặc có hướng.</p> <p>Nếu từ đỉnh u có cạnh nối đến đỉnh v thì v là đỉnh kề của u.</p> <p>Nếu G là vô hướng thì nếu v là đỉnh kề của u thì u cũng là đỉnh kề của v.</p> <p>Nếu cạnh e từ u đến v thì chúng ta sẽ kí hiệu $e: u \rightarrow v$, hay $u \rightarrow v$.</p> <p>Bậc của đỉnh u, kí hiệu $\deg(u)$, là số lượng các đỉnh kề với u.</p> <p>Nếu G là đồ thị có hướng chúng ta sẽ có các định nghĩa sau:</p> <ul style="list-style-type: none"> - Bậc ra của đỉnh u, kí hiệu $\deg^+(u)$ là số lượng các đỉnh kề với u. - Bậc vào của đỉnh u, kí hiệu $\deg^-(u)$, là số các đỉnh v có cạnh nối đến u.

GV nhận xét câu trả lời của các nhóm. Cùng cố kiến thức cần ghi nhớ.

HS: Ghi nhớ kiến thức.

Bước 1: Chuyển giao nhiệm vụ

GV: Nêu câu hỏi củng cố kiến thức

1. Khi nào một đỉnh của đồ thị có bậc bằng 0?
2. Xác định ma trận kề và danh sách kề của các đồ thị ở Hình 11.11.

Bước 2: Thực hiện nhiệm vụ

-

Đường đi từ đỉnh s đến đỉnh t là dãy các cạnh kề nhau $e_1, e_2, \dots, e_k$ nối từ đỉnh s đến đỉnh t và thoả mãn điều kiện: tồn tại dãy các đỉnh kề nhau và khác nhau từng đôi một $v_0, v_1, v_2, \dots, v_k$, sao cho e_i là cạnh nối đỉnh v_{i-1} đến v_i ($i = 1, 2, \dots, k$), $v_0 = s$, và $v_k = t$. Vì các đỉnh $v_0, v_1, \dots, v_k$ khác nhau từng đôi một, do đó các cạnh $e_1, e_2, \dots, e_k$ cũng khác nhau từng đôi một. Trong trường hợp đỉnh v_0 trùng với đỉnh v_k thì dãy các cạnh kề nhau $e_1, e_2, \dots, e_k$ ở trên là chu trình.

- Đồ thị $G = (V, E)$ được gọi là liên thông nếu với mọi đỉnh $u, v \in V$ thì tồn tại đường đi từ u đến v . Nếu G không là liên thông thì tập hợp V có thể được chia thành các tập hợp con rời nhau mà mỗi đồ thị con xác định bởi các tập hợp này là liên thông. Người ta gọi các tập hợp con đó là các thành phần liên thông của

G . Theo quy ước, nếu đồ thị G chỉ có một đỉnh thì đồ thị này là liên thông.

- Cho đồ thị $G = (V, E)$, V có n đỉnh. Các đỉnh V được kí hiệu $0, 1, 2, \dots, n-1$. Khi đó ma trận kề của đồ thị G là ma trận có kích thước $n \times n$, được định nghĩa như sau:

- Nếu G là đồ thị vô hướng thì ma trận kề là đối xứng, tức là $a_{ij} = a_{ji}$ với mọi i, j .

- Với mỗi $u \in V$, danh sách các đỉnh kề của u là $\text{Adj}(u) = \{v \mid (u, v) \in E\}$. Danh sách kề của đồ thị G , kí hiệu là Adj , là tập hợp danh sách đỉnh kề của các đỉnh của G .

Củng cố kiến thức

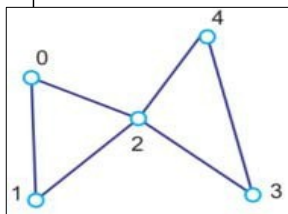
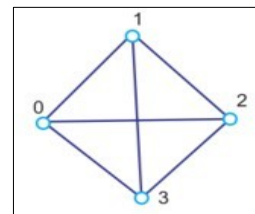
1. Một đỉnh của đồ thị có bậc 0 nếu không có cạnh đi vào hoặc đi ra từ đỉnh này.

2. a) b)

HS: Trả lời các câu hỏi trên vào vở ghi.

GV: Quan sát HS làm.

$$A = [a_{ij}], a_{ij} = \begin{cases} 1 & \text{nếu } (i, j) \in E \\ 0 & \text{nếu } (i, j) \notin E \end{cases}$$



Ma trận kê và danh sách
kê:

Ma trận kề và danh sách
kề:

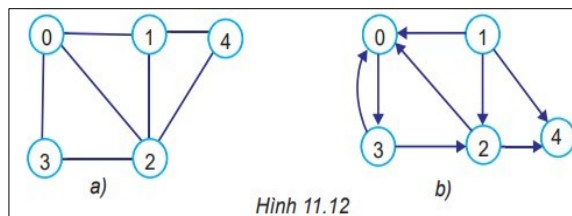
Bước 3: Báo cáo, thảo luận GV: gọi 2 HS lên bảng làm câu 2 phần a, b, một HS đứng tại vị trí trả lời câu 1. Bước 4: Kết luận, nhận định GV gọi HS nhận xét câu trả lời của các bạn. GV: Đánh giá bài làm chấm điểm. Cùng cố kiến thức.	$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ A: & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$ $\begin{pmatrix} 0 \rightarrow 1, 2 \\ 1 \rightarrow 0, 2 \\ Adj: 2 \rightarrow 0, 1, 3, 4 \\ 3 \rightarrow 2, 4 \\ 4 \rightarrow 2, 3 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ A: & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 \rightarrow 1, 2, 3 \end{pmatrix}$ $\begin{pmatrix} 1 \rightarrow 0, 2, 3 \\ Adj: 0, 1, 3 \\ 3 \rightarrow 0, 1, 2 \end{pmatrix}$
--	--	---

Hoạt động 2.3: Tìm hiểu biểu diễn đồ thị

- a) Mục tiêu: HS biết được cách tổ chức và biểu diễn các mô hình đồ thị trên máy tính.
b) Nội dung: HS hoàn thành phiếu học tập số 3.

PHIẾU HỌC TẬP SỐ 3

Tìm hiểu một số cách biểu diễn dữ liệu đồ thị trên máy tính. Thảo luận xem cách nào là hợp lý



Hình 11.12

- c) Sản phẩm: Hoàn thành phiếu học tập số 3.
d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và hoàn thành phiếu học tập số 3.

Hoạt động của GV và HS	Sản phẩm dự kiến
Bước 1: Chuyển giao nhiệm vụ GV: Chiếu phiếu học tập số 3 lên bảng cho các nhóm quan sát làm bài. Bước 2: Thực hiện nhiệm vụ HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt trả lời các câu hỏi trong phiếu học tập số 3. GV: Quan sát và trợ giúp các nhóm HS. Bước 3: Báo cáo, thảo luận	3. Biểu diễn đồ thị Để thiết lập và biểu diễn dữ liệu đồ thị trên máy tính, thực hiện như sau: – Tập các đỉnh V được đánh số từ 0 đến $n - 1$: $V = \{0, 1, 2, \dots, n - 1\}$ – Tập các cạnh E là dãy (danh sách) các cạnh, mỗi cạnh sẽ là một cặp hai chỉ số tương ứng với hai đỉnh. Mỗi cạnh của đồ thị vô hướng E được biểu diễn như một tập hợp, ví dụ $e = \{i, j\}$. Mỗi cạnh của đồ thị có hướng là cặp hai chỉ số có thứ tự, ví dụ kiểu tuple là $e = (i, j)$ hoặc list là $e = [i, j]$.

HS: Các nhóm HS đại diện lên bảng viết ma trận kề, danh sách kề của các đồ thị.

Các HS còn lại quan sát, nhận xét.

GV: Điều khiển hoạt động của các nhóm HS.

Bước 4: Kết luận, nhận định

GV nhận xét bài làm của các nhóm. Cùng cố kiến thức cần ghi nhớ.

HS: Ghi nhớ kiến thức.

Bước 1: Chuyển giao nhiệm vụ

GV: Nêu câu hỏi củng cố kiến thức

Ví dụ: Với hai đồ thị trong Hình 11.12, E được định nghĩa như sau:

a) $E = [\{0, 1\}, \{0, 2\}, \{0, 3\}, \{1, 2\}, \{1, 4\}, \{2, 4\}]$

b) $E = [(0, 3), (1, 0), (1, 2), (1, 4), (2, 0), (2, 4), (3, 0), (3, 2)]$

– Ma trận kề A của đồ thị là mảng hai chiều kích thước $n \times n$ được định nghĩa như sau:
 $A[i][j] = 1$ khi và chỉ khi tồn tại cạnh nối từ đỉnh v_i đến v_j .

Ví dụ: Ma trận kề của các đồ thị trong Hình 11.12:

– Danh sách kề Adj của đồ thị được định nghĩa như sau: $A[i]$ là danh sách các đỉnh kề với đỉnh v_i . Với định nghĩa này, ta có các bảng biểu diễn Adj của các đồ thị trong Hình 11.12:

Biểu diễn danh sách kề của đồ thị Hình 11.12 bằng kiểu list trong Python.

a) $adj = [[1, 2, 3], [0, 2, 4], [0, 1, 3, 4], [0, 2], [1, 2]]$

b) $adj = [[3], [2, 4], [0, 4], [0, 2], []]$

Củng cố kiến thức

Đồ thị a: $N = 4$, có $V = \{a, b, c, d\}$, $E = \{(a, b), (a, c), (a, d), (b, c), (c, d), (b, d)\}$.

Ma trận kề và danh sách kề của đồ thị a:

Thiết lập bộ dữ liệu biểu diễn gồm (n, V, E, A, Adj) cho các đồ thị sau:

$$\begin{pmatrix} 0111 \\ 1011 \\ 1101 \end{pmatrix} \quad |$$

$$\left\{ \begin{array}{l} a \rightarrow b, c, d \\ b \rightarrow a, c, d \\ \text{Adj} \uparrow \\ c \rightarrow a, b, d \end{array} \right\} \quad |$$

$\downarrow$
 $d \rightarrow$

|
(') (

$a\ b\ c\)$

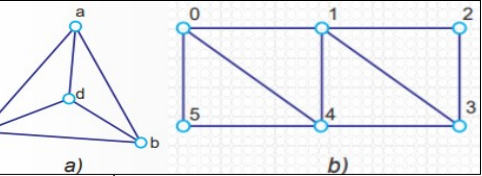
Bước 2: Thực hiện nhiệm vụ

$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$
---	---

Đỉnh	Danh sách kề	Đỉnh	Danh sách kề
0	1 2 3	0	3
1	0 2 4	1	0 2 4
2	0 1 3 4	2	0 4
3	0 2	3	0 2
4	1 2	4	<rỗng>

a) Danh sách kề của đồ thị trong Hình 11.10a

b) Danh sách kề của đồ thị trong Hình 11.10b



Đồ thị b: $N = 6$,
 $V = \{0, 1, 2, 3, 4, 5\}$,

HS: Gọi 4 HS lên bảng làm bài, mỗi HS làm một (vẽ và (0,1), (0,5), (1,2), (2,3), (2,4), (3,4), (4,5)).

Bước 3: Báo cáo, thảo luận

GV: Quan sát HS làm. Ma trận kề và danh sách kề của đồ thị b

Bước 4: Kết luận, nhận định

GV nhận xét đánh giá bài làm chấm điểm. Cùng cố kiến thức.

$(010011) \setminus 0 \rightarrow 1, 4, 5$
 $\begin{matrix} | & || & | \\ 101110 & || & 1 \rightarrow 0, 2, 3, 4 \\ | & 010100 & || & 2 \rightarrow 1, 3 \\ A = & | & | & Adj = & | & | \\ | & 011010 & || & 3 \rightarrow 1, 2, 4 \\ | & 110101 & || & 4 \rightarrow 0, 1, 3, 5 \\ & || & || & || & || \\ & \setminus 100010 & \wedge & 5 \rightarrow 0, 4 \end{matrix}$

3. Hoạt động luyện tập

a) Mục tiêu: HS tìm được bậc của các đỉnh, biểu diễn được đồ thị dưới dạng ma trận kề, danh sách kề.

b) Nội dung: Làm bài tập 1, 2 phần luyện tập SCD trang 55.

1. Đồ thị ứng với mô hình bài toán 7 cây cầu ở Königsberg có phải là đơn đồ thị không? Tính bậc của các đỉnh của đồ thị đó.

2. Cho đồ thị G vô hướng với ma trận kề như hình bên. Hãy vẽ đồ thị trên.

c) Sản phẩm: Bài làm của HS trong vở ghi.

d) Tổ chức thực hiện:

GV gọi 2 HS lên bảng làm bài, các HS còn lại làm bài vào vở ghi. Gọi một số HS nhận xét bài trên bảng.

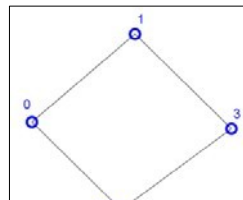
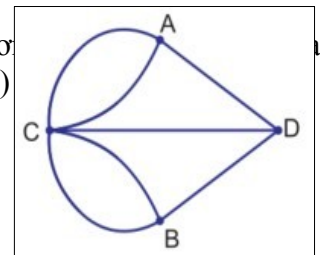
GV: Nhận xét, cho điểm những HS làm bài tốt.

Sản phẩm dự kiến:

1. Đồ thị tương ứng bài toán 7 cây cầu không phải là đồ thị đơn đồ thị này như sau: $\deg(A) = 3, \deg(B) = 3, \deg(C) = 5, \deg(D)$

2. Đồ thị có dạng như hình sau:

$\begin{pmatrix} 0110 \\ 1001 \\ 1001 \\ 0110 \end{pmatrix}$
 $A = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$



4. Hoạt động vận dụng

a) Mục tiêu: HS vận dụng kiến thức đã học về đồ thị biết dựng, biểu diễn đồ thị.

b) Nội dung: Làm bài tập 1, 2 SCD trang 55 phần vận dụng.

c) Sản phẩm: Bài làm của HS trong vở ghi.

d) Tổ chức thực hiện: HS làm ở nhà, tiết sau GV gọi 2 HS lên bảng chữa bài.

Sản phẩm dự kiến:

1. Các ma trận kề của đồ thị đầy đủ tương ứng $n = 2, 3, 4$.

(01)

$$\begin{matrix} A \\ (10) \end{matrix} \Big| \quad |$$

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

```

(0111)
1011
A 1101
1110

```

2. Đáp số: 3 thành phần liên thông.

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phần vận dụng.
- Đọc trước Bài 12. Biểu diễn đồ thị.

BÀI 12. BIỂU DIỄN ĐỒ THỊ

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Thiết lập và biểu diễn được đồ thị bằng ma trận kề và danh sách kề.

2. Năng lực

- Năng lực chung:
- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.
- Năng lực Tin học:
- Biểu diễn được đồ thị bằng ma trận kề và danh sách kề.
- Cài đặt được chương trình bằng ngôn ngữ Python.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành các bài tập theo yêu cầu của GV thông qua hệ thống câu hỏi, phiếu học tập, thông qua sản phẩm.

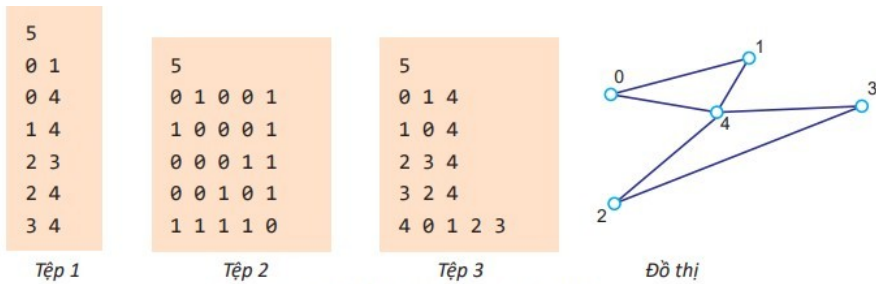
II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, máy tính, máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước Bài 12. Biểu diễn đồ thị.

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động 1: Khởi động

- Mục tiêu: HS được làm quen và nhận biết một số khuôn dạng dữ liệu đầu vào của đồ thị dưới dạng các tệp dữ liệu.
- Nội dung: Quan sát đồ thị Hình 12.1 và cho biết mỗi tệp dữ liệu sau có ý nghĩa gì?



Hình 12.1. Đồ thị và dữ liệu mô tả đồ thị

c) Sản phẩm: Câu trả lời của HS.

d) Tổ chức thực hiện: GV cho HS quan sát Hình 12.1 SCD trang 56 và trả lời. GV: Nhận xét câu trả lời của HS.

2. Hoạt động 2: Hình thành kiến thức

Hoạt động 2.1: Tìm hiểu các mô hình dữ liệu đồ thị (vô hướng hoặc có hướng)

a) Mục tiêu: HS hiểu được 3 khuôn dạng tệp dữ liệu đầu vào thường gặp nhất của đồ thị.

b) Nội dung: HS hoàn thành phiếu học tập số 1.

PHIẾU HỌC TẬP SỐ 1

Hãy điền vào chỗ để trống trong các câu 1, 2, 3

Hãy cho biết Tập 1 trong Hình 12.1 biểu diễn đồ thị dưới dạng

Trong đó dòng đầu tiên n là.....

n dòng sau, mỗi dòng cho biết.....

Hãy cho biết Tập 2 trong Hình 12.1 biểu diễn đồ thị dưới dạng

Trong đó dòng đầu tiên n là.....

n dòng sau, mỗi dòng cho biết.....

Hãy cho biết Tập 3 trong Hình 12.1 biểu diễn đồ thị dưới dạng

Trong đó dòng đầu tiên n là.....

n dòng sau, mỗi dòng cho biết.....

Vẽ đồ thị có tệp dữ liệu ma trận kề Hình 12.5. 2 (bên cạnh).

Có thể có hai tệp dữ liệu dạng danh sách kề khác nhau nhưng biểu diễn hai đồ thị hoàn toàn gi

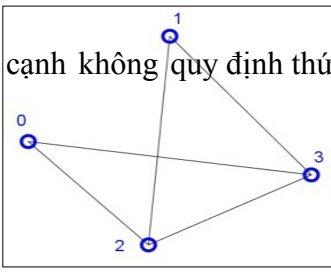
4
0 0 1 1
0 0 1 1
1 1 0 1
1 1 1 0

c) Sản phẩm: Hoàn thành phiếu học tập số 1.

