BỘ GIÁO DỤC VÀ ĐÀO TẠO

CHUYÊN ĐỀ HỌC TẬP TIN HỌC 12

ĐỊNH HƯỚNG KHOA HỌC MÁY TÍNH

SÁCH GIÁO VIÊN



NHÀ XUẤT BẢN GIÁO DỤC VIỆT NAM

PHẠM THẾ LONG (Tổng Chủ biên)
BÙI VIỆT HÀ (Chủ biên)
NGUYỄN HOÀNG HÀ – LÊ HỮU TÔN

CHUYÊN ĐỀ HỌC TẬP
TIN HỌC 12

ĐỊNH HƯỚNG KHOA HỌC MÁY TÍNH

SÁCH GIÁO VIÊN

KẾT NỐI TRI THỨC
VỚI CUỘC SỐNG

NHÀ XUẤT BẢN GIÁO DỤC VIỆT NAM

QUY ƯỚC VIẾT TẮT DÙNG TRONG SÁCH

CNTT	Công nghệ thông tin
CNTT&TT	Công nghệ thông tin và Truyền thông
CT GDPT	Chương trình Giáo dục phổ thông
GV	Giáo viên
HS	Học sinh
SCĐ	Sách Chuyên đề học tập Tin học 12 – Định hướng Khoa học máy tính
SGV	Sách giáo viên Chuyên đề học tập Tin học 12 – Định hướng Khoa học máy tính
THPT	Trung học phổ thông

KẾT NỐI TRI THỨC
VỚI CUỘC SỐNG

LỜI NÓI ĐẦU

Sách giáo viên Chuyên đề học tập Tin học 12 – Định hướng Khoa học máy tính được biên soạn nhằm hỗ trợ các thầy cô triển khai kế hoạch dạy học *Chuyên đề học tập Tin học 12 – Định hướng Khoa học máy tính* thuộc bộ sách *Kết nối tri thức với cuộc sống* của Nhà xuất bản Giáo dục Việt Nam.

Ngoài phần Hướng dẫn chung về mục tiêu môn học, giới thiệu khái quát những nét chung về phương pháp dạy học và kiểm tra, đánh giá theo tinh thần của Chương trình Giáo dục phổ thông 2018, nội dung chính của tài liệu là các bài hướng dẫn dạy học *Chuyên đề học tập Tin học 12 – Định hướng Khoa học máy tính*. Các bài hướng dẫn được viết theo cấu trúc chuyên đề, bài học trong sách chuyên đề và gồm các nội dung thống nhất như sau:

A. Mục tiêu, yêu cầu đối với học sinh: Nêu ngắn gọn kiến thức, năng lực và phẩm chất mà HS cần đạt được để đáp ứng các yêu cầu cần đạt nêu trong Chương trình Giáo dục phổ thông 2018.

B. Yêu cầu chuẩn bị: Gợi ý một số nội dung cần chuẩn bị trước để triển khai bài dạy.

C. Gợi ý các hoạt động dạy học chính: Trình bày ngắn gọn một số gợi ý triển khai các hoạt động chính trong mỗi bài học để giáo viên tham khảo. Lưu ý, có thể có nhiều cách tiếp cận khác nhau cho từng hoạt động cụ thể, giáo viên có thể lựa chọn cách thức tổ chức và triển khai để đạt mục tiêu bài học. Sự hấp dẫn, tính tích cực của giờ học tuỳ thuộc vào sự sáng tạo của mỗi giáo viên.

D. Thông tin bổ sung: Trình bày một số lưu ý hoặc một số kiến thức bổ sung giúp giáo viên có thêm thông tin tham khảo để làm phong phú nội dung dạy học tuỳ theo hoàn cảnh và điều kiện cụ thể của lớp học.

Việc phân công tác giả biên soạn các nội dung *Sách giáo viên Chuyên đề học tập Tin học 12 – Định hướng Khoa học máy tính* cụ thể như sau:

– Bài 1, 2, 4, 6, 7, 9, 11 – 17: Bùi Việt Hà;

– Bài 3, 5: Lê Hữu Tôn;

– Bài 8, 10: Nguyễn Hoàng Hà.

Nội dung cuốn sách chắc chắn không tránh khỏi sơ suất, rất mong được các thầy cô giáo góp ý để bổ sung, hoàn thiện. Xin trân trọng cảm ơn!

CÁC TÁC GIẢ

MỤC LỤC

Trang

Lời nói đầu 3

PHẦN MỘT. HƯỚNG DẪN CHUNG 5

PHẦN HAI. HƯỚNG DẪN DẠY HỌC CÁC BÀI CỤ THỂ 15

Chuyên đề 1. Tìm hiểu một vài kiểu dữ liệu tuyến tính 16

Bài 1. Mô hình dữ liệu ngăn xếp và hàng đợi 16

Bài 2. Kiểu dữ liệu ngăn xếp 20

Bài 3. Thực hành kiểu dữ liệu ngăn xếp 25

Bài 4. Kiểu dữ liệu hàng đợi 30

Bài 5. Thực hành kiểu dữ liệu ngăn xếp và hàng đợi 36

Chuyên đề 2. Tìm hiểu cây tìm kiếm nhị phân trong sắp xếp và tìm kiếm 39

Bài 6. Cây nhị phân 39

Bài 7. Cây tìm kiếm nhị phân 46

Bài 8. Thực hành cây tìm kiếm nhị phân 56

Bài 9. Các thuật toán duyệt trên cây tìm kiếm nhị phân 65

Bài 10. Thực hành tổng hợp với cây tìm kiếm nhị phân 71

Chuyên đề 3. Tìm hiểu kĩ thuật duyệt đồ thị và ứng dụng 79

Bài 11. Khái niệm đồ thị 79

Bài 12. Biểu diễn đồ thị 84

Bài 13. Thực hành thiết lập đồ thị 88

Bài 14. Kĩ thuật duyệt đồ thị theo chiều sâu 92

Bài 15. Thực hành duyệt đồ thị theo chiều sâu 100

Bài 16. Kĩ thuật duyệt đồ thị theo chiều rộng 104

Bài 17. Thực hành duyệt đồ thị tổng hợp 112

PHẦN
MỘT HƯỚNG DẪN CHUNG

I. MỤC TIÊU MÔN HỌC

1. Mục tiêu chung

Chương trình môn Tin học góp phần hình thành, phát triển những phẩm chất chủ yếu và năng lực chung đã được xác định trong Chương trình tổng thể, đồng thời góp phần chủ yếu hình thành, phát triển năng lực tin học cho HS với các thành phần sau: sử dụng và quản lí các phương tiện công nghệ thông tin và truyền thông; ứng xử phù hợp trong môi trường số; giải quyết vấn đề với sự hỗ trợ của CNTT&TT; ứng dụng CNTT&TT trong học và tự học; hợp tác trong môi trường số. Môn Tin học trang bị cho HS hệ thống kiến thức tin học phổ thông gồm ba mạch kiến thức hoà quyện:

– Học vấn số hoá phổ thông nhằm giúp HS hoà nhập với xã hội hiện đại, sử dụng được các thiết bị số và phần mềm cơ bản thông dụng một cách có đạo đức, văn hoá và tuân thủ pháp luật.

– Công nghệ thông tin và truyền thông nhằm giúp HS sử dụng và áp dụng hệ thống máy tính giải quyết vấn đề thực tế một cách hiệu quả và sáng tạo.

– Khoa học máy tính nhằm giúp HS hiểu biết các nguyên tắc cơ bản và thực tiễn của tư duy máy tính, tạo cơ sở cho việc thiết kế và phát triển các hệ thống máy tính.

Để hỗ trợ HS trong việc định hướng nghề nghiệp tương lai, ngoài các kiến thức cốt lõi chung, nội dung kiến thức môn Tin học được chia thành hai định hướng Khoa học máy tính và Tin học ứng dụng, mỗi định hướng có những chủ đề kiến thức riêng và các Chuyên đề học tập riêng. Định hướng Khoa học máy tính đáp ứng mục đích bước đầu tìm hiểu nguyên lí hoạt động của hệ thống máy tính, phát triển tư duy máy tính, khả năng tìm tòi, khám phá các hệ thống tin học, phát triển ứng dụng trên hệ thống máy tính. Định hướng tin học ứng dụng đáp ứng nhu cầu sử dụng máy tính như một công cụ của công nghệ kĩ thuật số trong cuộc sống, học tập và làm việc, đem lại sự thích ứng và khả năng phát triển dịch vụ trong xã hội số.

2. Mục tiêu cấp Trung học phổ thông

Chương trình môn Tin học ở cấp THPT giúp HS củng cố và nâng cao năng lực tin học đã được hình thành, phát triển ở giai đoạn giáo dục cơ bản, đồng thời cung cấp cho HS tri thức mang tính định hướng nghề nghiệp thuộc lĩnh vực tin học hoặc ứng dụng tin học, cụ thể là:

– Giúp HS có những hiểu biết cơ bản về hệ thống máy tính, một số kĩ thuật thiết kế thuật toán, tổ chức dữ liệu và lập trình; củng cố và phát triển hơn nữa cho HS tư duy giải quyết vấn đề, khả năng đưa ra ý tưởng và chuyển giao nhiệm vụ cho máy tính thực hiện.

– Giúp HS có khả năng ứng dụng tin học, tạo ra sản phẩm số phục vụ cộng đồng và nâng cao hiệu quả công việc; có khả năng lựa chọn, sử dụng, kết nối các thiết bị số, dịch vụ mạng và truyền thông, phần mềm và các tài nguyên số khác.