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và hoàn thành phiếu học tập số 1.

Hoạt động của GV và HS	Sản phẩm dự kiến
------------------------	------------------

<p>Bước 1: Chuyển giao nhiệm vụ GV: Chiếu phiếu học tập số 1. lên bảng cho các nhóm quan sát và hoàn thành.</p> <p>Bước 2: Thực hiện nhiệm vụ HS: Thảo luận theo nhóm, nghiên cứu SCĐ lần lượt hoàn thành phiếu học tập số 1. GV: Quan sát và trợ giúp các nhóm HS.</p> <p>Bước 3: Báo cáo, thảo luận HS: Các nhóm HS đại diện trả lời. Các nhóm còn lại nghe và nhận xét, bổ sung cho nhau. GV: Điều khiển hoạt động của của các nhóm HS.</p> <p>Bước 4: Kết luận, nhận định GV nhận xét câu trả lời của các nhóm. Cùng cố kiến thức cần ghi nhớ. HS: Ghi nhớ kiến thức.</p>	<p>1. Mô hình dữ liệu đồ thị Có nhiều cách biểu diễn dữ liệu của một đồ thị, trong đó thường gặp ba cách sau:</p> <ul style="list-style-type: none"> – Sử dụng danh sách các cạnh của đồ thị. – Sử dụng ma trận kề kích thước $n \times n$. – Sử dụng danh sách kề. Trong n là số đỉnh của đồ thị. <p>a) Dữ liệu danh sách các cạnh của đồ thị Tập dữ liệu loại này có dạng:</p> <ul style="list-style-type: none"> – Dòng đầu tiên là số n. – Các dòng tiếp theo, mỗi dòng là hai chỉ số mô tả một cạnh của đồ thị: <pre> n i₁ j₁ i₂ j₂ i_m j_m </pre> <p>b) Dữ liệu ma trận kề của đồ thị Tập dữ liệu loại này có dạng:</p> <ul style="list-style-type: none"> – Dòng đầu tiên là số n. – n dòng tiếp theo là dữ liệu ma trận kề A: <pre> n a₁₁ a₁₂ a_{1n} a₂₁ a₂₂ a_{2n} a_{n1} a_{n2} a_{nn} </pre> <p>c) Dữ liệu danh sách kề của đồ thị Tập dữ liệu loại này có dạng:</p> <ul style="list-style-type: none"> – Dòng đầu tiên là số n. – n dòng tiếp theo là dữ liệu danh sách kề Adj, cụ thể như sau: Dòng thứ i sẽ bắt đầu bằng số i, sau đó danh sách các đỉnh là kề của i, mỗi đỉnh ghi số thứ tự của đỉnh, cách nhau bởi dấu cách. <pre> n 0 s₀ t₀ ... u₀ 1 s₁ t₁ ... u₁ n-1 s_{n-1} t_{n-1} ... u_{n-1} </pre>
---	---

	<p>Củng cố kiến thức Đồ thị có dạng như hình bên. Có. Vì tập dữ liệu danh sách các cạnh không quy định thứ tự</p> 
--	--

Hoạt động 2.2: Tìm hiểu cách thiết lập đồ thị từ ma trận kề và danh sách kề

- a) Mục tiêu: HS biết và hiểu được cách thiết lập bộ dữ liệu chính của đồ thị từ tập dữ liệu ma trận kề và danh sách kề.
b) Nội dung: HS hoàn thành phiếu học tập số 2.

PHIẾU HỌC TẬP SỐ 2

Viết chương trình đọc dữ liệu từ tệp biểu diễn đồ thị là ma trận kề gồm có dòng đầu tiên chứa số n là số đỉnh của đồ thị.
Viết chương trình đọc dữ liệu từ tệp biểu diễn đồ thị là danh sách kề gồm có dòng đầu tiên chứa số n là số đỉnh của đồ thị.
Khẳng định dãy Adj[i] có số lượng phần tử bằng số các phần tử có giá trị 1 của hàng thứ i của ma trận kề A.
Khi nào ma trận kề A chỉ gồm toàn số 0.

- c) Sản phẩm: Hoàn thành phiếu học tập số 2.

1. Thiết lập đồ thị từ tập ma trận kề

```
def builgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)] A=[]
for line in f:
    A.append([int(x) for x in line.split()]) f.close()
return V,A
fi="data1.inp" V,A=builgraph(fi)
print(V)# in ra danh sách đỉnh
for i in range(len(V)):
    print(A[i]) # in ra ma trận kề
```

2. Thiết lập đồ thị từ tập danh sách kề

```
def builgraph(fname): f=open(fname) n=int(f.readline())
V=[i for i in range(n)] Adj=[[] for i in range(n)]
for i in range(n):
    line=[int(x) for x in f.readline().split()] line=line[1:] # lấy dãy các số từ vị trí thứ 2
    Adj[i].extend(line) # bổ sung vào hàng thứ i của Adj
f.close() return V,Adj
fi="data2.inp" V,Adj=builgraph(fi)
print(V) # in ra danh sách đỉnh
for i in range(len(V)):
    print(Adj[i]) # in ra danh sách kề
```

print(i,Adj[i])

in ra danh sách các đỉnh kề với đỉnh i

Củng cố kiến thức

3. Khẳng định đúng: Adj[i] bằng số lượng các phần tử 1 của hàng thứ i của ma trận kề A.
4. Ma trận kề A gồm toàn số 0 nếu đồ thị không có bất kì cạnh nào, tức là số cạnh bằng 0.
- d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và hoàn thành phiếu học tập số 2

Hoạt động của GV và HS

Bước 1: Chuyển giao nhiệm vụ

GV: Chiếu phiếu học tập số 2 lên bảng cho các nhóm quan sát và hoàn thành.

Bước 2: Thực hiện nhiệm vụ

HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt hoàn thành 2 câu hỏi trong phiếu học tập.

GV: Quan sát và trợ giúp các nhóm HS.

Bước 3: Báo cáo, thảo luận

HS: Các nhóm HS đại diện lên bảng viết chương trình. Các HS còn lại nghe và nhận xét, bổ sung.

GV: Điều khiển hoạt động của các nhóm HS.

Bước 4: Kết luận, nhận định

GV nhận xét bài làm của các nhóm. Củng cố kiến thức cần ghi nhớ.

HS: Ghi nhớ hai hàm xây dựng đồ thị biểu diễn bằng ma trận kề, danh sách kề.

Hoạt động 2.3: Tìm hiểu thiết lập đồ thị từ tệp danh sách cạnh

- a) Mục tiêu: HS biết được cách thiết lập bộ dữ liệu của đồ thị từ tệp dữ liệu danh sách các cạnh.
- b) Nội dung: HS hoàn thành phiếu học tập số 3.

PHIẾU HỌC TẬP SỐ 3

Cho tệp data3.inp biểu diễn của đồ thị dạng danh sách cạnh. Dòng đầu tiên chứa n là số đỉnh của đồ thị.

Các dòng tiếp theo mỗi dòng có hai số i, j mô tả cạnh nối đỉnh i với đỉnh j. Cập nhật vào ma trận kề và danh sách kề xét với hai trường hợp: Đồ thị vô hướng, đồ thị có hướng.

Yêu cầu viết chương trình xây dựng 4 hàm dựng đồ thị Buildgraph với các trường hợp:

- a) Đồ thị vô hướng chuyển từ danh sách cạnh sang ma trận kề.
- b) Đồ thị vô hướng chuyển từ danh sách cạnh sang danh sách kề.
- c) Đồ thị có hướng chuyển từ danh sách cạnh sang ma trận kề.
- d) Đồ thị có hướng chuyển từ danh sách cạnh sang danh sách kề.
- c) Sản phẩm: Hoàn thành phiếu học tập số 3.

Đồ thị vô hướng

- Thiết lập đồ thị biểu diễn bởi ma trận kề từ tệp dữ liệu danh sách các cạnh, áp dụng cho đồ thị vô hướng.

```
# Đồ thị vô hướng chuyển từ danh sách cạnh sang ma trận kề
def buildgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)]
A=[[0 for i in range(n)] for j in range(n)] # khai tap A
for line in f:
    edge=[int(x) for x in line.split()] i,j=edge[0],edge[1]
    A[i][j]=1
    A[j][i]=1
f.close() return V,A
fi="data3.inp" V,A=buildgraph(fi) print(V)
for i in range(len(V)): print(A[i])
```

- Thiết lập đồ thị biểu diễn bởi **danh sách kề từ tệp dữ liệu danh sách các cạnh**, áp dụng cho đồ thị vô hướng:

```
# Đồ thị vô hướng chuyển từ danh sách cạnh sang danh sách kề
def buildgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)]
Adj=[] for i in range(n) # khai tạo Adj
for line in f:
    edge=[int(x) for x in line.split()] i,j=edge[0],edge[1]
    Adj[i].append(j) # bổ sung j vào hàng thứ i của Adj Adj[j].append(i)
    # bổ sung i vào hàng thứ j của Adj
f.close() return V,Adj
fi="data3.inp"
```

```
V,Adj=buildgraph(fi) print(V)
for i in range(len(V)): print(i,Adj[i])
```

Đồ thị có hướng

- Thiết lập đồ thị biểu diễn bởi **ma trận kề từ tệp dữ liệu danh sách các cạnh**, áp dụng cho đồ thị có hướng.

```
# Đồ thị có hướng chuyển từ danh sách cạnh sang ma trận kề
def buildgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)]
A=[[0 for i in range(n)] for j in range(n)]
for line in f: i,j=map(int,line.split())
    A[i][j]=1
f.close() return V,A
fi="data3.inp" V,A=buildgraph(fi) print(V)
for i in range(len(V)): print(A[i])
```

- Thiết lập đồ thị biểu diễn bởi **danh sách kề từ tệp dữ liệu danh sách các cạnh**, áp dụng cho đồ thị có hướng.

```
def buildgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)]
Adj=[] for i in range(n)
for line in f: i,j=map(int,line.split()) Adj[i].append(j)
f.close() return V,Adj
```

```
fi="data3.inp" V,Adj=buildgraph(fi) print(V)
for i in range(len(V)): print(i,Adj[i])
```

Củng cố kiến thức

1. Một đơn đồ thị, vô hướng có n đỉnh, có thể có số cạnh lớn nhất là bao nhiêu?

Đồ thị đơn, vô hướng, n đỉnh thì số cạnh lớn nhất là tổ hợp chập 2 của n , tức là $\frac{n(n-1)}{2}$

2

2. Khi nào thì tất cả các phần tử của Adj đều rỗng?

Nếu đồ thị không có bất kì cạnh nào thì tất cả các phần tử của Adj đều rỗng.

d) Tổ chức thực hiện: Chia lớp thành 4 nhóm, các nhóm thảo luận và hoàn thành phiếu học tập số 3.

Hoạt động của GV và HS Bước 1: Chuyển giao nhiệm vụ

GV: Chiếu phiếu học tập số 3 lên bảng cho các nhóm quan sát làm bài.

Bước 2: Thực hiện nhiệm vụ

HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt viết các chương trình theo phiếu học tập số 3.

GV: Quan sát và trợ giúp các nhóm HS.

Bước 3: Báo cáo, thảo luận

HS: Các nhóm HS đại diện lên bảng viết chương trình hoặc chiếu bài đã làm trên máy tính Các HS còn lại quan sát, nhận xét.

GV: Điều khiển hoạt động của các nhóm HS. Hướng dẫn tạo các tệp input và chạy chương trình.

Bước 4: Kết luận, nhận định

GV nhận xét bài làm của các nhóm, cho điểm cộng với các nhóm làm tốt. Củng cố kiến thức cần ghi nhớ.

HS: Ghi nhớ 4 kiến thức chương trình xây dựng với đồ thị có hướng, vô hướng.

3. Hoạt động luyện tập

a) Mục tiêu: HS tìm được bậc của các đỉnh, biểu diễn được đồ thị dưới dạng ma trận kề, danh sách kề.

b) Nội dung: Làm bài tập 1, 2 phần luyện tập SCD trang 61.

1. Bổ sung thêm đoạn chương trình kiểm tra khi đọc dữ liệu danh sách các cạnh đồ thị của Hoạt động 3 như sau: Với mỗi dòng dữ liệu, nếu hai chỉ số $i = j$ thì bỏ qua dòng này.

2. Từ ma trận kề A của đồ thị G có thể tính được số các cạnh của đồ thị không? Nếu được thì tính bằng cách nào?

c) Sản phẩm: Bài làm của HS trong vở ghi, trên máy tính, điện thoại.

1.

Đồ thị vô hướng chuyển từ danh sách cạnh sang ma trận kề, nếu $i=j$ thì bỏ qua

```
def buildgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)]
```

```
A=[[0 for i in range(n)] for j in range(n)] # khởi tạo A
```

```
for line in f:
```

```
    edge=[int(x) for x in line.split()] i,j=edge[0],edge[1]
```

```
    if i!=j: A[i][j]=1; A[j][i]=1
```

f.close()

2. Từ ma trận kề A của đồ thị G có thể tính được số các cạnh của đồ thị:

– Nếu đồ thị là vô hướng thì số cạnh của đồ thị bằng tổng số các phần tử bằng 1 của ma trận A nằm phía trên đường chéo chính, hay bằng tổng các số 1 chia cho 2.

– Nếu đồ thị là có hướng thì số cạnh của đồ thị bằng tổng số các phần tử bằng 1 của ma trận.

d) Tổ chức thực hiện: Giao về nhà làm bài, tiết sau gọi lên bảng chữa bài.

GV gọi 2 HS lên bảng cùng làm bài 1, 1 HS làm bài 2 các HS còn lại mở vở GV kiểm tra bài làm của HS trong vở ghi.

Gọi một số HS nhận xét bài trên bảng.

GV: Nhận xét, cho điểm những HS làm bài tốt, chạy chương trình cho cả lớp quan sát kết quả.

4. Hoạt động vận dụng

a) Mục tiêu: HS vận dụng kiến thức đã học về đồ thị viết được các hàm dựng đồ thị, biểu diễn đồ thị.

b) Nội dung: Làm bài tập 1, 2 SCD trang 61 phần vận dụng.

1. Cho ma trận kề A của đồ thị vô hướng G. Viết hàm GraphEdge(A) trả lại danh sách E các cạnh của đồ thị G.

2. Cho danh sách kề Adj của đồ thị G. Viết hàm GraphEdge(Adj) trả lại danh sách E các cạnh của đồ thị G. Viết chương trình cho hai trường hợp riêng biệt, G là đồ thị vô hướng và G là đồ thị có hướng.

c) Sản phẩm: Bài làm của HS trong vở ghi.

d) Tổ chức thực hiện: HS làm ở nhà, tiết sau GV gọi 2 HS lên bảng chữa bài. GV gọi 3 HS lên bảng làm bài:

- 1 HS viết hàm GraphEdge(A) trả lại danh sách E các cạnh của đồ thị G vô hướng biết dữ liệu đọc vào từ ma trận kề A.

- 1 HS viết hàm GraphEdge(Adj) trả lại danh sách E các cạnh của đồ thị G vô hướng biết dữ liệu đọc vào từ danh sách kề Adj.

- 1 HS viết hàm GraphEdge(Adj) trả lại danh sách E các cạnh của đồ thị G có hướng biết dữ liệu đọc vào từ danh sách kề Adj.

Các HS còn lại mở vở, hoặc máy tính GV kiểm tra bài làm của HS đã giao về nhà từ tiết trước. Gọi một số HS nhận xét bài trên bảng.

GV: Nhận xét, cho điểm những HS làm bài tốt, chạy chương trình cho cả lớp quan sát kết quả.

1. Hàm graphEdge(A) của đồ thị vô hướng có thể được thiết lập như sau:

```
def graphEdge(A): n=len(A) E=[]
for i in range(n):
    for j in range(i+1,n):
        if A[i][j]==1:
            E.append({i,j})
return E
```

2. Xét hai trường hợp khác nhau, đồ thị vô hướng và có hướng.

a) Đồ thị vô hướng, hàm GraphEdge(Adj) được thiết lập như sau: def GraphEdge(Adj):

```
n = len(Adj) E = []
for i in range(n):
    for j in Adj[i]:
        if (i,j) not in E: E.append((i,j))
return E
```

b) Đồ thị có hướng, hàm GraphEdge(Adj) được thiết lập như sau:

```
def GraphEdge(Adj):
    n = len(Adj) E = []
    for i in range(n):
        for j in Adj[i]: E.append((i,j))
    return E
```

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phân vận dụng.
- Đọc trước Bài 13. Thực hành thiết lập đồ thị.

BÀI 13. THỰC HÀNH THIẾT LẬP ĐỒ THỊ

Thời gian thực hiện: 2 tiết thực hành

I. MỤC TIÊU

1. Kiến thức

- Biểu diễn được đồ thị bằng ma trận kề và danh sách kề.

2. Năng lực

- Năng lực chung:
- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.
- Năng lực Tin học:
- Thực hành viết chương trình thiết lập được đồ thị theo các bộ dữ liệu khác nhau.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành các bài tập theo yêu cầu của GV thông qua hệ thống câu hỏi, nhiệm vụ, thông qua sản phẩm.

II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, máy tính, máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước Bài 13. Thực hành thiết lập đồ thị.

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động 1: Khởi động

- a) Mục tiêu: HS được làm quen và nhận biết một số khuôn dạng dữ liệu đầu vào của đồ thị dưới dạng các tệp dữ liệu.
- b) Nội dung:
- Em hãy trao đổi, thảo luận và trả lời một số câu hỏi sau:
- Nếu đồ thị là vô hướng thì ma trận kề có đặc điểm gì?
 - Phân biệt sự giống nhau và khác nhau giữa ma trận kề và danh sách kề?
 - Khái niệm bậc của các đỉnh có gì khác nhau trong trường hợp đồ thị là vô hướng, có hướng?
- c) Sản phẩm: Câu trả lời của HS.
1. Nếu đồ thị là vô hướng thì ma trận kề có đặc điểm: Là ma trận đối xứng, nghĩa là với mọi đỉnh i, j ta có $A[i][j] = A[j][i]$.
 2. Phân biệt sự giống nhau và khác nhau giữa ma trận kề và danh sách kề: Sự khác nhau:
 - Khác nhau bởi định nghĩa.
 - Khác nhau về khuôn dạng thể hiện của hai đối tượng này. Sự giống nhau:
 - Đều là bộ dữ liệu xác định duy nhất để biểu diễn một đồ thị.
 - Đều được biểu diễn bằng list trong Python.
 3. Khái niệm bậc của các đỉnh có gì khác nhau trong trường hợp đồ thị là vô hướng, có hướng:

Với đồ thị vô hướng thì bậc ra = bậc vào của các đỉnh, còn với đồ thị có hướng thì bậc vào và bậc ra có thể khác nhau.
- d) Tổ chức thực hiện: GV chia lớp thành 4 nhóm, cho HS thảo luận và trả lời. GV: Nhận xét câu trả lời của HS.

2. Hoạt động 2: Hình thành kiến thức

Hoạt động 2.1. Nhiệm vụ 1: Thực hành viết chương trình hiển thị danh sách kề

- a) Mục tiêu: Viết được chương trình tính danh sách kề từ dữ liệu đầu vào là ma trận kề.
- b) Nội dung: Cho đơn đồ thị vô hướng $G = (V, E)$ được cho bởi ma trận kề A được cho trong tệp văn bản có dạng như hình bên. Yêu cầu xác định của đồ thị trên, kết quả thể hiện ra màn hình, ví dụ như sau:
- Đỉnh kề với 0: 1 2
 Đỉnh kề với 1: 0 2 3
 Đỉnh kề với 2: 0 1 3
 Đỉnh kề với 3: 1 2
- c) Sản phẩm: Chương trình trên máy tính, điện thoại.
- Sản phẩm dự kiến:**

Data.inp
4
0 1 1 0
1 0 1 1
1 1 0 1
0 1 1 0

```
def builgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)] A=[]
for line in f:
    A.append([int(x) for x in line.split()]) f.close()
return V,A
def in_ds_dinhke(A): n=len(A)
for i in range(n):
    print('Đỉnh kề với đỉnh ',i,',end=' ')
    for j in range(n):
        if A[i][j]==1:
            print(j,end=' ')
```

print() fi="data.inp" V,A=builgraph(fi) in ds dinhke(A)

d) Tổ chức thực hiện: 2 HS 1 nhóm, làm bài trên máy tính.

Hoạt động của GV và HS

Bước 1: Chuyển giao nhiệm vụ

GV: Yêu cầu tất cả HS đọc và thực hiện nhiệm vụ 1 SCD trang 62.

Bước 2: Thực hiện nhiệm vụ

HS: Nghiên cứu SCD, viết các chương trình thực hiện theo yêu cầu trên máy tính.

GV: Quan sát và trợ giúp các nhóm HS. Chú ý để in được các đỉnh kề của nhau, phải dựa vào Hướng dẫn tạo các tệp input và cách gọi chạy chương trình.

Bước 3: Báo cáo, thảo luận

HS: Tạo tệp Data.inp, chạy chương trình quan sát kết quả và báo cáo GV. GV: Quan sát bài làm

Bước 4: Kết luận, nhận định

GV nhận xét bài làm của các nhóm, cho điểm cộng với các nhóm làm tốt. Cùng cố kiến thức cần ghi nhớ.

HS: Ghi nhớ chương trình vào vở

Hoạt động 2.2. Nhiệm vụ 2: Thực hành viết chương trình hiển thị ma trận kề, danh sách kề và bậc của đồ thị

a) Mục tiêu: Viết được chương trình từ đầu vào là danh sách các cạnh, đầu ra là ma trận kề A, danh sách kề Adj và chỉ tiết bậc của các đỉnh của đồ thị.

b) Nội dung: Cho đơn đồ thị vô hướng $G = (V, E)$ được cho bởi danh sách các cạnh. Danh sách các cạnh được cho trong tệp văn bản Edges.inp, trong đó dòng đầu tiên là số các đỉnh của đồ thị, các dòng tiếp theo mỗi dòng mô tả một cạnh của đồ thị. Yêu cầu tính ma trận kề, danh sách kề và bậc của tất cả các đỉnh của đồ thị G. Kết quả đưa ra màn hình được thể hiện như sau:

Ma trận kề	Danh sách kề	Bậc của các đỉnh của đồ thị
0 1 1 0	0 1 2	Đỉnh 0: 2
1 0 1 1	1 0 3 2	Đỉnh 1: 3
1 1 0 1	2 0 3 1	Đỉnh 2: 3
0 1 1 0	3 1 2	Đỉnh 3: 2

Edges.inp	
4	
0 1	
0 2	
1 3	
1 2	
2 3	
3 1	
3 2	

c) Sản phẩm: Bài làm trên máy tính, điện thoại, vở ghi.

```
def buildgraph(fname): f=open(fname) n=int(f.readline())
V=[i for i in range(n)]; Adj=[[] for i in range(n)] A=[[0 for i in
range(n)] for j in range(n)]
for line in f: i,j=map(int,line.split()) A[i][j]=1; A[j][i]=1
Adj[i].append(j); Adj[j].append(i) f.close()
```

```

return V,A,Adj
def show_matran(A): n=len(A)
for i in range(n):
for j in range(n): print(A[i][j],end=' ')
print()
def show_dske(Adj): n=len(Adj)
for i in range(n): print(i,end=' ')
for j in range (len(Adj[i])): print(Adj[i][j],end=' ')

```

```

print()
def show_deg(Adj): n=len(Adj)
for i in range(n): print('Đỉnh ',i,'có bậc là:',len(Adj[i])) f1="Edges.inp"
V,A,Adj=buildgraph(f1) print('Ma trận kề:') show_matran(A) print('Danh sách kề')
show_dske(Adj) show_deg(Adj)

```

d) Tổ chức thực hiện: 2 HS 1 nhóm, làm bài trên máy tính.

Hoạt động của GV và HS Bước 1: Chuyển giao nhiệm vụ

GV: Yêu cầu tất cả HS đọc và thực hiện nhiệm vụ 2 SCD trang 63.

Bước 2: Thực hiện nhiệm vụ

HS: Nghiên cứu SCD, viết các chương trình thực hiện theo yêu cầu, tạo các tệp input Edges.inp như mô tả trên. Viết hàm 4 hàm theo yêu cầu như mô tả trên máy tính. Hàm buildgraph(fname) để đọc dữ liệu từ tệp danh sách cạnh, và biểu diễn đồ thị dưới dạng danh sách kề (Adj) và ma trận kề (A).

Hàm show_matran(): để in ra ma trận kề, hàm show_dske(Adj): để in ra danh sách kề.

Hàm show_deg(Adj): để in ra các bậc của đỉnh.

GV: Quan sát và trợ giúp các nhóm HS. Hướng dẫn và cách gọi chạy chương trình.

Bước 3: Báo cáo, thảo luận

HS: Tạo tệp Edges.inp, chạy chương trình quan sát kết quả và báo cáo GV. GV: Quan sát bài làm của những HS đã chạy được trên máy tính.

Bước 4: Kết luận, nhận định

GV nhận xét bài làm của các nhóm, cho điểm cộng với các nhóm làm tốt. Chia sẻ bài làm của nhóm viết chương trình đúng, khoa học cho cả lớp quan sát, học tập. Cùng cố kiến thức cần ghi nhớ.

HS: Ghi nhớ chương trình vào vở.

3. Hoạt động luyện tập

a) Mục tiêu: HS thông hiểu chương trình, chỉnh sửa được chương trình thiết lập đồ thị biểu diễn được đồ thị dưới dạng ma trận kề, danh sách kề.

b) Nội dung: Làm bài tập 1, 2 phần luyện tập SCD trang 64.

1. Trong Nhiệm vụ 1, hàm In_danh_sach_dinh_ke() lấy thông tin từ ma trận kề A. Có thể sử dụng hàm này với dữ liệu là danh sách kề Adj được không? Nếu có thì viết lại hàm này với dữ liệu đầu vào là danh sách kề Adj.

2. Trong Nhiệm vụ 2, chúng ta có thể thấy các đỉnh kề không được in ra theo thứ tự tăng dần của chỉ số trong biểu diễn danh sách kề. Em hãy giải thích tại sao. Có thể chỉnh sửa chương trình để in ra các đỉnh kề theo thứ tự chỉ số tăng dần được không.

c) Sản phẩm: Bài làm của HS trong vở ghi, trên máy tính, điện thoại.

Sản phẩm dự kiến:

1. Hàm `In_danh_sach_dinh_ke()` có thể viết lại với dữ liệu đầu vào `Adj` như sau:

```
def builgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)]
Adj=[] for i in range(n):
for i in range(n):
line=[int(x) for x in f.readline().split()] line=line[1:]
Adj[i].append(line) f.close()
return V,Adj
def in_ds_dinhke(Adj): n=len(Adj)
for i in range(n):
print('Đỉnh kề với đỉnh ',i,':',end=' ')
for j in Adj[i]: print(j,end=' ')
print() fi="data2.inp" V,Adj=builgraph(fi) in_ds_dinhke(Adj)
```

2. Muốn in ra danh sách các đỉnh kề theo thứ tự tăng dần của số thứ tự các đỉnh của đồ thị, cần thực hiện việc sắp xếp từng phần tử của `Adj` ngay từ khi nhập dữ liệu gốc.

d) Tổ chức thực hiện: Giao về nhà làm bài, tiết sau gọi lên bảng chữa bài.

GV gọi 2 HS lên bảng cùng làm bài 1, 1 HS làm bài 2 các HS còn lại mở vở GV kiểm tra bài làm của HS trong vở ghi.

Gọi một số HS nhận xét bài trên bảng.

GV: Nhận xét, cho điểm những HS làm bài tốt, chạy chương trình cho cả lớp quan sát kết quả chương trình.

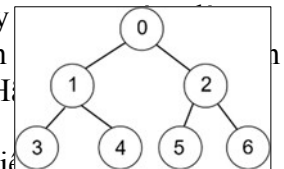
HS: Ghi nhớ vào vở nếu làm sai, làm bài trên máy tính, chạy chương trình.

4. Hoạt động vận dụng

a) Mục tiêu: HS vận dụng kiến thức đã học về đồ thị viết được các hàm dựng đồ thị, biểu diễn đồ thị.

b) Nội dung: Làm bài tập 1, 2 SCD trang 64 phần vận dụng.

1. Cây nhị phân có thể được coi là đồ thị vô hướng, các nút của cây đỉnh, còn quan hệ cha-con là cạnh nối của đồ thị. Với cây nhị phân được đánh số theo chỉ số của mảng biểu diễn tương ứng của cây. Hình của đồ thị tương ứng cây nhị phân ở hình bên.



2. Viết lại các hàm thiết lập đồ thị `BuildGraph(fname)` với tệp dữ liệu chứa danh sách các cạnh của đồ thị. Đầu ra của hàm là dãy các giá trị `V`, `E`, `A`, `Adj`. Viết hàm cho cả hai trường hợp đồ thị vô hướng và đồ thị có hướng.

c) Sản phẩm: Bài làm của HS trong vở ghi.

1. Ma trận kề của đồ thị tương ứng như sau:

```
(0110000)
|
1001100 |
|
1000011 |
|
A 0100000 |
|
0100000 |
|
0010000 |
|
0010000 |
```

2. Giả sử `fname` là tên của tệp dữ liệu chứa danh sách các cạnh của đồ thị. Chúng ta sẽ thiết lập hàm `buildgraph(fname)` đọc dữ liệu từ tệp này và trả về các thông số `V`, `E`, `A`,

Adj của đồ thị. Cần thiết lập riêng cho trường hợp đồ thị vô hướng và có hướng.

a) Trường hợp đồ thị vô hướng:

```
def BuildGraph(fname):
    f = open(fname)
    n = int(f.readline())
    A = [[0]*n for i in range(n)] V = [i for i in range(n)]
    E = []
    Adj = [[] for i in range(n)]
    for line in f:
        edge = [int(ch) for ch in line.split()]
        i,j = edge[0],edge[1];
        Adj[i].append(j);
        Adj[j].append(i)
    return V,E,A,Adj
```

b) Trường hợp đồ thị có hướng:

```
def BuildGraph(fname):
    f = open(fname)
    n = int(f.readline())
    A = [[0]*n for i in range(n)] V = [i for i in range(n)]
    E = []
    Adj = [[] for i in range(n)]
    for line in f:
        edge = [int(ch) for ch in line.split()] i,j = edge[0],edge[1]
        A[i][j] = 1
        Adj[i].append(j)
    E.append((i,j))
```

f.close() return V,E,A,Adj

d) Tổ chức thực hiện: HS làm ở nhà, tiết sau GV gọi 3 HS lên bảng chữa bài.

GV gọi 3 HS lên bảng làm bài, các HS còn lại mở vở GV kiểm tra bài làm của HS trong vở ghi hoặc trên máy tính.

Gọi một số HS nhận xét bài làm của các bạn trên bảng.

GV: Nhận xét, cho điểm những HS làm bài tốt, chạy chương trình cho cả lớp quan sát kết quả chương trình.

HS: Ghi nhớ vào vở nếu làm sai, làm bài trên máy tính, chạy chương trình.

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phần vận dụng.
- Đọc trước Bài 14. Kỹ thuật duyệt đồ thị theo chiều sâu.

BÀI 14. KỸ THUẬT DUYỆT ĐỒ THỊ THEO CHIỀU SÂU

Thời gian thực hiện: 3 tiết

I. MỤC TIÊU

1. Kiến thức

- Bài toán duyệt đồ thị.
- Thuật toán duyệt đồ thị theo chiều sâu.
- Mô phỏng thuật toán duyệt đồ thị theo chiều sâu.

2. Năng lực

- Năng lực chung:
- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.
- Năng lực Tin học:
- Trình bày được ý tưởng của duyệt đồ thị theo chiều sâu.
- Mô phỏng được thuật toán duyệt theo chiều sâu.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành các bài tập theo yêu cầu của GV thông qua hệ thống câu hỏi, phiếu học tập, thông qua sản phẩm.

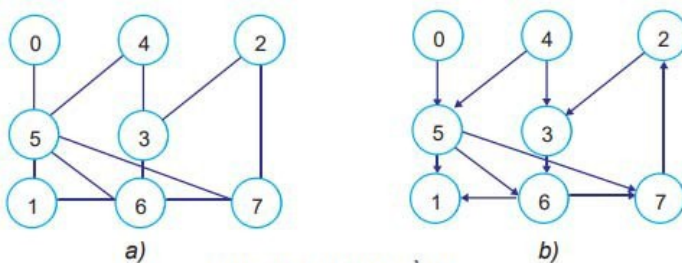
II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, máy tính, máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước Bài 14. Kỹ thuật duyệt đồ thị theo chiều sâu.

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động 1: Khởi động

- a) Mục tiêu: HS làm quen với bài toán duyệt các đỉnh của đồ thị.
- b) Nội dung: Quan sát đồ thị Hình 14.1 và cho biết việc duyệt các đỉnh của đồ thị sẽ được thực hiện như thế nào? Sử dụng kiểu dữ liệu nào để cài đặt cho phù hợp?



Hình 14.1. Duyệt đồ thị

- c) Sản phẩm: Câu trả lời của HS.
- d) Tổ chức thực hiện: GV cho HS quan sát Hình 14.1 SCD trang 65 và trả lời. GV: Nhận xét câu trả lời của HS.

2. Hoạt động 2: Hình thành kiến thức

Hoạt động 2.1: Tìm hiểu duyệt đồ thị

- a) Mục tiêu: Thông qua một ví dụ cụ thể HS làm quen và biết được cách duyệt đầu tiên: duyệt theo chiều sâu (DFS).
- b) Nội dung: HS hoàn thành phiếu học tập số 1.

PHIẾU HỌC TẬP SỐ 1

Chia lớp thành 6 nhóm, 3 nhóm 1, 3, 5 duyệt đồ thị Hình 14.1a với đỉnh xuất phát là 0, 3 nhóm 2, 4, 6 duyệt đồ thị Hình 14.1a với đỉnh xuất phát là 2.

Ghi lại các bước duyệt đồ thị vào bảng sau: đỉnh đã duyệt ghi màu đỏ, đỉnh chưa duyệt ghi màu đen với nhóm 1, 3, 5. Còn nhóm 2, 4, 6 ghi lại bằng hình ảnh đỉnh thăm rồi ghi màu đen chưa thăm ghi màu đen.

STT	Đỉnh hiện thời	Danh sách đỉnh kề	Duyệt (đánh dấu)	Tìm các đỉnh chưa duyệt	Trạng thái
1					
2					
...					
15					

c) Sản phẩm: Hoàn thành phiếu học tập số 1.

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và hoàn thành phiếu học tập số 1.

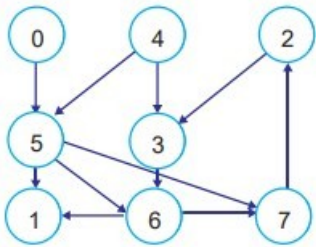
Hoạt động của GV và HS	Sản phẩm dự kiến
-------------------------------	-------------------------

Bước 1: Chuyển giao nhiệm vụ

GV: Phát phiếu học tập số 1.
cho các nhóm nghiên cứu SCD và hoàn thành.

Câu 1. Thứ tự các đỉnh trong danh sách kề có ảnh hưởng đến thứ tự các đỉnh được duyệt của thuật toán DFS không?

Câu 2. Mô tả quá trình duyệt theo chiều sâu của đồ thị có hướng trong Hình 14.1b nếu xuất phát từ đỉnh 4.

**Bước 2: Thực hiện nhiệm vụ**

HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt hoàn thành phiếu học tập số 1.

GV: Quan sát và trợ giúp các nhóm HS.

GV: Gọi 1 HS trả lời câu hỏi cùng số 1, câu 2 tất cả HS làm độc lập vào vở ghi.

Bước 3: Báo cáo, thảo luận

HS: Nộp phiếu học tập khi đã hoàn thành.

GV: Chiếu sản phẩm cho cả lớp quan sát so sánh kết quả các nhóm.

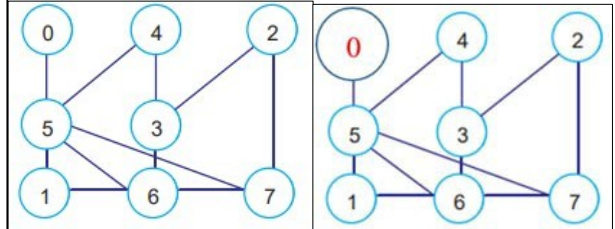
Kiểm tra vở ghi của một số HS.