– Giúp HS có khả năng hoà nhập và thích ứng được với sự phát triển của xã hội số, ứng dụng CNTT&TT trong học và tự học; tìm kiếm và trao đổi thông tin theo cách phù hợp, tuân thủ pháp luật, có đạo đức, ứng xử văn hoá và có trách nhiệm; có hiểu biết thêm một số ngành nghề thuộc lĩnh vực tin học, chủ động và tự tin trong việc định hướng nghề nghiệp tương lai của bản thân.

II. NỘI DUNG CHUYÊN ĐỀ HỌC TẬP TIN HỌC 12 – ĐỊNH HƯỚNG KHOA HỌC MÁY TÍNH

SCĐ gồm ba chuyên đề thành phần:

Chuyên đề 1. Tìm hiểu một vài kiểu dữ liệu tuyến tính

Giới thiệu hai cấu trúc dữ liệu: ngăn xếp (stack) và hàng đợi (queue). Các cấu trúc dữ liệu này, cùng với cấu trúc danh sách liên kết và mảng (đã được học trong chương trình lớp 11) là những cấu trúc dữ liệu tuyến tính đơn giản nhất, được sử dụng nhiều trong thực tế.

Chuyên đề 2. Tìm hiểu cây tìm kiếm nhị phân trong sắp xếp và tìm kiếm

Trình bày một cấu trúc dữ liệu có nhiều ứng dụng là cấu trúc dữ liệu cây (tree). Nội dung chính của chuyên đề tập trung vào cây nhị phân (binary tree) và cây tìm kiếm nhị phân (binary search tree), giúp HS thấy được cấu trúc cây tìm kiếm nhị phân có vai trò hết sức quan trọng trong giải quyết một cách hiệu quả các bài toán tìm kiếm và sắp xếp.

Chuyên đề 3. Tìm hiểu kĩ thuật duyệt đồ thị và ứng dụng

Làm quen bước đầu với mô hình đồ thị (graph), một cấu trúc dữ liệu đặc biệt và có nhiều ứng dụng. Giới thiệu cách biểu diễn các đồ thị vô hướng hoặc có hướng và hai thuật toán duyệt đồ thị nổi tiếng là duyệt theo chiều sâu (DFS) và duyệt theo chiều rộng (BFS).

Trong Chuyên đề này, cùng với việc mô tả các cấu trúc dữ liệu, HS được giới thiệu cách cài đặt các cấu trúc dữ liệu tương ứng trong ngôn ngữ lập trình Python cũng như sử dụng các cấu trúc dữ liệu ấy thông qua một số ví dụ đơn giản.

III. ĐỊNH HƯỚNG PHÁT TRIỂN PHẨM CHẤT, NĂNG LỰC VÀ PHƯƠNG PHÁP DẠY HỌC

CT GDPT 2018 môn Tin học đặc biệt chú trọng việc quán triệt yêu cầu đổi mới triệt để GDPT theo tinh thần chuyển từ việc chú trọng truyền thụ kiến thức sang phát triển toàn diện phẩm chất và năng lực cho HS. Bên cạnh việc góp phần thực hiện các yêu cầu cần đạt về phẩm chất chủ yếu và năng lực chung theo các mức độ phù hợp với môn học, cấp học đã được quy định trong Chương trình tổng thể, CT GDPT 2018 đã đưa ra các yêu cầu cần đạt

liên quan tới các năng lực đặc thù. Cụ thể, giáo dục tin học cần giúp HS hình thành, phát triển năng lực tin học với năm thành phần cụ thể sau đây:

– NLa: Sử dụng và quản lí các phương tiện CNTT&TT;

– NLb: Ứng xử phù hợp trong môi trường số;

– NLc: Giải quyết vấn đề với sự hỗ trợ của CNTT&TT;

– NLd: Ứng dụng CNTT&TT trong học và tự học;

– NLe: Hợp tác trong môi trường số.

Để thực hiện được các yêu cầu trên cần hết sức quan tâm tới việc thiết kế hoạt động dạy học cho mỗi nội dung, mỗi chủ đề học tập. Các phương pháp giáo dục chủ yếu được lựa chọn theo các định hướng sau:

– Tăng cường sử dụng các phương pháp dạy học phát huy tính tích cực, chủ động, sáng tạo và phù hợp với sự hình thành và phát triển phẩm chất, năng lực cho HS. Cùng với phương pháp dạy học nêu và giải quyết vấn đề để phát triển tư duy máy tính, cần hết sức coi trọng dạy học trực quan và thực hành. Tăng cường thực hành, trải nghiệm trong các nội dung dạy học, đặc biệt liên quan tới các chủ đề như Ứng dụng tin học, Giải quyết vấn đề với sự trợ giúp của máy tính cũng như các chuyên đề theo định hướng chuyên sâu Khoa học máy tính và Tin học ứng dụng.

– Phối hợp các phương pháp dạy học theo hướng hoạt động, phương pháp dạy học dự án để phát triển năng lực vận dụng kiến thức kĩ năng giải quyết các vấn đề thực tiễn, bồi dưỡng kĩ năng làm việc theo nhóm, kĩ năng tổ chức công việc và năng lực tự học cho HS.

– Có thể tổ chức các hoạt động học tập của HS dựa trên các hoạt động trải nghiệm; hướng dẫn HS quan sát, gắn kết kiến thức tin học với thực tiễn, vận dụng kiến thức để định hướng giải quyết các vấn đề thực tiễn nhằm nâng cao sự hứng thú của HS, nhất là với các chủ đề như Đạo đức, pháp luật và văn hoá trong môi trường số; Hướng nghiệp tin học;...

– Kết hợp giáo dục STEM trong dạy học tin học nhằm phát triển cho HS khả năng tích hợp các kiến thức kĩ năng của các lĩnh vực: Toán, Công nghệ và Tin học vào việc nghiên cứu giải quyết một số tình huống thực tiễn cũng như ứng dụng tin học trong học tập các môn học khác.

– Tuỳ theo nội dung bài học, GV vận dụng các phương pháp giáo dục một cách linh hoạt, sáng tạo, phù hợp với mục tiêu, nội dung giáo dục, đối tượng HS và điều kiện cụ thể. Tuỳ theo yêu cầu cần đạt, GV có thể sử dụng phối hợp nhiều phương pháp dạy học trong một chủ đề. Các phương pháp dạy học truyền thống (thuyết trình, đàm thoại,...) được sử dụng theo hướng phát huy tính tích cực, chủ động của HS. Tăng cường sử dụng các phương pháp dạy học hiện đại để cao vai trò chủ thể học tập của HS (dạy học thực hành, dạy học dựa trên giải quyết vấn đề, dạy học dựa trên dự án, dạy học dựa trên trải nghiệm, khám phá; dạy học phân hoá, dạy học tình huống,... cùng những kĩ thuật dạy học phù hợp).

IV. NHỮNG NÉT CHUNG VỀ PHƯƠNG PHÁP DẠY HỌC CHUYÊN ĐỀ HỌC TẬP

1. Định hướng, yêu cầu cơ bản chung về đổi mới phương pháp dạy học Chuyên đề học tập

Tất cả các phương pháp dạy học đều nhằm mục đích hình thành, phát triển, hoàn thiện năng lực và phẩm chất của HS. Phương pháp dạy học Chuyên đề học tập về cơ bản cũng được thực hiện theo các định hướng chung đối với môn Tin học. Các nhóm năng lực chung cũng như năng lực đặc thù của môn Tin học chỉ có thể hình thành và phát triển tốt với các phương pháp và hình thức dạy học tích cực theo các định hướng:

– Để phát triển năng lực nhận thức tin học cần tạo cho HS cơ hội huy động những hiểu biết, kinh nghiệm sẵn có để tham gia hình thành kiến thức mới một cách chủ động thông qua các hoạt động học tập tự khám phá, bổ sung kiến thức nhờ các công cụ tin học. Chú ý tổ chức các hoạt động trong đó HS có thể diễn đạt hiểu biết của mình bằng cách so sánh, phân loại, hệ thống hoá kiến thức, vận dụng kiến thức đã học để giải quyết vấn đề, qua đó kết nối được kiến thức mới với hệ thống kiến thức đã có. Trong *Chuyên đề học tập Tin học 12 – Định hướng Khoa học máy tính*, các chuyên đề thành phần liên quan tới các cấu trúc dữ liệu cơ bản: ngăn xếp, hàng đợi, cây tìm kiếm, đồ thị đều có thể sử dụng một cách hiệu quả phương pháp dạy học nêu và giải quyết vấn đề giúp phát triển tư duy máy tính cho HS cũng như phương pháp dạy học trực quan, dạy học thực hành.

– Trong quá trình tổ chức dạy học, dù sử dụng bất cứ phương pháp giảng dạy nào, cũng cần coi trọng việc khai thác tối đa tính đa dạng và phong phú của môi trường và công cụ số giúp cho HS hình thành, phát triển và hoàn thiện kĩ năng quan sát, thu thập thông tin; phân tích, phát hiện và giải quyết vấn đề; khả năng làm việc theo nhóm.

– Để phát triển và hoàn thiện năng lực vận dụng kiến thức, kĩ năng đã học, GV cần vận dụng một số phương pháp dạy học đặc thù có ưu thế trong việc phát triển năng lực này như phương pháp dạy học tìm tòi, khám phá, dạy học thực hành,... Tạo cơ hội cho HS đề xuất, tiếp cận với các tình huống thực tiễn, đọc, thu thập các thông tin về các vấn đề thực tiễn có liên quan đến những kiến thức và kĩ năng đã học để có thể sử dụng những kiến thức và kĩ năng này trong việc giải thích, đưa ra giải pháp giải quyết vấn đề. Quan tâm giúp HS hình thành và phát triển kĩ năng lập kế hoạch, hợp tác trong hoạt động nhóm, kĩ năng giao tiếp, thảo luận, tranh luận, báo cáo,... Cần chú ý tạo điều kiện cho HS cơ hội liên hệ vận dụng kiến thức, kĩ năng tích hợp trong môn Tin học với kiến thức, kĩ năng của các môn học khác, đặc biệt là Toán và Công nghệ vào việc giải quyết các vấn đề thực tiễn.