Bước 4: Kết luận, nhận định

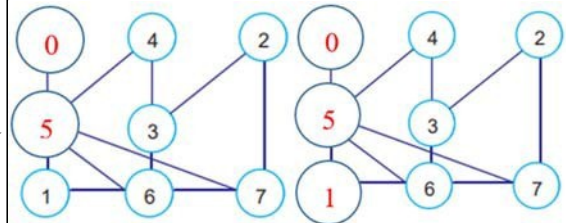
GV nhận xét bài làm của các nhóm. Cùng cố kiến thức cần ghi nhớ.

1. Bài toán duyệt đồ thị

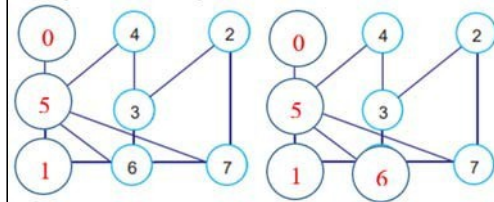
Duyệt đồ thị theo chiều sâu (DFS) bắt đầu từ việc “duyet theo chiều sâu từ đỉnh u” chưa được duyệt: duyệt đỉnh u, sau đó lần lượt xét các đỉnh kề v của đỉnh u, nếu có đỉnh v chưa được duyệt thì “duyet theo chiều sâu từ đỉnh v”, ngược lại quay lui về bước duyệt trước đó. Lặp lại cách duyệt này cho đến khi tất cả các đỉnh của đồ thị đã được duyệt.



Xuất phát từ đỉnh 0, tô màu cho đỉnh đó, tìm các đỉnh kề với đỉnh 0 là đỉnh 5.



Tô màu cho đỉnh 5, tìm các đỉnh kề với đỉnh 5 là [0, 1, 4, 6, 7]. Đỉnh 0 đã thăm, đỉnh 1, 4, 6, 7 chưa thăm, thăm đỉnh 1, tô màu cho đỉnh 1.



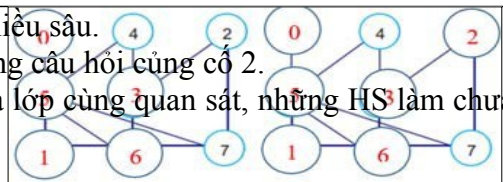
Kề với đỉnh 1 là đỉnh [6, 5]. Đỉnh 6 chưa thăm, nên thăm đỉnh 6, tô màu đỉnh 6.

Kề với đỉnh 6 là các đỉnh [5, 3, 7] đỉnh 3, 7 chưa thăm nên thăm đỉnh 3, tô màu đỉnh 3.

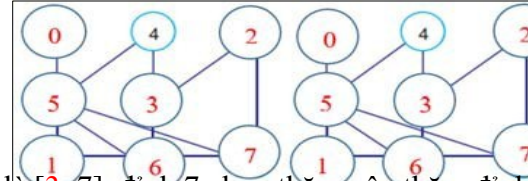
HS: Ghi nhớ kiến thức ý tưởng duyệt theo chiều sâu.

GV: Đánh giá cho điểm một số HS làm đúng câu hỏi củng cố 2.

Chia sẻ bài làm của HS lên máy chiếu cho cả lớp cùng quan sát, những HS làm chưa đúng rút

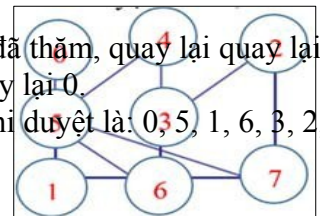


Kề với 3 là [6, 2, 4], đỉnh 2, 4 chưa thăm nên thăm đỉnh 2. Tô màu đỉnh 2.



Kề với 2 là [3, 7], đỉnh 7 chưa thăm nên thăm đỉnh 7.
Kề với 7 là đỉnh [2, 5, 6] các đỉnh đều thăm rồi, nên quay lại 2.

Kề với 4 là [5, 3] đã thăm, quay lại quay lại 3, kề với 3 là 6 đã thăm quay lại 2, quay lại 7, quay lại 6, quay lại 5, quay lại 4. Kết thúc quá trình duyệt. thứ tự đỉnh thăm là: 4, 5, 1, 6, 3, 2, 4, 7.



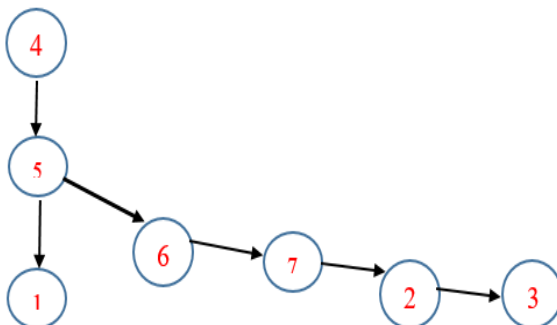
Sản phẩm câu 2 củng cố SCD trang 67: Duyệt đồ thị Hình 14.1b xuất phát từ đỉnh 4

Tô màu cho đỉnh 4, kề với 4 là các đỉnh [5, 3] tô màu cho đỉnh 5, kề với 5 là [1, 6, 7] tô màu

cho đỉnh 1, kề với 1 không có đỉnh nào quay lại 5, kề với 5 là [1, 6, 7] thăm đỉnh 6, tô màu cho 6, kề với 6 là 7, thăm và tô màu cho 7. Kề với 7 là 2, thăm và tô màu cho 2. Kề với 2 là 3, thăm và tô màu cho 3, kề với 3 là 6 đã thăm quay lại 2, quay lại 7, quay lại 6, quay lại 5,

quay lại 4. Kết thúc quá trình duyệt. thứ tự đỉnh thăm là 4, 5, 1, 6, 7, 2, 3.

Đồ thị sau khi duyệt xuất phát từ đỉnh 4 có dạng như sau:



Hoạt động 2.2: Tìm hiểu Thuật toán duyệt theo chiều sâu DFS

- a) Mục tiêu: HS biết và nắm được kĩ thuật cài đặt cụ thể thuật toán duyệt đồ thị theo chiều sâu DFS bằng đệ quy.
- b) Nội dung: Quan sát và thảo luận tìm hiểu thuật toán duyệt chiều sâu trên đồ thị vô hướng, có hướng.
- c) Sản phẩm: Viết được hàm duyệt chiều sâu cho đồ thị vô hướng, có hướng.
- d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và trả lời các câu hỏi.

Hoạt động của GV và HS	Sản phẩm dự kiến
<p>Bước 1: Chuyển giao nhiệm vụ</p> <p>GV: Yêu cầu các nhóm nghiên cứu SCD trang 68, 69. Trả lời các câu hỏi sau:</p> <ol style="list-style-type: none"> 1. Dùng danh sách mark để chứa thông tin gì? 2. mark = [False]*len(V) để làm gì? 3. Tìm các đỉnh kề với u lấy từ đâu? 4. not mark[u] cho kết quả là gì? 5. Hàm builgraph(fname) có sẵn hay phải xây dựng? 6. Hàm DFS(Adj, u) dùng để làm gì? <p>Giải thích ý nghĩa từng câu lệnh trong hàm.</p> <ol style="list-style-type: none"> 7. Hàm DFS_Traversal(G) có nhiệm vụ gì? Giải thích ý nghĩa từng câu lệnh trong hàm. 8. Hàm builgraph(fname) để làm gì? 9. Để in ra thứ tự các đỉnh sau khi duyệt làm thế nào? 10. Nếu đồ thị G chỉ bao gồm các đỉnh biệt lập, không có cạnh nào thì thuật toán duyệt sâu DFS sẽ được thực hiện như thế nào? <p>Bước 2: Thực hiện nhiệm vụ</p> <p>HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt trả lời các câu hỏi trên.</p> <p>GV: Quan sát và trợ giúp HS.</p> <p>Bước 3: Báo cáo, thảo luận</p> <p>GV: Gọi lần lượt một số HS trả lời các câu hỏi trên.</p> <p>Bước 4: Kết luận, nhận định</p>	<p>2. Thuật toán duyệt chiều sâu DFS</p> <p>Cho đồ thị $G = (V, E)$ vô hướng, có n đỉnh lưu trong tệp graph.inp gồm có dòng đầu tiên chữ số nguyên dương n là số lượng đỉnh của đồ thị, nhiều dòng sau mỗi dòng mô tả cạnh (u, v) có nghĩa u và v kề với nhau. Mô tả ý tưởng DFS:</p> <ul style="list-style-type: none"> + Xây dựng đồ thị biểu diễn dạng danh sách kề. + Khởi tạo các danh sách mark = [False] chứa n đỉnh lúc đầu chưa thăm gán là False, đỉnh nào thăm sẽ là True. + Với mỗi đỉnh u, tìm các đỉnh kề với u nếu chưa thăm thì gọi DFS. <p>Đoạn mã giả: DFS_Traversal(G):</p> <ul style="list-style-type: none"> + Thiết lập tất cả các đỉnh u thuộc V là chưa đánh dấu. + Với mỗi đỉnh u thuộc V. <p>Nếu u chưa đánh dấu: DFS(Adj, u).</p> <p>Hàm đệ quy DFS(Adj, u) thực hiện thuật toán duyệt theo chiều sâu bắt đầu từ đỉnh u chưa thăm:</p> <p>DFS(Adj, u):</p> <ol style="list-style-type: none"> 1. Đánh dấu đỉnh u. 2. Với mỗi đỉnh v là đỉnh kề của đỉnh u. Nếu đỉnh v chưa đánh dấu: DFS(Adj, v). Cài đặt: <pre>def DFS(Adj,u): mark[u]=True for v in Adj[u]: if not mark[v]: DFS(Adj,v) def DFS_Traversal(V,Adj): mark=[False]*len(V) for u in V:</pre>

GV nhận xét các câu trả lời.

Viết chương trình hoàn chỉnh duyệt theo chiều sâu DFS với đồ thị đã cho vô hướng,

chạy chương trình cho HS quan sát kết quả.

HS: Ghi nhớ thuật toán duyệt theo chiều sâu và chương trình cài đặt duyệt theo chiều sâu.

Chú ý khi gọi hàm DFS muốn xuất phát từ đỉnh nào thì truyền đỉnh đó vào DFS.

GV: Nếu đồ thị G chỉ bao gồm các đỉnh biệt lập thì tất cả các lệnh DFS(v) sẽ chỉ duyệt được đúng một đỉnh v.

```

if not mark[u]: DFS(Adj,u)
# Chương trình chính Fname= "graph.inp" V,Adj=Buildgraph(fname)
DFS_Traversal(V,Adj)

```

Code chương trình:

```

def Buildgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)]
Adj=[] for i in range(n)
for line in f: edge=[int(x) for x in
line.split()]
i,j=edge[0],edge[1]
Adj[i].append(j) Adj[j].append(i)# nếu đồ thị vô
hướng
f.close() return V,Adj
def DFS(Adj,u): mark[u]=True print(u, end=' ') for v in Adj[u]:
if not(mark[v]): DFS(Adj,v)
fi='graph.inp' V,Adj=buildgraph(fi) mark=[False]*(len(V)+2) DFS(Adj,0)

```

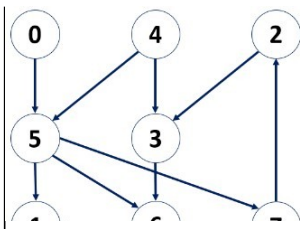
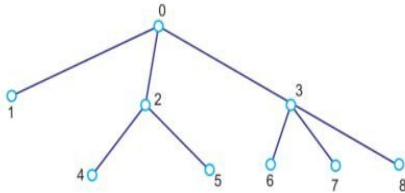
Hoạt động 2.3: Tìm hiểu thuật toán duyệt theo chiều sâu DFS không đệ quy

- Mục tiêu: HS biết và cài đặt được thuật toán duyệt theo chiều sâu không đệ quy sử dụng ngăn xếp.
- Nội dung: Quan sát và thảo luận tìm hiểu thuật toán duyệt chiều sâu trên đồ thị vô hướng, có hướng không sử dụng đệ quy.
- Sản phẩm: Viết hàm duyệt chiều sâu không đệ quy.
- Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và trả lời các câu hỏi và viết chương trình duyệt DFS.

Hoạt động của GV và HS	Sản phẩm dự kiến
------------------------	------------------

<p>Bước 1: Chuyển giao nhiệm vụ GV: Yêu cầu các nhóm nghiên cứu SCD trang 70. Thực hiện các yêu cầu sau:</p> <ol style="list-style-type: none"> Viết hàm DFS(Adj, u) sử dụng ngăn xếp để lưu các đỉnh đã thăm. Viết hàm DFS_Traversal(V, Adj) để duyệt tất cả các đỉnh chưa thăm. Hàm BuildGraph() dùng để làm gì, hàm này có sẵn trong thư viện nào? Nếu áp dụng thuật toán duyệt DFS() cho đồ thị có hướng trong Hình 14.1b thì thứ tự các đỉnh được đánh dấu sẽ theo thứ tự nào? Với đồ thị Hình 14.3 thì thứ tự duyệt theo chiều sâu, bắt đầu từ đỉnh 0 sẽ như thế nào? (theo 2 cách đệ quy và không đệ quy). <p>Bước 2: Thực hiện nhiệm vụ HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt trả lời các câu hỏi trên.</p> <p>GV: Quan sát và trợ giúp HS.</p> <p>Bước 3: Báo cáo, thảo luận GV: Gọi lần lượt một số HS trả lời các câu hỏi trên.</p> <p>Bước 4: Kết luận, nhận định GV nhận xét các câu trả lời. Viết chương trình hoàn chỉnh duyệt theo chiều sâu DFS với đồ thị đã cho vô hướng, chạy chương trình cho HS quan sát kết quả. HS: Ghi nhớ kiến thức chương trình duyệt theo chiều sâu. Chú ý khi gọi hàm DFS muốn xuất phát từ đỉnh nào thì truyền đỉnh đó vào DFS. Nếu đồ thị G chỉ bao gồm các đỉnh biệt lập thì tất cả các lệnh DFS(v) sẽ chỉ duyệt được đúng một đỉnh v.</p>	<p>Thuật toán duyệt không đệ quy Dùng ngăn xếp S để chứa các đỉnh là kẻ chưa thăm, đỉnh u thăm rồi thì bỏ ra khỏi ngăn xếp và đánh dấu đỉnh u đã thăm bằng lệnh mark[v] = True.</p> <pre>def Buildgraph(fname): f=open(fname) n=int(f.readline()) V=[i for i in range(n)] Adj=[[] for i in range(n)] for line in f: u,v=map(int,line.split()) Adj[u].append(v) Adj[v].append(u) # đồ thị vô hướng f.close() return V,Adj def DFS(Adj, u): S=[] # khởi tạo ngăn xếp rỗng S.append(u); while len(S)!=0: # ngăn xếp khác rỗng v=S.pop()# lấy đỉnh v ở đỉnh ngăn xếp ra if not mark[v]:# v chưa thăm mark[v]=True # thăm v print(v,end=' ') for w in Adj[v]:# w là kẻ của v S.append(w) # cho w vào ngăn xếp fi='graph.inp' V,Adj=buildgraph(fi) mark=[False]*len(V) DFS(Adj,0)</pre> <p>Củng cố kiến thức 1. Thứ tự các đỉnh được duyệt theo thuật toán DFS đệ quy là: 0 5 1 6 7 2 3 4, còn nếu thực hiện theo DFS không đệ quy thì thứ tự duyệt là: 0 5 7 2 3 6 1 4.</p>
--	--

		8	8
	0	5	0 0 0 0 0 1 0 0
	1		0 0 0 0 0 0 0 0



Hình 14.3

2 3	0 0 0 1 0 0 0 0
3 6	0 0 0 0 0 0 1 0
4 3 5	0 0 0 1 0 1 0 0
5 1 6 7	0 1 0 0 0 0 1 1
6 1 7	0 0 0 0 0 0 0 1
7 2	0 0 1 0 0 0 0 0
Danh sách kề	Ma trận kề

2. Thứ tự các đỉnh được duyệt theo thuật toán DFS để tìm đường đi ngắn nhất từ đỉnh 0 là: 0 1 2 4 5 3 6 7 8, còn nếu thực hiện theo DFS không đệ quy thì thứ tự duyệt là: 0 3 8 7 6 2 5 4 1.

Đồ thị biểu diễn dạng danh sách kề:

```

9
0 1 2 3
1 0
2 0 4 5
3 0 6 7 8
4 2
5 2
6 3
7 3
8 3

```

3. Hoạt động luyện tập

a) Mục tiêu: HS tìm được bậc của các đỉnh, biểu diễn được đồ thị dưới dạng ma trận kề, danh sách kề.

b) Nội dung: Làm bài tập 1, 2 phần luyện tập SCD trang 71.

1. Sửa chương trình của thuật toán DFS không đệ quy sao cho chương trình trong khi duyệt sẽ in ra các bước với thông tin sau:

- Thông tin ngăn xếp (hiện thời).
- Phần tử được lấy ra từ ngăn xếp.
- Phần tử được đánh dấu để chuẩn bị cho bước sau (mark).

2. Hàm DFS(Adj, u) có thể viết theo một cách khác, việc kiểm tra đỉnh u đã đánh dấu chưa được đưa vào bên trong hàm như sau:

```

1 def DFS(Adj,u):
2     if not mark[u]:
3         mark[u] = True # Đánh dấu đỉnh u
4         for v in Adj[u]:
5             DFS(Adj,v)

```

a) Trong trường hợp này, phần chương trình chính cần viết lại như thế nào?

b) Viết lại toàn bộ chương trình duyệt đồ thị theo chiều sâu theo hàm mô tả trên. Cách duyệt này có tương đương với cách đã thực hiện trong Hoạt động 2 không?

c) Sản phẩm: Bài làm của HS trong vở ghi, trên máy tính, điện thoại.