– Trong mỗi bài học cụ thể, có thể phối kết hợp đồng thời nhiều phương pháp và hình thức giáo dục tích cực khác nhau để có thể đạt hiệu quả cao nhất trong việc hình thành các phẩm chất, kĩ năng mà HS cần đạt được.

2. Hướng dẫn, gợi ý một số phương pháp, tổ chức hoạt động dạy học

Các phương pháp dạy học rất đa dạng và phong phú, song có thể chia thành hai loại chính: i) các phương pháp dạy học truyền thống (thuyết trình, vấn đáp và luyện tập củng cố,...); ii) các phương pháp dạy học tích cực (truy vấn; vấn đáp tìm tòi; đàm thoại phát hiện;

phát hiện và giải quyết vấn đề; dạy học theo dự án,...). Mỗi phương pháp dạy học đều có quy trình thực hiện cụ thể. Ví dụ:

– Dạy học phát hiện và giải quyết vấn đề (DHPH&GQVĐ) là phương pháp dạy học yêu cầu HS học cách phát hiện và giải quyết vấn đề một cách khoa học. Các bước của DHPH&GQVĐ gồm: i) Phát hiện vấn đề và phát biểu vấn đề cần xử lí; ii) Đề xuất giải pháp giải quyết vấn đề; iii) Thực hiện giải quyết vấn đề theo giải pháp đã đề xuất; iv) Thảo luận và rút ra kết luận; v) Báo cáo kết quả xử lí.

– Dạy học dự án (DHDA) được coi là phương pháp dạy học của kiểu học tích hợp việc học để biết với việc học để làm. Trong phương pháp dạy học này GV không những phải tạo điều kiện cho HS hình thành và phát triển kiến thức và kĩ năng thông qua nội dung chương trình học tập, mà còn áp dụng những gì họ đã học tập vào việc giải quyết vấn đề thiết thực của cuộc sống, tạo ra sản phẩm có ý nghĩa. Các bước của DHDA gồm: i) Lập kế hoạch dự án (đề xuất ý tưởng, chọn dự án, xác định mục tiêu, kế hoạch triển khai, phân công công việc trong nhóm); ii) Thực hiện dự án; iii) Trình bày, bảo vệ dự án (báo cáo về sản phẩm, đánh giá sản phẩm,...).

Trong nhiều trường hợp, bài học không nhất thiết sử dụng một phương pháp dạy học cụ thể mà là sự kết hợp nhiều phương pháp, kĩ thuật khác nhau hoặc dựa trên tinh thần của một/vài phương pháp hay quan điểm/lí thuyết dạy học nào đó. Trong những trường hợp như vậy, khi xây dựng Kế hoạch bài dạy, GV chỉ cần chỉ ra tên gọi của chúng, ví dụ "dạy học cộng tác"; "dạy học hợp tác"; "dạy học dựa trên tình huống"; "dạy học dựa trên trường hợp"; "dạy học dựa trên quan điểm hoạt động"; "dạy học kiến tạo";...

Trong thực tế, có một số hoạt động đặc thù có thể tổ chức cho HS thực hiện khi dạy học Chuyên đề học tập. Cách tổ chức các hoạt động này được xem như những phương pháp dạy học riêng, ví dụ "dạy học nhận dạng và thể hiện"; "dạy học khái niệm theo con đường suy diễn"; "dạy học phát triển tư duy thuật toán"; "dạy học phát triển tư duy máy tính";...

3. Một số lưu ý liên quan tới phương pháp giáo dục Chuyên đề học tập Tin học 12 – Định hướng Khoa học máy tính

– Nội dung Chuyên đề tập trung vào việc hệ thống hoá và bổ sung các kiến thức liên quan tới cấu trúc dữ liệu – kiến thức không thể thiếu đối với các chuyên gia tin học và là một trong hai nền tảng cốt lõi để có thể thực hiện việc giải quyết vấn đề với sự trợ giúp của máy tính theo công thức nổi tiếng: Chương trình = Cấu trúc dữ liệu + Thuật toán. Do vậy, việc kế thừa và tiếp tục phát triển các năng lực chung và năng lực tin học của HS đã được hình thành ở giai đoạn giáo dục cơ bản, nhất là các năng lực liên quan tới kĩ thuật lập trình trên Python, để nghiên cứu nội dung này là hết sức quan trọng.

– Coi trọng việc "Kết nối tri thức với cuộc sống" là một trong những phương thức quan trọng để HS hình thành và phát triển năng lực. Do vậy, cần chủ động gợi mở các tình huống liên hệ kiến thức với ứng dụng thực tế, tổ chức các hoạt động gắn với những bối cảnh thực tiễn, bổ sung/thay thế các ví dụ minh hoạ, các câu hỏi/bài tập trong Chuyên đề bằng các ví dụ/câu hỏi/bài tập mang tính thực tế.

– Các định hướng Tin học ứng dụng và Khoa học máy tính đều hỗ trợ phát triển các khả năng tư duy đặc trưng riêng. Do vậy, khi triển khai nội dung dạy học, cần lưu ý tới vấn đề này. Với định hướng Khoa học máy tính, cần chú ý trọng tâm phát triển tư duy giải quyết vấn đề, nhất là tư duy thuật toán. Do vậy, để dạy học Chuyên đề này hiệu quả, phương pháp dạy học phát hiện và giải quyết vấn đề một cách khoa học rất quan trọng. Việc kết hợp dạy học thực hành, cho HS quan sát các ví dụ minh hoạ, sau đó cùng trao đổi, thảo luận để hình thành phương án giải quyết sẽ giúp HS bồi dưỡng và phát triển tư duy giải quyết vấn đề, làm việc theo nhóm,...

– Trong giai đoạn giáo dục định hướng nghề nghiệp HS đã có sự trưởng thành không chỉ về thể chất, tâm sinh lí, mà cả về năng lực tư duy. Đây là giai đoạn HS bắt đầu có tư duy độc lập và sáng tạo hơn so với giai đoạn giáo dục cơ bản. Do vậy, GV cần tạo điều kiện để HS thể hiện năng lực phân tích, tổng hợp, phân tích, so sánh, phê phán,... và đặc biệt cần tôn trọng và hướng dẫn HS thể hiện chính kiến, tư duy độc lập, tư duy phản biện tích cực và có cơ sở khoa học, thực tiễn cũng như chủ động học tập, chủ động tìm kiếm tri thức mới và tự tin phát triển năng lực cá nhân.

– Đây cũng là thời điểm HS phát triển nhu cầu giao tiếp, nhu cầu tự khẳng định bản thân như một người lớn. Xu hướng chung của các HS đó là mong muốn thể hiện cá tính độc đáo để gây sự quan tâm chú ý từ những người xung quanh, nhất là các bạn bè cùng trang lứa. GV cần khai thác đặc điểm tâm sinh lí này để hướng dẫn, khích lệ HS sáng tạo những sản phẩm số thiết thực cho cuộc sống và học tập, động viên HS thể hiện tinh thần trách nhiệm không chỉ trong việc tự học mà cả trong mọi hoạt động chung, hoạt động theo nhóm.

– Cần vận dụng linh hoạt các phương pháp dạy học tích cực tuỳ theo các điều kiện, hoàn cảnh và tình huống cụ thể. Khuyến khích và đánh giá cao khả năng tự học chủ động, coi trọng hướng dẫn HS tự học, tạo điều kiện cho các hoạt động theo nhóm; không chỉ quan tâm tới kết quả cuối cùng mà cần quan tâm đến cả quá trình học tập của HS.

4. Gợi ý dạy học một số dạng hoạt động điển hình

4.1. Hoạt động khởi động

Nhằm hỗ trợ cho việc triển khai các phương pháp dạy học tích cực, các bài học trong Chuyên đề đều được viết theo mô hình hoạt động. Mỗi bài học đều được bắt đầu với một hoạt động khởi động. Hoạt động này vừa để hỗ trợ tổ chức dạy học tích cực, vừa để thu hút sự chú ý của HS, giúp dẫn dắt vào bài học một cách tự nhiên. Các hoạt động khởi động thường là: i) nhắc lại kiến thức cũ; ii) kể lại một trải nghiệm; iii) đề cập tới một tình huống, một thực tế, một vấn đề,...

Với các hoạt động khởi động dạng i), iii), GV có thể nêu vấn đề dưới dạng các câu hỏi truy vấn để HS thử suy nghĩ, trả lời. Tuỳ từng trường hợp cụ thể, có thể cho phép HS trao đổi nhóm để tìm cách giải quyết. *Cần lưu ý là không nhất thiết HS phải trả lời được ngay các câu hỏi đặt ra rồi mới bắt đầu nội dung bài học.* Vì không ít trường hợp câu trả lời cho tình huống đặt ra trong phần khởi động HS có thể tìm thấy trong các đơn vị kiến thức có trong bài học dưới sự dẫn dắt của GV.

Với tình huống khởi động dạng ii) có thể đơn giản yêu cầu HS đọc và GV bổ sung một số chi tiết để làm phong phú thêm trải nghiệm được nói tới.

4.2. Hoạt động khám phá kiến thức

Các đơn vị kiến thức được trình bày trong mỗi bài học đều được gắn với những hoạt động khám phá kiến thức cụ thể. Đây là thành phần rất quan trọng góp phần hình thành năng lực của HS, hỗ trợ việc biến quá trình đào tạo thành quá trình tự đào tạo. Hoạt động này cho phép HS tìm hiểu, khám phá kiến thức một cách tích cực dưới nhiều hình thức. Đó có thể là: i) Thảo luận một vấn đề; ii) Giải quyết một vấn đề; iii) Đọc hiểu nếu là đơn vị kiến thức liên quan tới những vấn đề mà HS có ít trải nghiệm.

Với hoạt động khám phá kiến thức dạng i), tuỳ từng tình huống cụ thể, GV chia lớp học thành các nhóm 2, 3 hoặc nhiều hơn số HS để các em trao đổi từ đó rút ra các kết luận tương ứng. Sau đó mời đại diện một, hai nhóm trình bày và các nhóm khác tham gia góp ý, trao đổi, tranh luận,...

Với hoạt động khám phá kiến thức dạng ii), việc giải quyết do các cá nhân hay các nhóm HS thực hiện cũng cần căn cứ nội dung khám phá cụ thể. Với những vấn đề khó, GV nên có những hướng dẫn gợi mở để HS đi tới lời giải cho vấn đề đặt ra, nhưng cố gắng không làm thay cho HS mà chỉ đóng vai trò dẫn dắt.

Với hoạt động khám phá kiến thức dạng iii), GV cho các HS tự đọc hiểu sau đó đưa ra một vài câu hỏi để thông qua đó vừa đánh giá mức độ đọc hiểu, vừa để dẫn dắt HS cùng đi tới nội dung kiến thức cần chốt.

4.3. Hoạt động củng cố và luyện tập

Các câu hỏi, bài tập củng cố kiến thức và luyện tập có thể được thực hiện theo nhóm hay từng cá nhân HS, tuỳ theo nội dung cụ thể.

Nói chung, các câu hỏi, bài tập củng cố kiến thức GV có thể thực hiện truy vấn trực tiếp cá nhân HS. Với trường hợp có khó khăn, lúng túng khi trả lời thì gợi ý HS xem lại ngay nội dung đơn vị kiến thức vừa được học ngay trước đó trong các hộp kiến thức hoặc cả mục nội dung kiến thức nếu cần.

Các câu hỏi, bài tập luyện tập thường có yêu cầu tổng hợp tri thức có trong bài học, GV có thể có những gợi ý, dẫn dắt để tất cả các HS hoàn thành tại lớp. Các phương pháp đàm thoại, gợi mở, nêu và giải quyết vấn đề,... có thể được vận dụng, phối hợp một cách linh hoạt.

4.4. Hoạt động vận dụng

Đây là hoạt động cuối cùng của mỗi bài học, giúp HS tăng cường năng lực vận dụng với nhiều loại hình phong phú. Hoạt động này mang tính mở, khuyến khích HS về nhà thực hiện tuỳ theo khả năng và điều kiện cụ thể, sau đó nộp bài dưới hình thức viết báo cáo. Nói chung, các bài tập, câu hỏi vận dụng thường yêu cầu mức độ tổng hợp tri thức ở mức cao hơn so với các bài luyện tập để tìm hiểu hay giải quyết một vấn đề thực tiễn hoặc thực hiện một trải nghiệm. GV nên có những hướng dẫn để HS về nhà tự làm (không bắt buộc).

V. MỘT SỐ GỢI Ý KIỂM TRA, ĐÁNH GIÁ KẾT QUẢ HỌC TẬP

1. Hướng dẫn chung

Căn cứ đánh giá là các yêu cầu cần đạt về phẩm chất và năng lực được quy định trong Chương trình tổng thể và Chương trình môn học. Phạm vi đánh giá là toàn bộ nội dung và yêu cầu cần đạt của Chương trình môn Tin học 12, trong đó, với Chuyên đề học tập phải thông qua đánh giá thường xuyên kết quả và các sản phẩm trong quá trình học tập Chuyên đề. Chuyên đề học tập Tin học 12 chủ yếu sử dụng các hình thức đánh giá sau:

– Đánh giá thông qua bài kiểm tra ngắn dưới dạng tự luận, trắc nghiệm khách quan,...

– Đánh giá thông qua vấn đáp: câu hỏi vấn đáp, phỏng vấn, thuyết trình,...

– Đánh giá thông qua quan sát: quan sát thái độ, hoạt động của HS qua bài thực hành, thảo luận nhóm, thực hiện các dự án học tập,... bằng một số công cụ như sử dụng bảng quan sát, bảng kiểm, hồ sơ học tập,...

2. Gợi ý đánh giá theo định hướng tiếp cận phẩm chất, năng lực

2.1. Mục tiêu và căn cứ

Mục tiêu đánh giá kết quả giáo dục là cung cấp thông tin chính xác, kịp thời, có giá trị về mức độ đáp ứng yêu cầu cần đạt của chương trình và sự tiến bộ của HS để hướng dẫn hoạt động học tập, điều chỉnh các hoạt động dạy, quản lí và phát triển chương trình, bảo đảm sự tiến bộ của từng HS và nâng cao chất lượng giáo dục.

Căn cứ đánh giá là các yêu cầu cần đạt về phẩm chất và năng lực được quy định trong Chương trình tổng thể và chương trình môn Tin học. Với các Chuyên đề học tập, phạm vi đánh giá được giới hạn trong nội dung và yêu cầu cần đạt của Chuyên đề.

2.2. Hình thức đánh giá

Chủ yếu là các hình thức đánh giá quá trình (đánh giá thường xuyên) tích hợp vào trong các hoạt động dạy và học của GV và HS.

2.3. Các nguyên tắc đánh giá

– Việc đánh giá thường xuyên phải bám sát các thành phần năng lực đặc thù môn Tin học (xem Mục 4) cũng như các biểu hiện của những phẩm chất và năng lực chung đã được xác định trong CT GDPT 2018. Các công cụ đánh giá cần được xây dựng trên cơ sở đáp ứng các yêu cầu cần đạt được nêu trong Chương trình môn Tin học 12.

– Trong kiểm tra, đánh giá cần coi trọng khả năng giải quyết vấn đề của HS. Với các mạch kiến thức về cấu trúc dữ liệu cần quan tâm tới năng lực hiểu và vận dụng kiến thức, kĩ năng để giải quyết những bài toán cụ thể, thiết thực, đồng thời cần coi trọng việc đánh giá năng lực tư duy hệ thống và sáng tạo.

– Đa dạng hoá các hình thức đánh giá; tạo điều kiện để HS tham gia đánh giá và tự đánh giá. Mục tiêu cao nhất của đánh giá là vì sự tiến bộ trong học tập, do vậy phải làm sao cho việc kiểm tra, đánh giá không trở thành gánh nặng đối với HS, mà phải là động lực để thúc đẩy HS đạt được những tiến bộ mới.

– Kết luận đánh giá của GV về năng lực tin học của mỗi HS được dựa trên sự tổng hợp các kết quả đánh giá thường xuyên trong suốt quá trình học tập.

2.4. Phương pháp đánh giá và công cụ đánh giá

– Kết hợp đánh giá của GV với tự đánh giá và đánh giá đồng đẳng của HS. Phối hợp đánh giá tình huống; đánh giá qua trắc nghiệm; đánh giá thông qua phản hồi và phản ánh; đánh giá thông qua quan sát.

– Kết hợp đánh giá sản phẩm học tập (bài kiểm tra tự luận ngắn, bài kiểm tra trắc nghiệm khách quan, trả lời miệng, thuyết trình, bài thực hành,...) với đánh giá qua quan sát (thái độ và hành vi trong thảo luận, làm việc nhóm, thực hành,...).

Lựa chọn các phương pháp, công cụ phù hợp để đánh giá một năng lực cụ thể:

– Để đánh giá các năng lực đặc thù NLa, NLc có thể sử dụng các câu hỏi (nói, viết), bài tập,... đòi hỏi HS phải trình bày, so sánh, hệ thống hoá kiến thức hay phải vận dụng kiến thức để giải thích, giải quyết vấn đề.

– Để đánh giá thành phần năng lực NLb, NLc, NLd có thể đưa ra các tình huống yêu cầu HS xử lí hoặc tìm kiếm các thông tin cần thiết để hỗ trợ.

– Để đánh giá thành phần năng lực NLe có thể dùng bảng kiểm hoặc ghi chép kết quả quan sát của GV theo các tiêu chí đã xác định về tiến trình thực hiện các nhiệm vụ tìm tòi, khám phá, tổ chức thực hiện của HS,...

Tất cả các phương pháp dạy học đều nhằm mục đích hình thành, phát triển, hoàn thiện năng lực và phẩm chất của HS. Các nhóm năng lực chung cũng như năng lực đặc thù của môn Tin học chỉ có thể hình thành và phát triển tốt với các phương pháp và hình thức dạy học tích cực theo các định hướng:

– Để phát triển năng lực nhận thức tin học cần tạo cho HS cơ hội huy động những hiểu biết, kinh nghiệm sẵn có để tham gia hình thành kiến thức mới một cách chủ động thông qua các hoạt động học tập tự khám phá, bổ sung kiến thức nhờ các công cụ tin học. Chú ý tổ chức các hoạt động trong đó HS có thể diễn đạt hiểu biết của mình bằng cách so sánh, phân loại, hệ thống hoá kiến thức, vận dụng kiến thức đã học để giải quyết vấn đề, qua đó kết nối được kiến thức mới với hệ thống kiến thức đã có.