Sản phẩm dự kiến:

```

fname="Graphh14.1a.inp"           # Trong Stack có các hàm khởi tạo, kiểm tra ngăn xếp
rỗng
def Buildgraph(fname):             # thêm vào đỉnh ngăn xếp, lấy phần tử ở đỉnh ngăn xếp
ra
# dựng dt danh sách kề fi=open(fname) n=int(fi.readline()) V=[i for i in range(n)]
Adj=[[] for i in range(n)]
for i in range(n):
line=[int(x) for x in fi.readline().split()] line=line[1:]
Adj[i].extend(line) fi.close()
return V, Adj
def DFS(s):
S=Stack() push(S,s)
while not isEmptyStack(S): st=str(S)
print(S, " *(30-len(st)),sep="",end="")
v=pop(S)
print(v, " *(10-len(str(v))),sep="",end="")
if not mark[v]: mark[v]=True print(v,end= " ") for u in Adj[v]:
push(S,u)
else:
print('Not found',end="") print()
# chương trình chính V,Adj=Buildgraph(fname) n=len(V) mark=[False]*n
print("Ngăn xếp                pop(S)      phần tử đánh dấu")
for s in V:
if not(mark[s]): DFS(s)

```

Kết quả chạy chương trình như sau:

Ngăn xếp	pop(S)	phần tử đánh dấu
[0]	0	0
[5]	5	5
[0, 1, 4, 6, 7]	7	7
[0, 1, 4, 6, 2, 5, 6]	6	6
[0, 1, 4, 6, 2, 5, 1, 3, 5, 7]	7	Not found
[0, 1, 4, 6, 2, 5, 1, 3, 5]	5	Not found
[0, 1, 4, 6, 2, 5, 1, 3]	3	3
[0, 1, 4, 6, 2, 5, 1, 2, 4, 6]	6	Not found
[0, 1, 4, 6, 2, 5, 1, 2, 4]	4	4
[0, 1, 4, 6, 2, 5, 1, 2, 3, 5]	5	Not found
[0, 1, 4, 6, 2, 5, 1, 2, 3]	3	Not found
[0, 1, 4, 6, 2, 5, 1, 2]	2	2
[0, 1, 4, 6, 2, 5, 1, 3, 7]	7	Not found
[0, 1, 4, 6, 2, 5, 1, 3]	3	Not found
[0, 1, 4, 6, 2, 5, 1]	1	1
[0, 1, 4, 6, 2, 5, 5, 6]	6	Not found
[0, 1, 4, 6, 2, 5, 5]	5	Not found
[0, 1, 4, 6, 2, 5]	5	Not found
[0, 1, 4, 6, 2]	2	Not found
[0, 1, 4, 6]	6	Not found
[0, 1, 4]	4	Not found
[0, 1]	1	Not found
[0]	0	Not found

Graphh14.1a.input

```

8
0 5
1 5 6
2 3 7
3 2 4 6
4 3 5
5 0 1 4 6 7
6 1 3 5 7
7 2 5 6

```

2. Chương trình chính trong trường hợp này cần viết lại như sau:

```

V, Adj = Buildgraph(fname) n = len(V)
Mark = [False]*n
for s in V: DFS(s)

```

Thứ tự các đỉnh đồ thị được duyệt vẫn hoàn toàn giống như thứ tự các đỉnh đã duyệt của thuật toán DFS() theo phương án DFS đệ quy đã trình bày trong sách chuyên đề. Như vậy thuật toán DFS “mới” này hoàn toàn tương đương với thuật toán duyệt DFS đã

mô tả trong SCD.

d) Tổ chức thực hiện: **Giao về nhà làm bài**, tiết sau gọi lên bảng chữa bài.

GV gọi 2 HS lên bảng cùng làm bài 1, 1 HS làm bài 2 các HS còn lại mở vở GV kiểm tra bài làm của HS trong vở ghi.

Gọi một số HS nhận xét bài trên bảng.

GV: Nhận xét, cho điểm những HS làm bài tốt, chạy chương trình cho cả lớp quan sát kết quả.

4. Hoạt động vận dụng

a) Mục tiêu: HS vận dụng kiến thức đã học về đồ thị viết được các hàm dựng đồ thị, biểu diễn đồ thị.

b) Nội dung: Làm bài tập 1, 2, 3 SCD trang 71 phần vận dụng.

1. Viết chương trình in ra thứ tự các đỉnh đã được duyệt bằng thuật toán DFS theo cả hai cách đệ quy và không đệ quy, áp dụng cho đồ thị có hướng Hình 14.1b trong phần Khởi động.

2. Các thuật toán DFS() đã mô tả trong các phần trên đều thực hiện trên đồ thị được biểu diễn bằng danh sách kề Adj. Hãy viết lại hàm DFS() được thực hiện trên đồ thị được biểu diễn bằng ma trận kề A.

3. Cho đồ thị $G = (V, E)$ và hai đỉnh s, t bất kì. Chứng minh tính chất: Tồn tại đường đi từ s đến t khi và chỉ khi quá trình duyệt theo chiều sâu từ đỉnh s sẽ duyệt qua đỉnh t , hay nói cách khác, quá trình duyệt theo chiều sâu từ đỉnh s sẽ đi qua tất cả các đỉnh nằm trong thành phần liên thông chứa đỉnh s , trong đó có đỉnh t .

c) Sản phẩm: Bài làm của HS trong vở ghi.

Sản phẩm dự kiến:

1. Viết chương trình in ra thứ tự các đỉnh đã được duyệt bằng thuật toán DFS theo đệ quy và không đệ quy cho đồ thị Hình 14.1b, đồ thị biểu diễn dưới dạng danh sách kề.

– Hàm đọc dữ liệu từ tệp để trả về bộ V, Adj cho đồ thị có hướng như sau:

```
def BuildGraph(fname):
```

```
    """ Thiết lập đồ thị có hướng theo dữ liệu danh sách đỉnh kề """ f = open(fname)
```

```
    n = int(f.readline())
```

```
    V = [i for i in range(n)] Adj = [[] for i in range(n)] for i in range(n):
```

```
    line = [int(x) for x in f.readline().split()] Adj[i].extend(line[1:])
```

```
    f.close()
```

```
    return V, Adj
```

Hàm chính của thuật toán duyệt DFS() đệ quy như sau, có in ra các đỉnh đã duyệt:

```
def DFS(s): # DFS đệ quy mark[s] = True
```

```
    print(s, end = " ") for v in Adj[s]: if not mark[v]: DFS(v)
```

– Hàm chính của thuật toán duyệt DFS() không đệ quy như sau, có in ra các đỉnh đã duyệt:

```
def DFS(s): # DFS không đệ quy S = Stack()
```

```
    push(S, s)
```

```
    while not isEmptyStack(S): v = pop(S)
```

```
    if not mark[v]:
```

```
mark[v] = True print(v, end = " ") for u in Adj[v]: push(S, u)
```

Dưới đây là đoạn chương trình chính, áp dụng cho cả hai phương án sử dụng DFS đệ quy hoặc không đệ quy. Chú ý tại dòng lệnh 2 sẽ ghi tên tệp dữ liệu đầu vào là danh sách kê:

```
from Stack import *
fname = "Graph14.1b_adj.inp" V, Adj = BuildGraph(fname) n = len(V)
mark = [False]*n
for s in V:
if not mark[s]: DFS(s)
```

2. Các thuật toán DFS() nếu viết lại theo mô tả là ma trận kề A sẽ được viết như sau:

a) Thuật toán DFS() đệ quy.

```
def DFS(A, s): mark[s] = True
for j in range(len(A)):
if A[s][j] == 1:
if not mark[j]:DFS(j)
```

b) Thuật toán DFS() không đệ quy (sử dụng Stack).

```
def DFS(A, u):
S = Stack() # khởi tạo Stack rỗng push(S, u)
while not isEmpty(S): v = pop(S)
if not mark[v]: mark[v] = True
for j in range(len(A)):
if A[v][j] == 1:
if not mark[j]:push(S, j)
```

3. Mệnh đề: “Thuật toán DFS(s) sẽ duyệt qua đỉnh t khi và chỉ khi tồn tại đường đi từ s đến t” có thể chứng minh bằng quy nạp theo số các cạnh của đường đi từ s đến t.

– Nếu số cạnh = 0 hay $t = s$ thì hiển nhiên DFS(s) sẽ đánh dấu s đầu tiên tức là đánh dấu t.

– Giả sử mệnh đề đã được chứng minh cho tất cả các đỉnh có đường đi độ dài nhỏ hơn k tính từ s. Xét đỉnh t với điều kiện có đường đi độ dài k tính từ s. Giả sử p là một đường đi như vậy.

$p = v_0 = s, v_1, v_2, \dots, v_{k-1}, v_k = t.$

Đặt $u = v_{k-1}$. Theo giả thiết quy nạp thì hàm DFS(s) sẽ đánh dấu u tại một thời điểm nào đó. Theo cách thiết lập DFS() thì sau khi đánh dấu u, DFS sẽ gọi đến tất cả các đỉnh nằm trong Adj(u), nhưng theo giả thiết thì t là đỉnh kề với u, do vậy t nằm trong Adj[u] do đó DFS(s) sẽ đánh dấu đỉnh t, đó điều phải chứng minh.

d) Tổ chức thực hiện: HS **làm ở nhà**, tiết sau GV gọi 3 HS lên bảng chữa bài. Bài 1, 2 làm trên máy tính, chiếu chương trình cho cả lớp quan sát nhận xét.

Bài 2 làm trên bảng.

GV: Gọi một số HS nhận xét, đánh giá, chấm điểm. củng cố các đoạn chương trình cho HS hiểu.

HS: Ghi nhớ vào vở nếu làm sai.

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phần vận dụng.
- Đọc trước Bài 15. Thực hành duyệt đồ thị theo chiều sâu.

BÀI 15. THỰC HÀNH DUYỆT ĐỒ THỊ THEO CHIỀU SÂU

Thời gian thực hiện: 2 tiết thực hành

I. MỤC TIÊU

1. Kiến thức

- Khái niệm chu trình.
- Kiểm tra đồ thị cho trước có tồn tại chu trình hay không.

2. Năng lực

- Năng lực chung:
- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.
- Năng lực Tin học:
- Thực hành duyệt đồ thị theo chiều sâu.
- Kiểm tra được đồ thị có chu trình hay không.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành nhiệm vụ, các bài tập theo yêu cầu của GV thông qua sản phẩm.

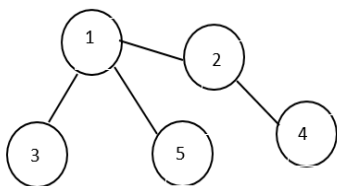
II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, máy tính, máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước bài 15. Thực hành duyệt đồ thị theo chiều sâu.

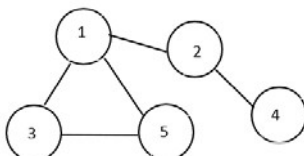
III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động 1: Khởi động

- a) Mục tiêu: HS ôn lại khái niệm chu trình trong đồ thị.
 b) Nội dung: HS hãy trao đổi, thảo luận và trả lời câu hỏi sau:
 Thế nào là 1 chu trình? Quan sát hai đồ thị sau đồ thị nào có chu trình?



Hình 1.a



Hình 1.b

- c) Sản phẩm: Câu trả lời của HS.

Trong đồ thị, chu trình là một đường đi khép kín, mỗi đỉnh đi qua đúng 1 lần, đỉnh xuất

phát và đỉnh kết thúc trùng nhau.

Hình 1a là đồ thị không có chu trình, Hình 1b là đồ thị có chu trình.

d) Tổ chức thực hiện: HS thảo luận và trả lời.

GV: Nhận xét câu trả lời của HS, củng cố khái niệm chu trình trong đồ thị.

2. Hoạt động 2: Hình thành kiến thức

Nhiệm vụ: Hệ thống chuyên đề học tập

a) Mục tiêu: HS biết và hiểu thuật toán kiểm tra một đồ thị có chu trình hay không và áp dụng giải bài toán có ý nghĩa cụ thể trên thực tế.

b) Nội dung: Viết chương trình giải bài toán sau:

Trường em từ năm học này sẽ tổ chức mở rất nhiều chuyên đề học tập cho HS lựa chọn, các chuyên đề sẽ được học trong các thời gian khác nhau. Các chuyên đề được đánh số từ 0 đến $n-1$ với n là số tự nhiên. Tuy nhiên giữa các chuyên đề này có quan hệ ràng buộc kiến thức, ví dụ quan hệ (i,j) chỉ ra rằng muốn học chuyên đề i thì cần học trước chuyên đề j .

Data.inp
5
1 2
2 4
4 1
1 0

Dữ liệu đầu vào dưới dạng tệp văn bản Data.inp như sau:

- Dòng đầu tiên là số tự nhiên n (số các chuyên đề học tập của trường em).
- Các dòng tiếp theo mô tả quan hệ dạng (i,j) , mỗi dòng là hai số i và j cách nhau bởi dấu cách.

Hệ thống các chuyên đề của nhà trường được gọi là hợp lý nếu về nguyên tắc mỗi HS đều có thể đăng kí để học tất cả các chuyên đề.

Viết chương trình nhập bộ dữ liệu trên, kiểm tra và thông báo hệ thống chuyên đề có hợp lý hay không.

Ví dụ với bộ dữ liệu trên thì hệ thống không hợp lý vì muốn học chuyên đề 1 phải học chuyên đề 2, muốn học chuyên đề 2 thì cần phải học chuyên đề 4, muốn học chuyên đề 4 thì phải học chuyên đề 1 điều này mâu thuẫn. Vậy hệ thống không hợp lý.

c) Sản phẩm: Chương trình HS viết trên máy tính.

Sản phẩm dự kiến:

```
def buildgraph(fname): f=open(fname,"r") n=int(f.readline()) Adj=[[ ] for i in range(n)]
V=[int(i) for i in range(n)]
for line in f: i,j=map(int,line.split()) Adj[i].append(j)
f.close() return V,Adj
```

```
def DFS_acyclic(Adj,s): status[s]=1
for v in Adj[s]:
if status[v]==1: return False
elif status[v]==0:
if not DFS_acyclic(Adj,v): return False
status[s]=2 return True
def Acyclic(V,Adj):
for u in V:
```

```

if status[u]==0:
if not DFS_acyclic(Adj,u): return False
return True
# chương trình chính f1="data15.inp" V,Adj=buildgraph(f1) status=[0]*len(V)
if Acyclic(V,Adj): print("Hệ thống chuyên đề học tập hợp lý")
else: print("Hệ thống chuyên đề học tập không hợp lý")

```

d) Tổ chức thực hiện: 2 HS 1 nhóm, làm bài trên máy tính.

Hoạt động của GV và HS

Bước 1: Chuyển giao nhiệm vụ

GV: Yêu cầu tất cả HS đọc và thực hiện nhiệm vụ SCD trang 72, nêu câu hỏi cho HS.

1. Mỗi chuyên đề coi là 1 đỉnh hay 1 cạnh của đồ thị?

2. Mỗi quan hệ ràng buộc (i,j) coi là một gì trong đồ thị?

3. Đồ thị là có hướng hay vô hướng.

4. Để kiểm tra xem hệ thống các chuyên đề đưa ra có hợp lý không, bản chất là kiểm tra đồ thị có đặc điểm gì?

5. Để duyệt các đỉnh của đồ thị dùng thuật toán duyệt như thế nào?

6. Mảng status[v] có ý nghĩa là gì?

7. Hàm DFS_acyclic(Adj,s) dùng để làm gì?

8. Hàm Acyclic(V, Adj) kiểm tra gì?

9. Viết Hàm Buildgraph(fname) để dựng đồ thị.

10. Viết chương trình in ra thông báo hệ thống chuyên đề học tập hợp lý hay không hợp lý.

Bước 2: Thực hiện nhiệm vụ

HS: Nghiên cứu SCD, lần lượt trả lời các câu hỏi từ 1 đến 8. Viết chương trình thực hiện theo yêu cầu 9, 10 trên máy tính.

GV: Quan sát và trợ giúp các nhóm HS. Hướng dẫn viết hàm Buildgraph(fname).

Bước 3: Báo cáo, thảo luận

HS: Tạo tệp Data.inp, chạy chương trình quan sát kết quả và báo cáo GV. GV: Quan sát bài làm của những HS đã chạy được trên máy tính.

Bước 4: Kết luận, nhận định:

GV nhận xét bài làm của các nhóm, cho điểm cộng với các nhóm làm tốt. Cùng cố kiến thức cần ghi nhớ.

HS: Ghi nhớ chương trình vào vở.

+ Mỗi đỉnh của đồ thị là 1 chuyên đề học tập.

+ Mỗi cạnh của 1 đồ thị là quan hệ ràng buộc kiến thức (chuyên đề i, chuyên đề j).

+ Đồ thị là có hướng.

+ Kiểm tra đồ thị có chu trình không.

+ Mảng status[v]=0 nếu đỉnh v chưa duyệt, =1 nếu đỉnh v đang duyệt, =2 nếu đỉnh v đã duyệt xong.

+ Hàm DFS_acyclic(Adj,s) trả về giá trị True nếu kiểm tra vùng duyệt từ đỉnh s không có chu trình. Ngược lại giá trị False có chu trình.

+ Hàm Acyclic(V,Adj) kiểm tra toàn bộ đồ thị xem có 1 chu trình nào không, nếu có trả về True, ngược lại trả về False.

+ Hàm buildgraph(V, Adj) dựng đồ thị biểu diễn dưới dạng danh sách kề Adj.

3. Hoạt động luyện tập

a) Mục tiêu: HS thông hiểu các hàm đã xây dựng và chỉnh sửa được chương trình đáp ứng yêu cầu bài toán.

b) Nội dung: Làm bài tập 1, 2 phần luyện tập SCD trang 74.

1. Hàm kiểm tra chu trình của đồ thị trên còn đúng không nếu đồ thị ban đầu là vô hướng?

2. Viết lại hàm kiểm tra chu trình DFS_acyclic(Adj,s) trong chương trình trên nhưng sử dụng phương án không đệ quy của thuật toán DFS().

c) Sản phẩm: Bài làm của HS trong vở ghi hoặc trên máy tính, điện thoại.

Sản phẩm dự kiến:

1. Hàm kiểm tra chu trình của đồ thị không đúng nếu đồ thị ban đầu là vô hướng. Hàm kiểm tra trên chỉ áp dụng cho đồ thị có hướng. Với đồ thị vô hướng cần chỉ sửa để có thể áp dụng.

2. Hàm DFS_acyclic(Adj,s) có thể viết lại nếu sử dụng phương án không đệ quy như sau:

Khi sử dụng thuật toán DFS không đệ quy, mảng status sẽ có ý nghĩa mới như sau:

- status[v] = 0 nếu s chưa được duyệt.

- status[v] = 1 nếu s đã được duyệt và vẫn đang nằm trong ngăn xếp S của lệnh duyệt theo chiều sâu.

- status[v] = 2 nếu s đã được duyệt và đã ra khỏi ngăn xếp S.

```
def DFS_acyclic(Adj,s):
    S=Stack()
    push(S,s)
    while not isEmptyStack(S):
        v=top(S)
        if status[v]==0:
            status[v]=1
        else:
            pop(S)
            status[v]=2
        for u in Adj[v]:
            if status[u]==0:
                push(S,u)
            elif status[u]==1:
                return False
    return True
```

d) Tổ chức thực hiện: Giao về nhà làm bài, tiết sau gọi một số HS lên bảng chữa bài. GV gọi 1 HS đứng tại vị trí trả lời bài 1.

Gọi 2 HS lên bảng cùng làm bài 2 các HS còn lại mở vở GV kiểm tra bài làm của HS trong vở ghi hoặc bài trên máy tính, điện thoại đã chuẩn bị từ tiết trước.

Gọi một số HS nhận xét bài trên bảng.

GV: Nhận xét, chạy chương trình cho cả lớp quan sát kết quả chương trình, cho điểm những HS làm bài tốt.

HS: Làm bài trên máy tính, chạy chương trình, ghi nhớ vào vở nếu làm sai.

4. Hoạt động vận dụng

a) Mục tiêu: HS vận dụng kiến thức đã học về đồ thị, thuật toán duyệt chiều sâu chỉnh sửa, viết được chương trình kiểm tra đồ thị có chu trình hay không biết đồ thị biểu diễn đồ thị dưới dạng ma trận kề hoặc danh sách kề.

b) Nội dung: Làm bài tập 1, 2 SCD trang 74 phần vận dụng.

c) Sản phẩm: Bài làm của HS trong vở ghi.

Sản phẩm dự kiến:

1. Sửa chương trình bổ sung thêm thông báo nếu hệ thống chuyên đề không hợp lý thì in dãy các chuyên đề có mâu thuẫn về kiến thức.

from Stack import *

def buildgraph(fname): f=open(fname,"r") n=int(f.readline()) Adj=[] **for** i **in** range(n)

V=[int(i) **for** i **in** range(n)]

for line **in** f: i,j=map(int,line.split()) Adj[i].append(j)

f.close() **return** V,Adj

def DFS_acyclic(Adj,s): status[s]=1 push(S,s)

for v **in** Adj[s]:

if status[v]==1:

prev[v]=s

bổ sung thêm

danh sách prev[v]=s có nghĩa

trước đỉnh v là đỉnh s

return False **elif** status[v]==0:

prev[v]=s

if not DFS_acyclic(Adj,v): **return False**

status[s]=2 **return True**

def Acyclic(V,Adj):

for u **in** V:

if status[u]==0:

if not DFS_acyclic(Adj,u): **return False**

return True

Hàm in dãy các chuyên đề mâu thuẫn kiến thức

def inpath(t): s=t; q=[] q.append(t)

while prev[t]!=s:

t=prev[t]

lần ngược mảng prev ta tìm được đường đi từ sôt

q.append(t) # lưu thứ tự các đỉnh trong 1 chu trình vào q

u=q[-1]

while q!=[]:

v=q.pop() print(v,"-->",sep="",end="")

print(u)

chương trình chính fi="data.inp" V,Adj=buildgraph(fi) status=[0]*len(V) prev=[None]*len(V)

S=Stack()

if Acyclic(V,Adj):

print("Hệ thống chuyên đề học tập hợp lý")

else:

print("Hệ thống chuyên đề học tập không hợp lý") t=pop(S)

print("dãy các chuyên đề có mâu thuẫn:") inpath(t)

2. Chương trình kiểm tra đồ thị có hướng $G = (V, Adj)$ có chu trình hay không, nếu có thì in ra chu trình. Đồ thị đề bài cho biểu diễn bởi danh sách cạnh.

from Stack import *

def buildGraph(fname):

""" Đồ thị có hướng nhập theo danh sách các cạnh """ f = open(fname)

n = int(f.readline())

```
V = [i for i in range(n)] Adj = [[] for i in range(n)] for line in f:
    i,j=map(int,line.split()) Adj[i].append(j)
f.close() return V,Adj
```

```
def DFS_cycle(s): status[s] = 1 push(S,s)
for v in Adj[s]:
    if status[v] == 0: prev[v] = s
    if DFS_cycle(v): return True
    elif status[v] == 1: prev[v] = s return True
status[s] = 2 return False
```

```
def isCyclic():
for v in V:
    if status[v] == 0:
    if DFS_cycle(v): return True
return False
```

```
def printpath(t):
s = t; q = []; q.append(t)
while prev[t] != s: t = prev[t] q.append(t)
u = q[-1]
while q != []:
```

```
v = q.pop() print(v,"-->",end = "")
print(u)
```

```
# Chương trình chính fi = "vd2tr74.inp" V,Adj = buildGraph(fi)
n = len(V); status = [0]*n;prev = [None]*n S = Stack()
if isCyclic():
print("Đồ thị có chu trình:") t = pop(S)
printpath(t)
else:
print("Đồ thị không có chu trình.")
```

d) Tổ chức thực hiện: HS làm ở nhà từ tiết trước, tiết sau GV gọi một số HS lên bảng chữa bài.

GV gọi 2 HS lên bảng làm bài, các HS còn lại mở vở GV kiểm tra bài làm của HS trong vở ghi hoặc trên máy tính, điện thoại.

Gọi một số HS nhận xét bài làm của các bạn trên bảng.

GV: Chạy chương trình của HS lên bảng chữa cho cả lớp quan sát kết quả chương trình.

Cho điểm những HS làm bài tốt, gọi một số HS chỉ ra lỗi nếu HS còn mắc phải, phân tích các câu lệnh trong chương trình giúp HS củng cố kiến thức.

HS: Quan sát và ghi nhớ vào vở nếu làm sai, làm bài trên máy tính, chạy chương trình.

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phần vận dụng.
- Đọc trước Bài 16. Kỹ thuật duyệt đồ thị theo chiều rộng.

BÀI 16. KỸ THUẬT DUYỆT ĐỒ THỊ THEO CHIỀU RỘNG

Thời gian thực hiện: 2 tiết

I. MỤC TIÊU

1. Kiến thức

- Khái niệm đường đi và khoảng cách (hay độ dài) của đường đi.
- Thuật toán duyệt đồ thị theo chiều rộng.

2. Năng lực

- Năng lực chung:
- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.
- Năng lực Tin học:
- Trình bày được ý tưởng của duyệt đồ thị theo chiều rộng.
- Mô phỏng được thuật toán duyệt theo chiều rộng.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành các bài tập theo yêu cầu của GV thông qua hệ thống câu hỏi, phiếu học tập, thông qua sản phẩm.

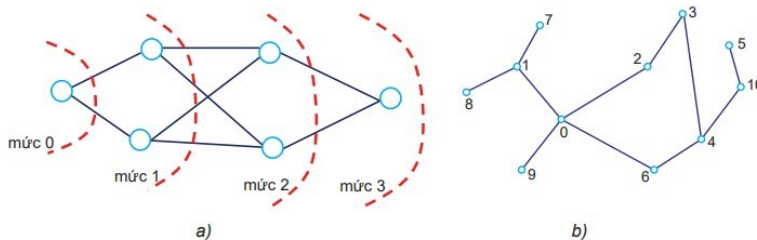
II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, máy tính, máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước bài 14. Kỹ thuật duyệt đồ thị theo chiều sâu.

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động 1: Khởi động

- a) Mục tiêu: HS làm quen với ý tưởng của thuật toán duyệt đồ thị theo chiều rộng và dự đoán cách thực hiện của thuật toán này.
- b) Nội dung: Quan sát đồ thị Hình 16.1 và cho biết việc duyệt các đỉnh của đồ thị sẽ được thực hiện như thế nào? Nếu xuất phát từ đỉnh số 0 sau khi duyệt theo chiều rộng ta được thứ tự các đỉnh duyệt trong hình 16.1b như thế nào?



Hình 16.1. Duyệt đồ thị lan toả theo các mức.

- c) Sản phẩm: Câu trả lời của HS.
- d) Tổ chức thực hiện: GV cho HS quan sát Hình 16.1 SCD trang 75 và trả lời câu hỏi. GV: Nhận xét câu trả lời của HS, đề biết thứ tự thăm khi xuất phát từ đỉnh 0 chúng ta cùng tìm hiểu ý tưởng duyệt đồ thị theo chiều rộng trong bài học hôm nay.

2. Hoạt động 2: Hình thành kiến thức

Hoạt động 2.1: Làm quen ý tưởng của thuật toán duyệt đồ thị theo chiều rộng

- a) Mục tiêu: HS biết và hiểu được ý tưởng của duyệt đồ thị theo chiều rộng (BFS) và thực hiện được việc duyệt đồ thị thủ công.
b) Nội dung: HS hoàn thành phiếu học tập số 1.

PHIẾU HỌC TẬP SỐ 1

1. HS nghiên cứu SCD trang 75 thiết lập bảng sau và điền các thông tin vào các cột.

Thực hiện duyệt chiều rộng cho đồ thị hình 16.1b. Bảng các đỉnh được duyệt theo chiều rộng, xuất phát từ đỉnh 0.

Mức	Các đỉnh được duyệt	Ghi chú
0	0	Mức 0 chỉ có đúng đỉnh xuất phát ban đầu 0
1		
2		
3		
4		
5		

2. Thứ tự các đỉnh được duyệt là?

3. Cho 2 đỉnh s và f. Nếu tồn tại đường đi từ s $\rightarrow$ f khoảng cách từ s $\rightarrow$ f là lớn nhất hay nhỏ nhất?

Củng cố kiến thức

1. Mệnh đề sau đúng hay sai? Giả sử gọi BFS(Adj,s) là chương trình duyệt đồ thị theo chiều rộng bắt đầu từ đỉnh s. Khi đó với mọi đỉnh v thuộc V, hàm BFS(Adj,s) sẽ duyệt qua đỉnh v khi và chỉ khi tồn tại đường đi từ s đến v.

2. Trả lời các câu hỏi dựa trên đồ thị Hình 16.2. a) Các đỉnh kề với a là đỉnh nào? b) Khoảng cách từ đỉnh a đến e là bao nhiêu? c) Nếu thực hiện duyệt đồ thị theo chiều rộng bắt đầu từ đỉnh a thì thứ tự các đỉnh được duyệt có thể như thế nào.

c) Sản phẩm: Hoàn thành phiếu học tập số 1.

Sản phẩm dự kiến:

Mức	Các đỉnh được duyệt	Ghi chú
0	0	Mức 0 chỉ có đúng đỉnh xuất phát ban đầu 0
1	1, 2, 6, 9	Các đỉnh kề với đỉnh 0
2	3, 4, 7, 8	Các đỉnh kề với đỉnh 2, 6, 1 và chưa thăm
3	10	Kề với đỉnh 4 và chưa thăm Đường đi từ 0: 0 $\rightarrow$ 4 $\rightarrow$ 10
4	5	Kề với đỉnh 10 và chưa thăm Đường đi từ 0: 0 $\rightarrow$ 6 $\rightarrow$ 4 $\rightarrow$ 10 $\rightarrow$ 5
5	Không có	Dừng tìm kiếm

Thứ tự duyệt là: 0 1 2 6 9 3 4 7 8 10 5.

Đường đi từ đỉnh 0 đến đỉnh 5 có 2 đường đi là: 0 $\rightarrow$ 6 $\rightarrow$ 4 $\rightarrow$ 10 $\rightarrow$ 5 và 0 $\rightarrow$ 2 $\rightarrow$ 3

→4 →10 →5.

Trong đó $0 \rightarrow 6 \rightarrow 4 \rightarrow 10 \rightarrow 5$ là đường đi ngắn nhất (qua ít cạnh nhất).

Thuật toán duyệt chiều rộng luôn cho đường đi ngắn nhất từ $s \rightarrow f$ nếu tồn tại đường đi.

Ghi nhớ: Với đỉnh s bất kì, ý tưởng của thuật toán duyệt đồ thị theo chiều rộng xuất phát từ đỉnh s là sẽ lần lượt duyệt các đỉnh theo từng mức hay theo khoảng cách từ đỉnh s đến đỉnh được duyệt.

Củng cố kiến thức

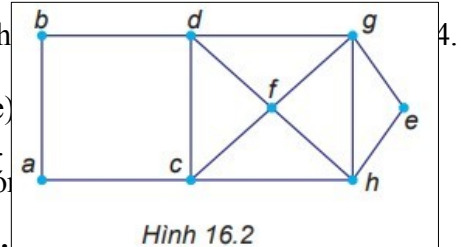
1. Mệnh đề này đúng. Cách chứng minh tương tự nh

2. a) Các đỉnh kề với đỉnh a là b, c .

b) Khoảng cách từ a đến e là 3 (đi qua các đỉnh c, h, e).

c) Thứ tự duyệt các đỉnh từ a sẽ là: a, b, c, d, f, h, e, g .

d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm học tập số 1.



Hình 16.2

Hoạt động của GV và HS Bước 1: Chuyển giao nhi

GV: Phát phiếu học tập số 1 cho các nhóm nghiên cứu SCD và hoàn thành phiếu học tập số 1.

Bước 2: Thực hiện nhiệm vụ

HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt hoàn thành phiếu học tập số 1. GV: Quan sát và trợ giúp các nhóm HS.

Bước 3: Báo cáo, thảo luận

HS: Hoàn thành phiếu học tập số 1.

GV: Chiều sản phẩm cho cả lớp quan sát so sánh kết quả các nhóm, kiểm tra vở ghi của một số HS.

Gọi một số HS đại diện các nhóm đứng tại vị trí trả lời câu hỏi củng cố kiến thức 1, 2 SCD trang 76.

Bước 4: Kết luận, nhận định

GV nhận xét bài làm của các nhóm. Củng cố kiến thức cần ghi nhớ. HS: Ghi nhớ kiến thức ý tưởng duyệt theo chiều rộng.

GV: Đánh giá cho điểm một số HS làm đúng câu hỏi củng cố 2.

Chia sẻ bài làm của HS lên máy chiếu cho cả lớp cùng quan sát, những HS làm chưa đúng rút kinh nghiệm.

Hoạt động 2.2: Tìm hiểu thuật toán duyệt theo chiều rộng (BFS)

a) Mục tiêu: HS biết và thực hiện được thuật toán, chương trình duyệt đồ thị theo chiều rộng sử dụng hàng đợi.

b) Nội dung: Quan sát và thảo luận tìm hiểu thuật toán duyệt chiều rộng trên đồ thị vô hướng trả lời các câu hỏi trong phiếu học tập số 2.

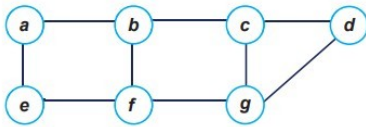
PHIẾU HỌC TẬP SỐ 2

STT	Queue	dequeue	Mark	Trạng thái
Mảng mark[y] dùng để làm gì.	0	0	0	Mark[0]=True
Viết hàm BFS(Adj,s) và giải thích ý nghĩa các dòng lệnh trong hàm.	1	1	1	
Điền vào chỗ trống trong 2 bảng sau khi thực hiện duyệt theo chiều rộng.				

3			...		
Thứ tự các đỉnh đã thăm như thế nào?					
4			...		
Đề duyệt toàn bộ các đỉnh của đồ thị (V, Adj) dùng hàm nào?					
241					
Hàm buildgraph có sẵn không? Xây dựng hàm đó như thế nào?					
Các hàm Queue, enqueue và dequeue có ý nghĩa gì? Lấy các hàm này ở đâu?					

Củng cố kiến thức

1. Thứ tự các đỉnh có trong danh sách đỉnh kề Adj có ảnh hưởng đến thứ tự các đỉnh được đánh dấu trong thuật toán duyệt theo chiều rộng hay không?
2. Cho đồ thị Hình 16.4. Nếu thực hiện duyệt theo chiều sâu và chiều rộng bắt đầu từ đỉnh a thì thứ tự các đỉnh được duyệt sẽ như thế nào?



Hình 16.4

- c) Sản phẩm: Viết được hàm duyệt chiều rộng cho đồ thị vô hướng.

Sản phẩm dự kiến:

Dùng mảng $\text{mark}[v] = \text{True}$ để đánh dấu các đỉnh đã duyệt, ban đầu thiết lập $\text{mark}[v] = \text{False}$ với mọi đỉnh $v \in V$.

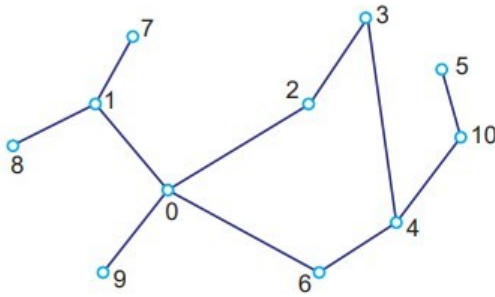
Hàm chính duyệt theo chiều rộng được mô tả như sau:

```

1      def BFS(Adj,s):
2          Q=Queue()
3          enqueue(Q,s)      # đẩy đỉnh s vào hàng đợi Q
4          while not isEmptyQueue(Q):
5              v=dequeue(Q)    # lấy đỉnh v từ hàng đợi ra
6              if not mark[v]:
7                  mark[v]=True # Đánh dấu đỉnh v đã thăm
8                  print(v, end=" ")      # In ra thứ tự các đỉnh được duyệt
9                  for u in Adj[v]:
10                     enqueue(Q,u)      # đưa tất cả các đỉnh là kề của v vào hàng đợi

```

Hàm được thiết lập để duyệt bắt đầu từ đỉnh s chưa được đánh dấu. Khi bắt đầu, hàng đợi Q được thiết lập và s được đưa vào Q tại dòng 2 và 3. Thao tác duyệt theo chiều rộng được thực hiện tại vòng lặp 4. Vòng lặp này sẽ thực hiện cho đến khi Q rỗng. Lần lượt lấy v ra khỏi Q, nếu v chưa được đánh dấu thì đánh dấu v và đưa các đỉnh kề của v vào hàng đợi Q.