– Trong quá trình tổ chức dạy học, dù sử dụng bất cứ phương pháp giảng dạy nào, cũng cần coi trọng việc khai thác tối đa tính đa dạng và phong phú của môi trường và công cụ số giúp cho HS hình thành, phát triển và hoàn thiện kĩ năng quan sát, thu thập thông tin; phân tích, giải quyết vấn đề và làm việc theo nhóm.

– Để phát triển và hoàn thiện năng lực vận dụng kiến thức, kĩ năng đã học, GV cần vận dụng một số phương pháp dạy học đặc thù có ưu thế trong việc phát triển năng lực này như phương pháp dạy học tìm tòi, khám phá, dạy học thực hành,... Tạo cơ hội cho HS đề xuất, tiếp cận với các tình huống thực tiễn, đọc, thu thập các thông tin về các vấn đề thực tiễn có liên quan đến những kiến thức và kĩ năng đã học để có thể sử dụng những kiến thức và kĩ năng này trong việc giải thích, đưa ra giải pháp giải quyết vấn đề. Quan tâm giúp HS hình thành và phát triển kĩ năng lập kế hoạch, hợp tác trong hoạt động nhóm, kĩ năng giao tiếp, thảo luận, tranh luận, báo cáo,... Cần chú ý tạo điều kiện cho HS cơ hội liên hệ vận dụng

kiến thức, kĩ năng tích hợp trong môn Tin học với kiến thức, kĩ năng của các môn học khác, đặc biệt là Toán và Công nghệ vào việc giải quyết các vấn đề thực tiễn. Giáo dục STEM (hay STEAM) cần được coi là biện pháp hữu hiệu để thực hiện yêu cầu trên.

– Trong mỗi bài học cụ thể, có thể phối kết hợp đồng thời nhiều phương pháp và hình thức giáo dục tích cực khác nhau để có thể đạt hiệu quả cao nhất trong việc hình thành các phẩm chất, kĩ năng mà HS cần đạt được.

VI. PHÂN PHỐI CHƯƠNG TRÌNH VÀ DỰ KIẾN KẾ HOẠCH DẠY HỌC

Dưới đây là một gợi ý phân bổ thời lượng. Tuỳ điều kiện thực tế và trình độ của HS, GV có thể điều chỉnh thời lượng các bài học cho phù hợp.

Bài số	Tên bài/Chuyên đề	Số tiết LT	Số tiết TH
	Chuyên đề 1. Tìm hiểu một vài kiểu dữ liệu tuyến tính		
1	Mô hình dữ liệu ngăn xếp và hàng đợi	2	
2	Kiểu dữ liệu ngăn xếp	2	
3	Thực hành kiểu dữ liệu ngăn xếp		2
4	Kiểu dữ liệu hàng đợi	2	
5	Thực hành kiểu dữ liệu ngăn xếp và hàng đợi		2
	Chuyên đề 2. Tìm hiểu cây tìm kiếm nhị phân trong sắp xếp và tìm kiếm		
6	Cây nhị phân	2	
7	Cây tìm kiếm nhị phân	2	
8	Thực hành cây tìm kiếm nhị phân		2
9	Các thuật toán duyệt trên cây tìm kiếm nhị phân	2	
10	Thực hành tổng hợp với cây tìm kiếm nhị phân		2
	Chuyên đề 3. Tìm hiểu kĩ thuật duyệt đồ thị và ứng dụng		
11	Khái niệm đồ thị	2	
12	Biểu diễn đồ thị	2	
13	Thực hành thiết lập đồ thị		2
14	Kĩ thuật duyệt đồ thị theo chiều sâu	3	
15	Thực hành duyệt đồ thị theo chiều sâu		2
16	Kĩ thuật duyệt đồ thị theo chiều rộng	2	
17	Thực hành duyệt đồ thị tổng hợp		2
	Tổng	**21**	**14**

PHẦN
HAI

HƯỚNG DẪN DẠY HỌC CÁC BÀI CỤ THỂ

MỘT SỐ GỢI Ý DẠY HỌC CHUNG

Phương pháp chung giảng dạy phần kiến thức lí thuyết:

– Với các bài liên quan đến mô hình và cấu trúc dữ liệu:

1. Giới thiệu một số mô hình hoặc bài toán thực tế có liên quan đến cấu trúc dữ liệu mới.

2. Mô tả mô hình và các thông số, phương thức hoạt động của mô hình dữ liệu mới.

3. Thiết lập cách cài đặt cấu trúc dữ liệu mới, các lệnh, thao tác hoặc hàm chính của cấu trúc dữ liệu trên môi trường lập trình cụ thể, ở đây là Python.

4. Giải hoặc minh hoạ một số bài toán thực tế sử dụng cấu trúc dữ liệu mới.

– Với các bài liên quan đến thuật toán:

1. Giới thiệu bài toán, ý nghĩa thực tế của thuật toán.

2. Mô tả chính xác yêu cầu bài toán, đầu vào, đầu ra của thuật toán.

3. Mô tả, giải thích ý nghĩa các bước thực hiện thuật toán. Tại bước này có thể minh hoạ thuật toán trên các bộ dữ liệu đầu vào đơn giản để HS quan sát được.

4. Mô tả chi tiết thuật toán.

5. Thực hiện viết chương trình cài đặt thuật toán đã mô tả ở bước trên.

Tất cả các bước trên GV có thể thực hiện và tổ chức giảng dạy theo các cách khác nhau tuỳ thuộc vào tình tình cụ thể tại lớp của mình.

Phương pháp chung giảng dạy phần thực hành:

1. GV giới thiệu tổng quan mục đích của bài thực hành, các nhiệm vụ cần thực hiện.

2. Với mỗi nhiệm vụ, GV đọc đề bài, phân tích cách làm, sau đó yêu cầu HS thực hiện dựa trên SCĐ đã có.

3. Trong quá trình thực hiện nếu HS gặp khó khăn, GV sẽ hướng dẫn và giải quyết các khó khăn đó của HS.

KẾT NỐI TRI THỨC
VỚI CUỘC SỐNG

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

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

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

1. Kiến thức

– Mô hình dữ liệu ngăn xếp và hàng đợi.

– Cơ chế hoạt động LIFO của ngăn xếp.

– Cơ chế hoạt động FIFO của hàng đợi.

2. Năng lực

Biết và 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

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. Hoặc có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2
Tìm hiểu mô hình dữ liệu ngăn xếp.	Tìm hiểu mô hình dữ liệu hàng đợi.

Hoạt động khởi động

Mục đích: HS làm quen với những hiện tượng, sự vật trong cuộc sống có liên quan đến mô hình ngăn xếp và hàng đợi.

Tổ chức hoạt động: GV đưa ra một số mô hình thực tế có hoạt động gần giống như ngăn xếp, hàng đợi và yêu cầu các nhóm HS trả lời.

Lưu ý:

– GV có thể đưa thêm các ví dụ thực tế, không nhất thiết phải giống như SCĐ. Mỗi mô hình ngăn xếp hay hàng đợi nên có nhiều hơn một ví dụ thực tế minh hoạ.

– Phần này không yêu cầu HS trả lời chính xác. Mục đích của hoạt động khởi động là để HS làm quen và tò mò muốn biết những mô hình này có điều gì đặc biệt, giống nhau và khác nhau như thế nào, tất cả sẽ được học trong bài học.

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

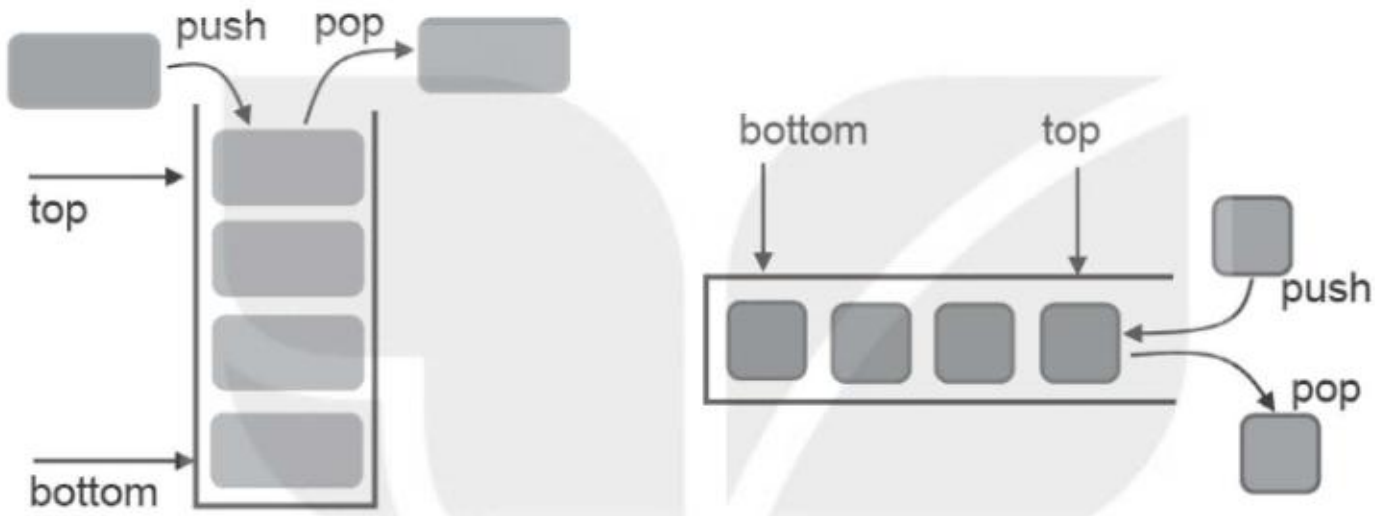
Mục đích: HS biết và nắm được cơ chế hoạt động LIFO (*vào sau, ra trước*) của ngăn xếp, các thao tác (lệnh) chính và ý nghĩa của các lệnh này.

Tổ chức hoạt động: GV chia nhóm HS và thực hiện các bước sau:

1. Yêu cầu 1 nhóm mô tả kĩ hơn, chính xác hơn các thao tác dựa trên một ví dụ đã có trong phần khởi động, nhấn mạnh đến cơ chế *vào sau, ra trước* LIFO.

2. Yêu cầu 1 nhóm mô tả kĩ hơn hoạt động của thao tác Undo trong phần mềm soạn thảo văn bản, nhấn mạnh đến cơ chế *vào sau, ra trước*.

3. Yêu cầu 1 nhóm trình bày các định nghĩa chính liên quan đến ngăn xếp thông qua sơ đồ và phần kiến thức đã trình bày trong SCĐ.



Hình 1.1. Mô hình ngăn xếp (Hình 1.2 SCĐ)

GV chốt lại mô hình hoạt động của ngăn xếp thông qua các thao tác cơ bản với ngăn xếp (Bảng 1.1 SCĐ) và liên hệ các thao tác, lệnh cơ bản này với các ví dụ đã biết.

Số thứ tự	Thao tác	Hàm/Lệnh
1	Tạo một ngăn xếp rỗng.	`S = Stack()`
2	Đưa phần tử x vào đỉnh ngăn xếp S.	`push(S,x)`
3	Lấy ra một phần tử từ đỉnh của ngăn xếp S và trả về phần tử này.	`pop(S)`
4	Kiểm tra ngăn xếp rỗng. Trả về True nếu S rỗng, ngược lại trả về False.	`isEmptyStack(S)`
5	Trả về phần tử tại vị trí đỉnh của ngăn xếp S, S không thay đổi.	`top(S)`

Hình 1.2. Các thao tác cơ bản trên dữ liệu ngăn xếp (Bảng 1.1 SCĐ)

Hoạt động củng cố kiến thức

1. Cần dùng lệnh pop(S) để lấy ra tất cả các phần tử của ngăn xếp cho đến khi ngăn xếp trở nên rỗng, phần tử cuối cùng nhận được chính là phần tử cần tìm.

2. Đáp án: S chứa các phần tử (tính từ đáy lên hay trái sang): 1, 10.

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

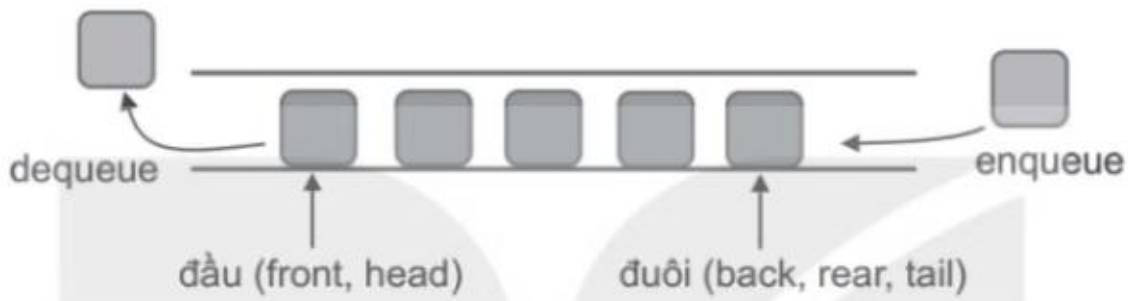
Mục đích: HS biết và nắm được cơ chế hoạt động FIFO (*vào trước, ra trước*) của hàng đợi, các thao tác (lệnh) chính và ý nghĩa của các lệnh này.

Tổ chức hoạt động: GV chia nhóm HS và thực hiện các bước sau:

1. Yêu cầu 1 nhóm mô tả kĩ hơn, chính xác hơn các thao tác dựa trên một ví dụ đã có trong phần khởi động, nhấn mạnh đến cơ chế *vào trước, ra trước* FIFO.

2. Yêu cầu 1 nhóm mô tả kĩ hơn hoạt động của thao tác in tài liệu trong văn phòng sử dụng chung máy in, nhấn mạnh đến cơ chế *vào trước, ra trước*.

3. Yêu cầu 1 nhóm trình bày các định nghĩa chính liên quan đến hàng đợi thông qua sơ đồ và phần kiến thức đã trình bày trong SCĐ.



Hình 1.3. Mô hình hàng đợi (Hình 1.3 SCĐ)

GV chốt lại mô hình hoạt động của hàng đợi thông qua các thao tác cơ bản với hàng đợi (Bảng 1.2 SCĐ) và liên hệ các thao tác, lệnh cơ bản này với các ví dụ đã thảo luận trước đây.

STT	Thao tác	Hàm/Lệnh
1	Tạo một hàng đợi rỗng.	`Q = Queue()`
2	Đưa phần tử x vào cuối của hàng đợi Q.	`enqueue(Q,x)`
3	Lấy ra một phần tử tại đầu của hàng đợi Q và trả về phần tử này.	`dequeue(Q)`
4	Kiểm tra hàng đợi rỗng. Hàm trả về True nếu Q rỗng, ngược lại trả về False.	`isEmptyQueue(Q)`
5	Trả về phần tử đầu của hàng đợi Q và Q không thay đổi.	`front(Q)`

Hình 1.4. Các thao tác cơ bản trên dữ liệu hàng đợi (Bảng 1.2 SCĐ)

Hoạt động củng cố kiến thức

1. Giống nhau:

– Là mô hình lưu trữ dữ liệu.

– Cho phép đưa dữ liệu vào và lấy dữ liệu ra.

Khác nhau ở vị trí đưa dữ liệu vào và lấy dữ liệu ra:

– Với ngăn xếp, đưa vào và lấy ra ở cùng một đầu, theo nguyên tắc *vào sau, ra trước* (LIFO).

– Với hàng đợi, đưa vào và lấy ra ở hai đầu khác nhau, theo nguyên tắc *vào trước, ra trước* (FIFO).

2. Đáp án: Q chứa phần tử 1.

Hoạt động luyện tập

1. Thu được dãy số sắp xếp ngược lại của dãy gốc.

2. Các số được in ra màn hình là: 1, 5.

Hoạt động vận dụng

1. GV hỗ trợ HS tìm thêm các mô hình thực tế của ngăn xếp và hàng đợi.

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.

D THÔNG TIN BỔ SUNG

Các chú ý và kiến thức sau có tính nâng cao, GV có thể tuỳ theo tình hình học tập trên lớp để đưa thêm vào tiết học, hoặc chuyển thành phần đọc thêm, vận dụng, dành cho HS khá, giỏi.

1. Trong mô tả kiểu dữ liệu ngăn xếp và hàng đợi có khái niệm "dãy tuyến tính các phần tử", khái niệm này có thể hơi khó hiểu đối với HS. GV có thể giải thích thêm từ "tuyến tính" ở đây được hiểu theo nghĩa đơn giản là "thẳng hàng", gần giống với khái niệm "dãy" hay "tuần tự". Tuy nhiên mô tả này không có ý nghĩa quyết định đến việc cài đặt dữ liệu cụ thể cho ngăn xếp và hàng đợi (sẽ học trong Bài 2, Bài 4).

2. Mô hình **Undo** và **Redo** trên thực tế được thiết lập như sau (phần kiến thức này tương đối phức tạp nên đã không đưa vào sách chuyên đề):

Các lệnh (thao tác) trong khi soạn thảo có một số lệnh có tính chất "khả ngược", tức là có lệnh ngược, chẳng hạn lệnh gõ một kí tự, lệnh ngược là xoá kí tự đó. Thao tác xoá một kí tự có lệnh ngược là gõ kí tự đó. Có một số thao tác không có lệnh ngược, chẳng hạn như tìm kiếm, in ấn, phóng to, thu nhỏ,...

Các lệnh Undo và Redo là hai lệnh ngược nhau, được tổ chức như sau trong phần mềm soạn thảo văn bản, mô hình một người sử dụng.

– Bộ nhớ sẽ có hai ngăn xếp, tạm gọi là ngăn xếp Undo và ngăn xếp Redo.

– Trong khi soạn thảo, mỗi khi chúng ta thực hiện một lệnh có lệnh ngược thì lệnh này sẽ được đưa (push) vào ngăn xếp Undo.

– Khi thực hiện lệnh Undo, nếu ngăn xếp Undo không rỗng thì thực hiện lấy ra (pop) lệnh từ ngăn xếp Undo, thực hiện lệnh ngược của lệnh này và đưa (push) lệnh này vào ngăn xếp Redo.

– Khi thực hiện lệnh Redo, nếu ngăn xếp Redo không rỗng thì thực hiện lấy ra (pop) lệnh này từ ngăn xếp Redo, thực hiện lại lệnh này, sau đó đưa (push) lệnh này vào ngăn xếp Undo.

3. Cần chú ý về yêu cầu độ phức tạp thời gian O(1) cho tất cả các thao tác cơ bản trên ngăn xếp (Bảng 1.1 SCĐ) và hàng đợi (Bảng 1.2 SCĐ). Đây là những yêu cầu đặt ra đối với các kiểu dữ liệu này về lí thuyết. Khi tiến hành cài đặt kiểu dữ liệu ngăn xếp hay hàng đợi trên một ngôn ngữ lập trình cụ thể thì cần kiểm tra xem điều kiện về thời gian này có đạt hay không. Mỗi mô hình kiểu dữ liệu, ví dụ ngăn xếp hay hàng đợi, sẽ có nhiều cách cài đặt cụ thể, ví dụ, Python. Các yêu cầu về độ phức tạp thời gian là tương đối khó với HS, GV cần chú ý khi giảng dạy nội dung này.