Hình 16.3. Đồ thị và dữ liệu danh sách kề

graph.inp

```
11
0 1 2 6 9
1 0 7 8
2 0 3
3 2 4
4 3 6 10
5 10
6 0 4
7 1
8 1
9 0
10 4 5
```

Hàm thực hiện thuật toán duyệt theo chiều rộng trên toàn bộ đồ thị G. Nếu bộ dữ liệu đầu vào của đồ thị là (V, Adj) thì hàm duyệt sẽ có dạng BFS_Traversal(V, Adj).

```
def BFS_traversal(V, Adj):
    for s in V:
        if not mark[s]:
            BFS(Adj, s)
```

Đoạn chương trình sau mô tả phần thiết lập thông tin ban đầu của đồ thị, sau đó thực hiện thuật toán duyệt theo chiều rộng trên toàn bộ đồ thị G:

```
fname="graph.inp"
V, Adj=buildgraph(fname)
mark=[False]*len(V)
BFS_traversal(V, Adj)
```

Chú ý hàm buildgraph(fname) dựng đồ thị dưới dạng ma trận kề, dữ liệu nhập vào biểu diễn dưới dạng ma trận kề. Các hàm enqueue và dequeue được gọi trong thư viện Queue đã học trong Bài 4. Kiểu dữ liệu hàng đợi.

```
from Queue import *
def buildgraph(fname):
    f=open(fname, "r")
    n=int(f.readline())
    V=[int(i) for i in range(n)]
    Adj=[[ ] for i in range(n)]
    for line in f:
        c=[int(x) for x in line.split()]
        for k in range(1, len(c)):
            Adj[c[0]].append(c[k])
    return V, Adj
```

Nhắc lại các hàm trong Queue đã xây dựng:

- + Queue để khởi tạo hàng đợi rỗng.
- + enqueue(Q, s) dùng để thêm s vào cuối hàng đợi.
- + dequeue(Q) dùng để lấy ra và trả về phần tử ở đầu của hàng đợi Q.

```

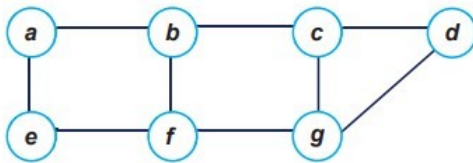
def Queue():
    return []
def enqueue(Q, x):
    Q.append(x)
def isEmptyQueue(Q):
    return len(Q)==0
def dequeue(Q):
    if isEmptyQueue(Q):
        raise ValueError('Hàng đợi rỗng không thể thực hiện được')
    else: return Q.pop(0)
def front(Q):
    if isEmptyQueue(Q):
        raise ValueError('Hàng đợi rỗng không thể thực hiện được')
    else: return Q[0]

```

Ghi nhớ: Thuật toán duyệt đồ thị theo chiều rộng bắt đầu tại đỉnh s sẽ lần lượt duyệt các đỉnh có đường đi từ s với số cạnh lần lượt là 0, 1, 2,... Thuật toán BFS có cấu trúc điều khiển tương tự như thuật toán duyệt không đệ quy theo chiều sâu DFS, trong đó sử dụng hàng đợi thay thế cho ngăn xếp.

Củng cố kiến thức

1. Thứ tự các đỉnh có trong danh sách đỉnh kề Adj có ảnh hưởng đến thứ tự các đỉnh được đánh dấu trong thuật toán duyệt theo chiều rộng.
2. – Duyệt theo chiều sâu bắt đầu từ a thì thứ tự duyệt như sau: a, b, c, d, g, f, e.
– Duyệt theo chiều rộng bắt đầu từ a thì thứ tự duyệt các đỉnh sẽ như sau: a, b, e, c, f, d, g.



Hình 16.4

- d) Tổ chức thực hiện: Chia lớp thành 6 nhóm, các nhóm thảo luận và trả lời các câu hỏi.

Hoạt động của GV và HS

Bước 1: Chuyển giao nhiệm vụ

GV: Yêu cầu các nhóm nghiên cứu SCD trang 77, 78 hoàn thành phiếu học tập số 2.

Bước 2: Thực hiện nhiệm vụ

HS: Thảo luận theo nhóm, nghiên cứu SCD lần lượt trả lời các câu hỏi trong phiếu học tập số 2.

GV: Quan sát và trợ giúp HS.

Bước 3: Báo cáo, thảo luận

GV: Gọi lần lượt một số HS trả lời các câu hỏi trên.

Bước 4: Kết luận, nhận định

GV nhận xét các câu trả lời, Viết chương trình hoàn chỉnh duyệt theo chiều rộng BFS với đồ thị

HS: Ghi nhớ thuật toán duyệt theo chiều rộng và chương trình cài đặt duyệt theo chiều rộng.
 GV: Chú ý khi gọi hàm duyệt theo chiều rộng sẽ sử dụng ngăn xếp để đưa các đỉnh chưa thăm vào hàng đợi, đỉnh nào lấy ra sẽ đánh dấu đã thăm, nên khai thác các hàm trong thư viện Queue đã xây dựng trong bài 4

3. Hoạt động luyện tập

- a) Mục tiêu: HS viết được hàm dựng được đồ thị biểu diễn được đồ thị dưới dạng ma trận kề, danh sách kề. Viết được chương trình trong đó hàm BFS sử dụng ma trận kề, danh sách kề.
- b) Nội dung: Làm bài tập 1, 2 phần luyện tập SCD trang 79.
- Viết lại hàm BFS() in ra các đỉnh đã duyệt. Áp dụng vào đồ thị trong bài để kiểm tra thứ tự các đỉnh đã duyệt có đúng như phần mô phỏng thủ công các hoạt động trên không.
 - Viết lại hàm BFS() duyệt theo chiều rộng nhưng sử dụng dữ liệu là ma trận kề A của đồ thị.
- c) Sản phẩm: Bài làm của HS trong vở ghi hoặc trên máy tính, điện thoại.

Sản phẩm dự kiến:

Bài luyện tập 1, SCD trang 79.

```
from Queue import * fname="graph.inp"
def buildGraph(fname):
    """ Thiết lập đồ thị theo dữ liệu danh sách đỉnh kề """ f = open(fname)
    n = int(f.readline())
    V = [i for i in range(n)] Adj = [[] for i in range(n)] for i in range(n):
    line = [int(x) for x in f.readline().split()] Adj[i].extend(line[1:])
    f.close() return V,Adj
```

```
def BFS(s):
    stt = 0
    Q = Queue() enqueue(Q,s)
    while not isEmptyQueue(Q): stt = stt + 1
    print(stt,".", " "*(3-len(str(stt))),sep = "",end = "") st = str(Q)
    print(st + " "*(30 - len(st)),end = "") v = dequeue(Q)
    print(v, " "*(10-len(str(v))), end = "") if not mark[v]:
    mark[v] = True print(v,end = " ") for u in Adj[v]:
    enqueue(Q,u)
```

```
else:
    print("--",end = "") print()

    # Chương trình chính
    V,Adj = buildGraph(fname)
    n = len(V)
    mark = [False]*n
    print("Stt Hàng đợi                               dequeue           Đánh dấu")
    print("-----") BFS(0)
```

Bài luyện tập 2 SCD trang 79.

```
def BFS(s):
```

```

Q=Queue() enqueue(Q,s)
while not isEmptyQueue(Q): v=dequeue(Q)
if not mark[v]: mark[v]=True print(v, end=" ")
for u in range(len(A)):
if A[u][v]==1: enqueue(Q,u)
Chú ý đồ thị khi dựng phải biểu diễn dưới dạng ma trận kề Chương trình đầy đủ in ra thứ tự các
đỉnh duyệt
from Queue import *
def buildgraph(fname): f=open(fname,"r") n=int(f.readline()) V=[int(i) for i in range(n)] A=[]
for i in range(len(V)):
c=[int(x) for x in f.readline().split()] A.append(c)
return V,A
def BFS(s):
Q=Queue() enqueue(Q,s)
while not isEmptyQueue(Q): v=dequeue(Q)
if not mark[v]: mark[v]=True print(v, end=" ")
for u in range(len(A)):
if A[u][v]==1: enqueue(Q,u)

fname="graph.inp" V,A=buildgraph(fname)

for u in range(len(V)): print(A[u])
mark=[False]*(len(V)+2) BFS(0)

```

d) Tổ chức thực hiện: Giao về nhà làm bài, tiết sau gọi lên bảng chữa bài.
GV gọi 2 HS lên bảng cùng làm bài 1, 2 HS làm bài 2 các HS còn lại mở vở, máy tính,
điện thoại GV kiểm tra chuẩn bị bài làm của HS ở nhà.
Gọi một số HS nhận xét bài trên bảng.
GV: Nhận xét, chạy chương trình cho cả lớp quan sát kết quả, cho điểm những HS làm
bài tốt.

4. Hoạt động vận dụng

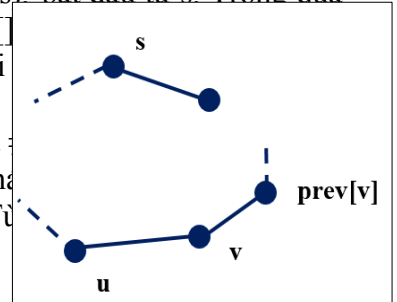
- Mục tiêu: HS vận dụng kiến thức đã học về đồ thị viết được chương trình kiểm tra xem đồ thị G có chu trình hay không, Kiểm tra xem có tồn tại đường đi từ s đến t hay không nếu có chỉ ra hành trình đi qua các đỉnh từ s đến t .
- Nội dung: Làm bài tập 1, 2 SCD trang 79 phần vận dụng.
 - Cho đơn đồ thị vô hướng $G = (V, E)$. Sử dụng thuật toán duyệt theo chiều rộng BFS, viết chương trình kiểm tra xem G có chu trình hay không. Chu trình (cycle) ở đây được hiểu là một đường đi khép kín, đỉnh xuất phát trùng với đỉnh kết thúc. Cần thiết lập hàm dạng $Acycle(G)$, hàm trả lại True nếu G không có chu trình, ngược lại hàm trả lại False.
 - Cho đơn đồ thị $G = (V, E)$ vô hướng hoặc có hướng. Cho trước hai đỉnh bất kì s và t . Viết chương trình kiểm tra xem có tồn tại đường đi từ s đến t hay không. Nếu có thì chương trình cần chỉ ra dãy các đỉnh tương ứng trên đường đi từ s đến t , nói cách khác chương trình cần chỉ ra một dãy các đỉnh $v_0, v_1, \dots, v_k$ sao cho: (v_{j-1}, v_j) là cạnh của đồ thị với $j = 1, 2, \dots, k$; $s = v_0, t = v_k$.
- Sản phẩm: Bài làm của HS trong máy tính, điện thoại hoặc vở ghi.

Sản phẩm dự kiến:

Bài 1 Vận dụng trang 79: Để kiểm tra đồ thị có chu trình không dùng 2 mảng `mark[]` và `prev[]` có ý nghĩa như sau:

- mark[] dùng để đánh dấu các đỉnh đã duyệt, mark[v] = True khi và chỉ khi đỉnh v đã được duyệt.
- prev[] dùng để đánh dấu đỉnh phía trước trong quá trình duyệt. Ví dụ nếu chúng ta đi theo một cạnh từ đỉnh v đến u (u bắt buộc là đỉnh kề của v), khi đó đặt prev[u] = v với ý nghĩa trong đường đi hiện thời, v đứng trước u.

Ý tưởng của thuật toán phát hiện chu trình như sau: BFS(s) bắt đầu từ s. Trong quá trình duyệt, các đỉnh được duyệt sẽ được đánh dấu mark[] duyệt thì prev[v] chỉ ra đỉnh đứng trước v trên đường đi trong quá trình duyệt, ví dụ từ đỉnh v, gặp đỉnh kề u mà mark[u] = True, tức là đã có 1 đường đi từ s đến u. Nếu u = prev[v] (xem hình bên) thì suy ra đã có một đường đi khác đường đi từ s đến u qua v, chúng ta thu được chu trình. Từ đây suy ra thuật toán phát hiện chu trình theo BFS.



```
from Queue import *
def buildgraph(fname): f=open(fname,"r") n=int(f.readline()) V=[int(i) for i in range(n)]
Adj=[] for i in range(n) for line in f:
c=[int(x) for x in line.split()]
for k in range(1,len(c)): Adj[c[0]].append(c[k])
return V,Adj
def BFS_cycle(s): Q=Queue() enqueue(Q,s) mark[s]=True
while not isEmptyQueue(Q): v=dequeue(Q)
for u in Adj[v]:
if not mark[u]: mark[u]=True prev[u]=v enqueue(Q,u)
elif prev[v]!=u: return True
return False def isCycle():
for v in V:
if not mark[v]:
if BFS_cycle(v): return True
return False
```

```
fname="graph.inp" V,Adj=buildgraph(fname) n=len(V) mark=[False]*n prev=[None]*n
if isCycle():
print("Đồ thị có chu trình")
else:
print("Đồ thị không có chu trình")
```

Bài 2. Vận dụng SCD trang 79

Xây dựng hàm xử lý chính là FindPath(s,t), hàm sẽ kiểm tra xem có đường đi từ đỉnh s đến đỉnh t hay không, nếu có sẽ thông báo đường đi từ s đến t, ngược lại thông báo không tồn tại đường đi. Để tồn tại đường đi từ s → t thì pre[t] != -1.

```
from Queue import *
def buildgraph(fname):
""" Đồ thị đã cho biểu diễn danh sách cạnh """
```

```
f=open(fname,"r") n=int(f.readline()) V=[i for i in range(n)]
Adj=[] for i in range(n)
for line in f: i,j=map(int,line.split()) Adj[i].append(j)
Adj[j].append(i) f.close()
return V,Adj
```

```

def BFS(s):
    Q=Queue() mark[s]=True enqueue(Q,s)
    while not isEmptyQueue(Q): v=dequeue(Q)
    for u in Adj[v]:
        if not mark[u]: mark[u]=True prev[u]=v enqueue(Q,u)
    def FindPath(s,t): BFS(s)
    if prev[t]==-1:
        print('Không tồn tại đường đi từ đỉnh',s,'đến đỉnh ',t)
    else:
        q=[]
        q.append(t)
        while prev[t]!=-1: t=prev[t] q.append(t)
        while q!=[]:
            v=q.pop()
            if len(q)>0:
                print(v,"-->",end="")
            else:
                print(v)
        # chương trình chính fname="vd2tr79.inp" V,Adj=buildgraph(fname) n=len(V) mark=[False]*n
        prev=[-1]*n
        s,t=map(int,input('nhập vào 2 đỉnh s,t:').split()) FindPath(s,t)

```

d) Tổ chức thực hiện: HS làm ở nhà, tiết sau GV gọi 4 HS lên bảng chữa bài, trong đó hai HS làm bài 1, hai học sinh làm bài 2 trên máy tính, chiếu chương trình cho cả lớp quan sát nhận xét.

GV: Gọi một số HS nhận xét, đánh giá, chạy chương trình cho HS quan sát kết quả gọi một số HS giải thích ý nghĩa các câu lệnh, các hàm để HS hiểu chương trình.

HS: Ghi nhớ vào vở nếu làm sai.

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phân vận dụng.
- Đọc trước bài 17. Thực hành duyệt đồ thị tổng hợp.

BÀI 17. THỰC HÀNH DUYỆT ĐỒ THỊ TỔNG HỢP

Thời gian thực hiện: 2 tiết thực hành

I. MỤC TIÊU

1. Kiến thức

- Thuật toán duyệt đồ thị theo chiều rộng.
- Một số ứng dụng thực tế của duyệt đồ thị theo chiều rộng.

2. Năng lực

- Năng lực chung:
- Tự chủ và tự học: Chủ động học tập, tìm hiểu nội dung bài học.
- Giải quyết vấn đề và sáng tạo: Trả lời được các câu hỏi, giải quyết được các vấn đề với sự hỗ trợ của công nghệ thông tin và truyền thông.
- Giao tiếp và hợp tác: Biết lựa chọn hình thức làm việc nhóm với quy mô phù hợp với yêu cầu và thực hiện tốt nhiệm vụ.
- Năng lực Tin học:

- Thực hành duyệt đồ thị theo chiều rộng.
- Thực hành ứng dụng kĩ thuật duyệt đồ thị trong một số bài toán thực tế.

3. Phẩm chất

- Chăm chỉ: Tích cực tìm tòi và sáng tạo trong học tập.
- Trách nhiệm: Hoàn thành nhiệm vụ, các bài tập theo yêu cầu của GV thông qua sản phẩm.

II. THIẾT BỊ DẠY HỌC VÀ HỌC LIỆU

- GV: Sách chuyên đề, SGK, tài liệu tham khảo, máy tính, máy chiếu.
- HS: Sách chuyên đề, vở ghi, máy tính, đọc trước Bài 17. Thực hành duyệt đồ thị tổng hợp.

III. TIẾN TRÌNH DẠY HỌC

1. Hoạt động 1: Khởi động

a) Mục tiêu: HS thấy được các ứng dụng của thuật toán duyệt đồ thị theo chiều rộng.

b) Nội dung: HS hãy trao đổi, thảo luận và trả lời câu hỏi sau:

Thuật toán duyệt theo chiều rộng được ứng dụng giải quyết các bài toán nào?

c) Sản phẩm: Câu trả lời của HS.

d) Tổ chức thực hiện: HS thảo luận và trả lời.

GV: Nhận xét câu trả lời của HS, củng cố ứng dụng của thuật toán duyệt đồ thị theo chiều rộng vào các bài toán: Tìm đường đi ngắn nhất.

2. Hoạt động 2: Hình thành kiến thức

Nhiệm vụ: Tìm đường đi xe đạp

a) Mục tiêu: HS tìm hiểu, trao đổi để hiểu bài toán cần giải quyết.

b) Nội dung: Viết chương trình giải bài toán sau:

Các bạn HS lớp em (được đánh số từ 0 đến $n - 1$) có nhà ở trải rộng khắp thành phố.

Trong thành phố có những đường đi chỉ dành cho xe cơ giới, nhưng cũng có đường đi dành cho xe đạp. Các bạn HS lớp em chỉ biết đi xe đạp. Dữ liệu đầu vào gồm hai tệp.

Tệp Danh-sach.inp sẽ lưu tên các bạn trong lớp, tên mỗi bạn ghi trên một dòng. Tệp thứ hai, xe-dap.inp mô tả các con đường có thể đi xe đạp từ nhà một bạn trong lớp đến nhà bạn khác. Tệp này cũng có nhiều dòng, mỗi dòng là hai số tự nhiên i, j cách nhau bởi dấu cách, chỉ ra từ nhà bạn thứ i có thể đi xe đạp được đến nhà bạn j . Các đường đi xe đạp này là hai chiều.

Tệp danh sách lớp học Danh-sach.inp bắt đầu là số tự nhiên n , n dòng tiếp theo mỗi dòng là tên của các bạn HS trong lớp. Tệp lưu thông tin các đường đi xe đạp xe-dap.inp có dòng đầu tiên là số tự nhiên n , các dòng tiếp theo, mỗi dòng chỉ một đường đi bằng xe đạp giữa hai bạn HS trong lớp được mô tả bằng hai số tự nhiên cách nhau bởi dấu cách.