4. Trong bài học này, HS chỉ được học và nhận biết mô hình ngăn xếp, hàng đợi ở mức mô hình và cơ chế hoạt động mà chưa có phần thực hành, cài đặt trên một ngôn ngữ lập trình cụ thể. Bài 2 và Bài 4 sẽ mô tả chi tiết quá trình cài đặt các kiểu dữ liệu này trên Python.

BÀI 2. KIỂU DỮ LIỆU NGĂN XẾP

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

1. Kiến thức

– Cách thiết lập kiểu dữ liệu ngăn xếp bằng list trong Python.

– Các thao tác cơ bản với ngăn xếp được thực hiện trên kiểu dữ liệu list trong Python.

2. Năng lực

– Biểu diễn được ngăn xếp bằng mảng một chiều.

– Giải thích và viết được chương trình con sử dụng các hàm cơ bản của kiểu dữ liệu ngăn xếp.

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. GV có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2
Dùng kiểu dữ liệu mảng để biểu diễn ngăn xếp.	Tìm hiểu các hàm của kiểu dữ liệu ngăn xếp.

Hoạt động khởi động

Mục đích: Gợi ý, gây tò mò cho HS về các cách khác nhau có thể biểu diễn mô hình ngăn xếp trong ngôn ngữ lập trình bậc cao cụ thể.

GV chia lớp thành ba nhóm và lần lượt, hoặc đồng thời, cho các nhóm trả lời câu hỏi: Có thể dùng kiểu dữ liệu này (ba kiểu đã mô tả trong sách) để biểu diễn mô hình dữ liệu ngăn xếp được không? Nếu có thì các thao tác chính như pop() và push() sẽ được thiết lập như thế nào?

Lưu ý: Trong phần khởi động này không yêu cầu HS phải trả lời đúng và chính xác. Mục đích là đặt ra vấn đề để HS làm quen với bài toán chính sẽ được học trong bài.

Hoạt động 1: Dùng kiểu dữ liệu mảng để biểu diễn ngăn xếp

Mục đích: HS biết và nắm được cách sử dụng kiểu dữ liệu list trong Python để mô tả ngăn xếp thông qua sơ đồ hoặc mô tả bằng lời, chưa cần lập trình.

Tổ chức hoạt động: GV tổ chức hoạt động này theo hai bước.

Bước 1: 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 ngăn xếp.

Tại bước này, thông qua trao đổi nhóm hoặc dưới dạng hỏi đáp, GV hướng dẫn HS nhớ lại một số kiến thức liên quan đến kiểu dữ liệu list. Ví dụ:

– 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 = 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 append(). Ví dụ A.append(x) sẽ bổ sung x vào cuối dãy A.

– Phần tử cuối của list A có chỉ số len(A)−1 hoặc −1.

– Phương thức (hàm) A.pop() sẽ xoá phần tử cuối của A và trả về giá trị phần tử này.

Bước 2: Sử dụng list để biểu diễn ngăn xếp.

Tại bước này, thông qua trao đổi nhóm hoặc dưới dạng hỏi đáp, GV hướng dẫn HS nắm được cách dùng kiểu dữ liệu danh sách (list) của Python để biểu diễn ngăn xếp. 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
push(S, x)	Bổ sung x vào list bằng lệnh append()
pop(S)	Phương thức pop()
top – đỉnh của S	Phần tử cuối của list
bottom – đáy của S	Phần tử đầu tiên của list

GV chốt kiến thức, khẳng định rằng có thể biểu diễn ngăn xếp bằng list của Python.

Lưu ý: GV nhấn mạnh rằng đây chỉ là một cách và không phải là cách duy nhất. Có nhiều cách biểu diễn khác nhưng trong phạm vi Bài 2 chỉ học cách này.

Hoạt động củng cố kiến thức

1. Kết quả thu được trong ngăn xếp S sẽ còn lại dãy: 1, 3, 5.

2. Dãy các số được đưa ra khỏi ngăn xếp lần lượt là các số: 2, 3, 1.

Hoạt động 2: Tìm hiểu các hàm của kiểu dữ liệu ngăn xếp

Mục đích: HS hiểu và thực hiện được việc cài đặt các hàm chính của kiểu dữ liệu ngăn xếp bằng list của Python.

Tổ chức hoạt động: GV chia lớp thành các nhóm, mỗi nhóm được giao nhiệm vụ viết các hàm cơ bản của ngăn xếp trên giấy hoặc trực tiếp trên máy tính, sau đó GV cùng HS phân tích đúng sai và rút ra kết luận. Các hàm cần viết bao gồm:

STT	Hàm	Ý nghĩa hàm
1	S = Stack()	Tạo ngăn xếp rỗng S.
2	push(S, x)	Đưa phần tử x vào đỉnh ngăn xếp S.
3	isEmptyStack(S)	Kiểm tra ngăn xếp S là rỗng hay không.
4	pop(S)	Lấy ra phần tử ở đỉnh ngăn xếp.
5	top(S)	Trả về giá trị phần tử ở đỉnh ngăn xếp.

GV chốt kiến thức.

Hoạt động củng cố kiến thức

1. Sửa lại các hàm pop() và top() theo yêu cầu như sau:

Hàm pop(S)

```
1     def pop(S):
2         if isEmptyStack(S):
3             print("Ngăn xếp rỗng không thể thực hiện được lệnh này.")
4         else:
5             return S.pop()
```

Hàm top(S)

```
1     def top(S):
2         if isEmptyStack(S):
3             print("Ngăn xếp rỗng không thể thực hiện được lệnh này.")
4         else:
5             return S[-1]
```

2. Khi sử dụng list để biểu diễn ngăn xếp thì ta luôn có chỉ số phần tử ở đáy là 0, chỉ số phần tử ở đỉnh là len(S)−1, do đó không cần các biến top hay bottom.

Hoạt động luyện tập

1. Hàm length(S) có thể viết như sau:

```
1     def length(S):
2         return len(S)
```

2. Các lệnh đã thực hiện lần lượt là:

```
push(S,2)
push(S,7)
push(S,6)
pop(S)
pop(S)
push(S,1)
pop(S)
pop()
```

Hoạt động vận dụng

1. Hàm kiểm tra xâu **bt** chỉ bao gồm các kí 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.

```
1     from Stack import *
2     def CheckBT(bt):
3         S = Stack()
4         kq = True
5         for ch in bt:
6             if ch == "(":
7                 push(S,ch)
8             if ch == ")":
9                 if isEmptyStack(S):
10                    kq = False
11                    break
12                else:
13                    pop(S)
14        return kq and isEmptyStack(S)
```

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:

Stack_segment.py

```
# Thiết kế ngăn xếp sử dụng mảng độ dài cố định N
# Ngăn xếp có 2 thuộc tính là mảng T cho trước độ dài N và biến topIdx
# Khai báo các biến chung mô tả ngăn xếp: N, T, topIdx
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, không thể thực hiện được.")
    else:
        topIdx = topIdx + 1
        S[topIdx] = x

def pop(S):
    global topIdx
    if isEmptyStack(S):
        raise ValueError("Ngăn xếp rỗng, không thể thực hiện được.")
    else:
        x = S[topIdx]
        topIdx = topIdx - 1
        return x

def top(S):
    if isEmptyStack(S):
        raise ValueError("Ngăn xếp rỗng, không thể thực hiện được.")
    else:
        return S[topIdx]
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

D THÔNG TIN BỔ SUNG

Với cách thiết lập ngăn xếp sử dụng list của Python như trong bài học này thì độ phức tạp thời gian của các hàm cơ bản được cho trong bảng sau:

STT	Hàm	Độ phức tạp thời gian
1	S = Stack()	O(1)
2	push(S, x)	O(1)
3	isEmptyStack(S)	O(1)
4	pop(S)	O(1)
5	top(S)	O(1)

Như vậy mô hình cài đặt này của ngăn xếp thoả mãn tất cả các điều kiện về độ phức tạp thời gian của mô hình kiểu dữ liệu ngăn xếp.

BÀI 3. THỰC HÀNH KIỂU DỮ LIỆU NGĂN XẾP

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

1. Kiến thức

Cách sử dụng cấu trúc dữ liệu ngăn xếp.

2. Năng lực

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

Hình thành tư duy lôgic phân tích yêu cầu cần đạt để thực hiện một nhiệm vụ cụ thể.

B YÊU CẦU CHUẨN BỊ

GV có thể thực hiện bài giảng tuỳ theo điều kiện vật chất cụ thể của nhà trường. Trừ các tiết học thực hành HS bắt buộc học và thực hành trên phòng máy, còn tiết lí thuyết thì có nhiều lựa chọn, ví dụ:

– Nếu lớp học không có máy tính thì GV sẽ dạy hoàn toàn trên bảng (viết lệnh mà không cần máy tính).

– Nếu GV có máy tính và máy chiếu thì có thể giảng dạy và thực hiện lệnh trực tiếp trên máy tính, chiếu cho cả lớp theo dõi.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động khởi động

Mục đích: GV nhắc lại các kiến thức đã học về kiểu dữ liệu ngăn xếp, lưu ý cho HS khi nào nên sử dụng kiểu dữ liệu ngăn xếp.

Hoạt động thực hành

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

Tổ chức hoạt động:

– GV giao HS đọc và phân tích yêu cầu bài toán trước.

– Sau khoảng 2 phút, gọi một số HS đưa ra nhận xét và GV tổng kết câu trả lời.

– GV có thể chuẩn bị một số thẻ giấy ghi lại giá trị các dãy số trong dãy và yêu cầu một số HS mô phỏng lại quá trình sắp xếp dãy số bằng cách sử dụng thẻ giấy.

– GV hướng dẫn HS viết chương trình và chạy chương trình mô phỏng quá trình duyệt web.

Hoạt động thực hành

Nhiệm vụ 2: Viết chương trình kiểm tra các dấu ngoặc trong biểu thức

Tổ chức hoạt động:

– GV giao HS đọc và phân tích yêu cầu bài toán trước.

– Sau khoảng 2 phút, gọi một số HS đưa ra nhận xét và GV tổng kết câu trả lời. Có thể đưa ra thêm một số ví dụ về các biểu thức đúng, biểu thức sai, sau đó yêu cầu HS sửa các biểu thức sai thành biểu thức đúng.

– GV hướng dẫn HS viết chương trình và chạy chương trình.

Hoạt động luyện tập

Mục đích: GV hướng dẫn HS thực hiện các yêu cầu trong phần luyện tập để đảm bảo HS thực sự hiểu được chương trình của Nhiệm vụ 1 và Nhiệm vụ 2 chứ không dừng lại ở mức sao chép và chạy chương trình.

1. Sửa lại chương trình trong Nhiệm vụ 1 để thêm chức năng đi đến trang web kế tiếp.

Để làm được yêu cầu này, chúng ta cần phải sử dụng thêm 1 ngăn xếp "forward" để lưu trữ địa chỉ website trước khi 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 tuỳ chọn 1 để nhập địa chỉ mới, tuỳ chọn 2 để go backward,

tuỳ chọn 3 để go forward và tuỳ chọn 4 để dừng chương trình. Nhiệm vụ này có thể được giải quyết như sau:

web.py