Danh-sach.inp	Xe-dap.inp
6	6
Hòa	0 1
Vinh	1 3
Lê	2 3
Quân	4 5
Quang	
Vân	

Yêu cầu của bài toán sau khi nhập dữ liệu như sau:

- Nhập từ bàn phím hai số tự nhiên i, j ($0 \leq i < j \leq n - 1$), hai số này sẽ tương ứng với hai HS trong lớp.
- Thông báo có cách đi xe đạp từ nhà bạn thứ i đến nhà bạn thứ j hay không, nếu có thì ghi ra dãy các đường đi lần lượt từ nhà bạn thứ i , đi qua một số nhà bạn khác, cuối cùng đến được nhà bạn thứ j . Trong ví dụ trên, nếu nhập $i = 0, j = 2$ thì kết quả thể hiện đường đi xe đạp như sau:

Hòa → Vinh → Quân → Lê

Còn nếu nhập hai chỉ số $i = 0, j = 5$ thì kết quả sẽ thông báo như sau: Không tồn tại đường đi xe đạp từ nhà bạn Hòa đến nhà bạn Vân.

c) Sản phẩm: Chương trình HS viết trên máy tính.

Sản phẩm dự kiến:

```
from Queue import * fname="danhsach.inp" fxe="xedap.inp"
def buildgraph(fxe): f=open(fxe,"r") n=int(f.readline()) V=[i for i in range(n)]
Adj=[[] for i in range(n)]
for line in f: i,j=map(int,line.split()) Adj[i].append(j)
Adj[j].append(i) f.close()
return V,Adj
def BFS(Adj,s): Q=Queue() mark[s]=True enqueue(Q,s)
while not isEmptyQueue(Q): v=dequeue(Q)
for u in Adj[v]:
if not mark[u]: mark[u]=True prev[u]=v enqueue(Q,u)

def printpath(s,t):
if t==s:
print(names[s],end=" ")
else:
if prev[t]==None:
print("không tồn tại đường đi")
else:
printpath(s,prev[t]) print("-->",names[t],end=" ")
def getnames(fname): f=open(fname,"r",encoding="UTF-8") n=int(f.readline())
names=[]
for i in range(n): names.append(f.readline().strip())
f.close() return names
```

```
# chương trình chính names=getnames(fname) V,Adj=buildgraph(fxe) n=len(V)
mark=[False]*n # khởi tạo mảng False
prev=[None]*n # khởi tạo mảng prev s,t=map(int,input("Nhập số thứ tự của hai
HS").split()) BFS(Adj,s)
printpath(s,t)
```

d) Tổ chức thực hiện: 2 HS một nhóm, làm bài trên máy tính.

Hoạt động của GV và HS Bước 1: Chuyển giao nhiệm vụ

GV: Yêu cầu tất cả HS đọc và thực hiện nhiệm vụ SCD trang 80, nêu câu hỏi cho HS.

1. Mỗi tên HS là 1 đỉnh hay 1 cạnh của đồ thị?
2. Mỗi đường đi bằng xe đạp giữa hai bạn trong lớp có là 1 cạnh trong đồ thị không?
3. Đồ thị là có hướng hay vô hướng?

4. Để kiểm tra xem có đi xe đạp được từ nhà bạn thứ i đến nhà bạn thứ j không phải làm thế nào?
5. Để duyệt các đỉnh của đồ thị dùng thuật toán duyệt như thế nào?
6. Mảng mark[v], prev[v] có ý nghĩa là gì?
7. Hàm BFS(Adj,s) dùng để làm gì?
8. Hàm printpath(s,t) để làm gì?
9. Hàm getnames(fname) để làm gì? Lệnh strip() có tác dụng gì?
9. Viết hàm buildgraph(fname) để dựng đồ thị
10. Viết chương trình in ra thông báo đường đi hoặc thông báo không tồn tại đường đi.

Bước 2: Thực hiện nhiệm vụ

HS: Nghiên cứu SCD, lần lượt trả lời các câu hỏi từ 1-> 8. Viết chương trình thực hiện theo yêu cầu 9, 10 trên máy tính.

GV: Quan sát và trợ giúp các nhóm HS.

Bước 3: Báo cáo, thảo luận

HS: Tạo tệp Danhsach.inp, xedap.inp chạy chương trình quan sát kết quả và báo cáo GV.

GV: Quan sát bài làm của những HS đã chạy được trên máy tính.

Bước 4: Kết luận, nhận định

GV nhận xét bài làm của các nhóm, cho điểm cộng với các nhóm làm tốt. Cùng cố kiến thức cần ghi nhớ.

HS: Ghi nhớ chương trình vào vở.

+ Mỗi đỉnh của đồ thị là 1 HS.

+ Mỗi cạnh của 1 đồ thị là một đường đi bằng xe đạp giữa hai bạn trong lớp.

+ Đồ thị là vô hướng.

+ Thuật toán duyệt theo chiều rộng.

+ Mảng mark[v]=False nếu đỉnh v chưa duyệt, =True nếu đỉnh v đã duyệt.

+ prev[v] =u có ý nghĩa là lưu đỉnh trước của v là đỉnh u, nếu trước v không có đỉnh nào sẽ nhận giá trị None.

+ Hàm BFS(Adj,s) dùng để duyệt theo chiều rộng bắt đầu từ đỉnh s.

+ Hàm printpath(s,t) để in ra đường đi từ đỉnh s tới đỉnh t trong đó có dùng mảng names[] để lưu tên HS.

+ Hàm getnames(fname) để đọc dữ liệu từ tệp danhsach.inp lưu tên HS vào mảng names.

+ Lệnh strip() để xóa các khoảng trắng ở đầu và sau tên

3. Hoạt động luyện tập

a) Mục tiêu: HS thông hiểu các hàm đã viết trong nhiệm vụ Tìm đường đi xe đạp chính sửa, bổ sung thêm vào chương trình đáp ứng yêu cầu bài toán.

b) Nội dung: Làm bài tập 1, 2 phần luyện tập SCD trang 82.

1. Sửa lại phần nhập dữ liệu hai HS: sẽ nhập trực tiếp tên hai HS, kiểm tra các tên này có nhập đúng không và thực hiện yêu cầu như trong chương trình trên.

2. Viết lại hàm BFS() trong chương trình trên nhưng sử dụng ma trận kề A thay thế cho danh sách kề Adj.

c) Sản phẩm: Bài làm của HS trong vở ghi hoặc trên máy tính, điện thoại.

Sản phẩm dự kiến:

Bài luyện tập 1 SCD trang 82.

Viết hàm nhapdl để nhập vào tên của từng HS, sau đó kiểm tra xem tên nhập vào có trong danh sách không dùng lệnh `d1=names.count(hs1)` nếu `hs1` có mặt trong `names` sẽ

cho kết quả >0, nếu không có mặt sẽ cho kết quả =0. Nếu có mặt trong danh sách ta sẽ dùng lệnh `names.index(hs1)` để tìm vị trí xuất hiện trong `names` để gọi hàm BFS.

```
def nhapdl(names):
    hs1=input("Nhập tên HS thứ nhất:") hs2=input("Nhập tên HS thứ hai:") d1=names.count(hs1)
    d2=names.count(hs2)
    if d1==0 or d2==0: return -1, -1
    else:
        i=names.index(hs1) j=names.index(hs2) return i,j
```

code chương trình đầy đủ.

```
from Queue import * fname="danhsach.inp" fxe="xedap.inp"
def buildgraph(fxe): f=open(fxe,"r") n=int(f.readline()) V=[i for i in range(n)]
Adj=[[ ] for i in range(n)]

for line in f: i,j=map(int,line.split())
#print(i,j) Adj[i].append(j)
Adj[j].append(i) f.close()
return V,Adj
def BFS(Adj,s): Q=Queue() mark[s]=True enqueue(Q,s)
while not isEmptyQueue(Q): v=dequeue(Q)
for u in Adj[v]:
    if not mark[u]: mark[u]=True prev[u]=v enqueue(Q,u)
def printpath(s,t):
    if t==s:
        print(names[s],end=" ")
    else:
        if prev[t]==None:
            print("Không tồn tại đường đi")
        else:
            printpath(s,prev[t]) print("-->",names[t],end=" ")
def getnames(fname): f=open(fname,"r",encoding="UTF-8") n=int(f.readline())
    names=[ ]
    for i in range(n): names.append(f.readline().strip())
    f.close() return names
def nhapdl(names):
    hs1=input("Nhập tên HS thứ nhất:") hs2=input("Nhập tên HS thứ hai:") d1=names.count(hs1)
    d2=names.count(hs2)
    if d1==0 or d2==0: return -1, -1
    else:
        i=names.index(hs1) j=names.index(hs2) return i,j
# chương trình chính names=getnames(fname) V,Adj=buildgraph(fxe) n=len(V)
```

```
mark=[False]*n prev=[None]*n s,t=nhapdl(names) if s!=-1 and t!=-1:
    BFS(Adj,s) printpath(s,t)
else:
    print("Tên nhập không có trong danh sách HS")
```

Để duyệt BFS sử dụng ma trận kề thì trong hàm dựng đồ thị buildgraph() phải biểu diễn đồ thị dưới dạng ma trận kề $A[u][v]=1$ nếu đỉnh u và v kề nhau, $A[u][v]=0$ nếu u và v không kề với nhau.

Hàm BFS() sử dụng ma trận kề.

```
def BFS(A,s):
    Q=Queue() mark[s]=True enqueue(Q,s)
    while not isEmptyQueue(Q): v=dequeue(Q)
    for u in range(len(A)):
        if A[u][v]==1:
            if not mark[u]: mark[u]=True prev[u]=v enqueue(Q,u)
```

Code chương trình đầy đủ

```
from Queue import * fname="danhsach.inp" fxe="xedap.inp"
def buildgraph(fxe): f=open(fxe,"r") n=int(f.readline()) V=[i for i in range(n)]
A=[[0 for i in range(n)] for j in range(n)]
for line in f: i,j=map(int,line.split()) A[i][j]=1
A[j][i]=1
f.close() return V,A
def BFS(A,s):
    Q=Queue() mark[s]=True enqueue(Q,s)
    while not isEmptyQueue(Q): v=dequeue(Q)
    for u in range(len(A)):
        if A[u][v]==1:
            if not mark[u]: mark[u]=True prev[u]=v enqueue(Q,u)

def printpath(s,t):
    if t==s:
        print(names[s],end=" ")
    else:
        if prev[t]==None:
            print("không tồn tại đường đi")
        else:
            printpath(s,prev[t]) print("-->",names[t],end=" ")
def getnames(fname): f=open(fname,"r",encoding="UTF-8") n=int(f.readline())
names=[]
for i in range(n): names.append(f.readline().strip())
f.close() return names
# chương trình chính names=getnames(fname) V,A=buildgraph(fxe) n=len(V) mark=[False]*n
prev=[None]*n
s,t=map(int,input("Nhập số thứ tự của hai HS").split()) BFS(A,s)
printpath(s,t)
```

d) Tổ chức thực hiện: Giao về nhà làm bài, tiết sau gọi một số HS lên bảng chữa bài. GV gọi 2 HS lên bảng làm bài 1.

Gọi 2 HS lên bảng cùng làm bài 2 các HS còn lại mở vở GV kiểm tra bài làm của HS trong vở ghi hoặc bài trên máy tính, điện thoại đã chuẩn bị từ tiết trước.

Gọi một số HS nhận xét bài trên bảng

GV: Nhận xét, chạy chương trình cho cả lớp quan sát kết quả chương trình, cho điểm những HS làm bài tốt.

HS: Làm bài trên máy tính, chạy chương trình, ghi nhớ vào vở nếu làm sai.

4. Hoạt động vận dụng

a) Mục tiêu: HS vận dụng kiến thức đã học về đồ thị, thuật toán duyệt chiều sâu chỉnh sửa, viết được chương trình kiểm tra đồ thị có chu trình hay không biết đồ thị biểu diễn đồ thị dưới dạng ma trận kề hoặc danh sách kề.

b) Nội dung: Làm bài tập 1, 2 SCD trang 82 phần vận dụng.

1. Bổ sung thêm yêu cầu của nhiệm vụ trên như sau: Có hay không hai bạn HS trong lớp mà không thể đi xe đạp từ nhà bạn này đến nhà bạn kia. Nếu có thì thông báo tên hai bạn HS đó.

2. Thiết lập hàm printpath(s,t) không đệ quy có tính năng tương tự hàm cùng tên trong chương trình trên.

c) Sản phẩm: Bài làm của HS trong vở ghi hoặc trong máy tính, điện thoại.

Sản phẩm dự kiến:

Bài vận dụng 1 SCD trang 82.

Hướng dẫn: Để kiểm tra xem có hai HS nào đó không thể đi xe đạp đến nhà nhau. Ta xây dựng hàm check() để kiểm tra. Với 2 HS nào đó trong lớp thực chất là ta phải duyệt tất cả các trường hợp nên $s \in [0, n-1]$ và $t \in [0, n-1]$ với mỗi lần duyệt phải khởi tạo lại mảng mark và prev, và gọi BFS(s).

Nếu $s! = t$ và $prev[t] = -1$ thì chứng tỏ s không đi đến t được. Khi đó dừng lại không cần duyệt tiếp nên dùng thêm biến đánh dấu $ok = \text{True}$, nếu không tìm thấy, $ok = \text{False}$ và dừng luôn.

```
def Check(): kq = True
for s in range(n):
    for t in range(n):
        mark[ : ] = [False]*n
        prev[ : ] = [-1]*n
        BFS(s)
        if s!=t and prev[t]==-1:
            print("Hai bạn",names[s],"và",names[t],"không thể đi đến nhà nhau")
            kq = False
            break
    if not kq:
        break
    if kq:
        print("Bất kì 2 bạn nào cũng đến được nhà nhau bằng xe đạp")
```

Code chương trình đầy đủ

```
from Queue import *
fname="danhsach.inp"
fxe="xedap.inp"
def buildgraph(fxe):
    f=open(fxe,"r")
    n=int(f.readline())
    V=[i for i in range(n)]
    Adj=[[] for i in range(n)]
    for line in f:
        i,j=map(int,line.split())
        #print(i,j)
        Adj[i].append(j)
        Adj[j].append(i)
    f.close()
    return V,Adj
```

```

def BFS(s):
    Q=Queue() mark[s]=True enqueue(Q,s)
    while not isEmptyQueue(Q): v=dequeue(Q)
    for u in Adj[v]:
    if not mark[u]: mark[u]=True prev[u]=v enqueue(Q,u)

def Check(): kq = True
for s in range(n):
for t in range(n):
mark[ : ] = [False]*n prev[ : ] = [-1]*n BFS(s)
if s!= t and prev[t] == -1:
print("Hai bạn",names[s],"và",names[t],"không thể đi đến
nhà nhau")

kq = False
break
if not kq:
break
if kq:
print("Bất kì 2 bạn nào cũng đến được nhà nhau bằng xe đạp")

def getnames(fname): f=open(fname,"r",encoding="UTF-8") n=int(f.readline())
names=[]
for i in range(n): names.append(f.readline().strip())
f.close() return names
# chương trình chính names=getnames(fname) V,Adj=buildgraph(fxe) n=len(V)
mark=[False]*n prev=[-1]*n
Check()

```

Bài vận dụng 2 SCD trang 82.

Hướng dẫn: Để xây dựng hàm printpath(s,t) viết dưới dạng không đệ quy ta có nhận xét sau:

- BFS(s) để xuất phát từ đỉnh s được thăm, khi tìm thấy đỉnh t trong duyệt BFS thì thăm t và dừng lại không cần duyệt tiếp.
- Nếu prev[t]==None chứng tỏ không tồn tại đường đi từ s →t.
- Nếu prev[t]!=None chứng tỏ tồn tại đường đi từ s →t.
- Để in ra hành trình đi từ s →t qua các đỉnh dựa vào mảng prev đã lưu lật ngược lại ta sẽ được một đường đi từ s →t dùng thêm mảng q để lưu các đỉnh tìm được.

Code chương trình đầy đủ

```

from Queue import * fname="danhsach.inp" fxe="xedap.inp"
def buildgraph(fxe): f=open(fxe,"r") n=int(f.readline()) V=[i for i in range(n)]
Adj=[[] for i in range(n)]
for line in f: i,j=map(int,line.split())
#print(i,j) Adj[i].append(j)
Adj[j].append(i) f.close()
return V,Adj
def BFS(s,t):
Q=Queue() mark[s]=True enqueue(Q,s)
while not isEmptyQueue(Q): v=dequeue(Q)
for u in Adj[v]:

```

```
if not mark[u]: mark[u]=True prev[u]=v enqueue(Q,u) if u==t:
return
```

```
def printpath(s,t):
if prev[t]==None:
print('Hai HS',names[s],'và',names[t],'không đến được nhà nhau')
else:
q=[]
q.append(t)
while prev[t]!=None: t=prev[t] q.append(t)
while q!=[]:
v=q.pop()
if len(q)>0:
print(names[v],"-->",end="")
else:
```

```
print(names[v])
def getnames(fname): f=open(fname,"r",encoding="UTF-8") n=int(f.readline())
names=[]
for i in range(n): names.append(f.readline().strip())
f.close() return names
# chương trình chính names=getnames(fname) V,Adj=buildgraph(fxe) n=len(V)
mark=[False]*n prev=[None]*n
s,t=map(int,input("Nhập số thứ tự của hai HS").split()) BFS(s,t)
printpath(s,t)
```

d) Tổ chức thực hiện: HS làm ở nhà từ tiết trước, tiết sau GV gọi một số HS lên bảng chữa bài.

GV gọi hai HS lên bảng làm bài, các HS còn lại mở vở GV kiểm tra bài làm của HS trong vở ghi hoặc trên máy tính, điện thoại.

Gọi một số HS nhận xét bài làm của các bạn trên bảng.

GV: Chạy chương trình của HS lên bảng chữa cho cả lớp quan sát kết quả chương trình. Cho điểm những HS làm bài tốt, gọi một số HS chỉ ra lỗi nếu HS còn mắc phải, phân tích các câu lệnh trong chương trình giúp HS củng cố kiến thức.

HS: Quan sát và ghi nhớ vào vở nếu làm sai, viết chương trình trên máy tính, chạy chương trình quan sát kết quả.

IV. HƯỚNG DẪN VỀ NHÀ

- Ghi nhớ kiến thức trong bài.
- Hoàn thành các bài tập phân vận dụng.
- Ôn tập lại toàn bộ kiến thức và các bài tập đã làm trong Chuyên đề 3.