```
from Stack import *
backward = Stack()
forward = Stack()
current_web =""
option = 0
while option!= 4:
    option = int(input("Hãy nhập vào lựa chọn của bạn:\n"))
    if option ==1:
        website = input("Hãy nhập vào địa chỉ website muốn đi đến:\n")
        if (current_web!= ""):
            push(backward,current_web)
        print("Đang đi đến trang web:"+ website)
        current_web = website
    if option ==2:
        if isEmptyStack(backward):
            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 web:"+ current_web)
```

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

Trong bài toán này, chúng ta chỉ in ra kết quả là 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 *tong_tron* để đếm số lượng ngoặc '(' và *tong_vuong* để đếm số lượng ngoặc '['. Chương trình trong Nhiệm vụ 2 có thể được sửa lại như sau:

kiemtrabieuthuc.py

```
from Stack import *
def kiemtrabt(bieuthuc, tong_tron, tong_vuong):
    hople = True
    ngoacmo= Stack()
    for i in range(0, len(bieuthuc)):
        if bieuthuc[i] in {"(","["}:
            push(ngoacmo,bieuthuc[i])
            if bieuthuc[i]=="(":
                tong_tron = tong_tron + 1
            else :
                tong_vuong = tong_vuong + 1
        elif bieuthuc[i] in {")","]"}:
            tmp = pop(ngoacmo)
          if (bieuthuc[i] == ")" and tmp!="(") or (bieuthuc[i] == "]" and tmp!="["):
                hople = False
                break
    if not isEmptyStack(ngoacmo):
        hople = False
    return(hople,tong_tron, tong_vuong)
bieuthuc = input("Hãy nhập vào một biểu thức:\n")
tong_tron = 0
tong_vuong = 0
hople, tong_tron, tong_vuong = kiemtrabt(bieuthuc, tong_tron, tong_vuong)
if hople == True :
    print("Biểu thức hợp lệ")
    print("Tổng số cặp ngoặc tròn là: ", tong_tron)
    print("Tổng số cặp ngoặc vuông là: ", tong_vuong)
else:
    print("Biểu thức không hợp lệ")
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

Hoạt động vận dụng

1. Bài toán sắp xếp và lấy sách ra khỏi ngăn tủ có thể thực hiện như sau:

Bước 1: Đọc dữ liệu từ tệp đầu vào (file input) chứa tên các quyển sách. Ví dụ nội dung của tệp đầu vào *sach.inp* như sau:

```
English Physic Maths Chemistry History Biology
```

Bước 2: Khởi tạo kiểu dữ liệu ngăn xếp, với mỗi quyển sách đọc từ tệp đầu vào, sử dụng hàm *push* để đưa vào ngăn xếp.

Bước 3: Cho phép người dùng nhập tên một quyển sách cần lấy. Lần lượt lấy các quyển sách ra khỏi ngăn xếp bằng cách sử dụng hàm *pop* cho đến khi lấy được quyển sách cần lấy. Đồng thời tiến hành đếm số quyển sách được lấy ra khỏi ngăn.

Bước 4: In ra kết quả số quyển sách được lấy ra trước khi lấy được quyển sách cần lấy.

Chương trình có thể như sau:

tusach.py

```
from Stack import *
input_file = open("sach.inp","r")
data = input_file.read()
tusach = Stack()
for sach in data.split(' '): # tên sách được ngăn cách nhau bởi dấu cách
    push(tusach,sach)
input_file.close()
sach = input("Hãy nhập vào quyển sách cần lấy:\n")
thanhcong = 0
count = 0
while not isEmptyStack(tusach):
        tmp = pop(tusach)
        if tmp == sach:
            thanhcong = 1
            break
        count = count + 1
if thanhcong == 0:
    print("Không tìm được sách cần lấy trong ngăn tủ")
else:
    print(" Số quyển sách cần lấy ra trước là: ", count)
```

2. Bài toán kiểm tra tính hợp lệ của biểu thức với ba loại dấu đóng, mở ngoặc.

Chương trình này tương tự chương trình ở Nhiệm vụ 2: HS cần chú ý sửa chương trình để kiểm tra thêm loại dấu ngoặc thứ ba. Chương trình được sửa lại có thể như sau:

kiemtrabieuthuc.py

```
from Stack import *
def kiemtrabt(bieuthuc):
    hople = True
    ngoacmo= Stack()
    for i in range(0, len(bieuthuc)):
        if bieuthuc[i] in {"(","[", "{"}:
            push (ngoacmo,bieuthuc[i])
        elif bieuthuc[i] in {")","]"}:
```

```
9                   tmp = pop(ngoacmo)
10                  if (bieuthuc[i] == ")" and tmp!="(") or (bieuthuc[i] == "]" and
    tmp!="[") or (bieuthuc[i] == "}" and tmp!="{"):
11                      hople = False
12                      break
13          if not isEmptyStack(ngoacmo):
14              hople = False
15          return hople
16      bieuthuc = input("Hãy nhập vào một biểu thức:\n")
17      hople = kiemtrabt(bieuthuc)
18      if hople == True :
19          print("Biểu thức hợp lệ")
20      else:
21          print("Biểu thức không hợp lệ")
```

D THÔNG TIN BỔ SUNG

Khi sử dụng dữ liệu list trong Python làm ngăn xếp thì các thao tác cơ bản trên ngăn xếp có thể được mô tả bằng các lệnh hoặc phương thức đơn giản đối với list như sau:

STT	Thao tác chuẩn với ngăn xếp	Phương thức trên list tương ứng
1	S = Stack()	S = []
2	isEmptyStack(S)	S == [] hoặc len(S) == 0
3	top(S)	S[-1] hoặc S[len(S)-1]
4	pop(S)	S.pop()
5	push(S, x)	S.append(x)

BÀI 4. KIỂU DỮ LIỆU HÀNG ĐỢI

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

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

– Biểu diễn được hàng đợi bằng mảng một chiều.

– Giải thích và viết được các chương trình con sử dụng các hàm cơ bản của kiểu dữ liệu hàng đợi.

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. GV có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2
Dùng kiểu dữ liệu mảng để biểu diễn hàng đợi.	Tìm hiểu các hàm của kiểu dữ liệu hàng đợi.

Hoạt động khởi động

Mục đích: Gợi ý, gây tò mò cho HS về các cách khác nhau có thể biểu diễn mô hình hàng đợi trong ngôn ngữ lập trình bậc cao.

GV chia lớp thành các nhóm và lần lượt, hoặc đồng thời, cho các nhóm trả lời câu hỏi: Có thể dùng kiểu dữ liệu này (hai kiểu đã mô tả trong sách) để biểu diễn mô hình dữ liệu hàng đợi được không? Nếu có thì các thao tác chính như enqueue() và dequeue() sẽ được thiết lập như thế nào?

Chú ý trong phần khởi động này không yêu cầu HS phải trả lời đúng và chính xác. Mục đích là đặt ra vấn đề để HS làm quen với bài toán chính sẽ được học trong bài học.

Hoạt động 1: Dùng kiểu dữ liệu mảng để biểu diễn hàng đợi

Mục đích: HS biết và nắm được cách sử dụng kiểu dữ liệu list của Python để mô tả hàng đợi thông qua sơ đồ hoặc mô tả bằng lời, chưa cần lập trình.

Tổ chức hoạt động: GV tổ chức hoạt động này theo hai bước.

Bước 1: 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.

Tại bước này, thông qua trao đổi nhóm hoặc dưới dạng hỏi đáp, GV hướng dẫn HS nhớ lại một số kiến thức liên quan đến kiểu dữ liệu list. Ví dụ:

– 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 = 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 append(). Ví dụ A.append(x) sẽ bổ sung x vào cuối dãy A.

– Phần tử cuối của list A có chỉ số len(A)−1 hoặc −1.

– Phương thức (hàm) A.pop(0) sẽ xoá phần tử đầu của A và trả về giá trị phần tử này.

Bước 2: Sử dụng list để biểu diễn hàng đợi.

Tại bước này, thông qua trao đổi nhóm hoặc dưới dạng hỏi đáp, GV cần hướng dẫn HS nắm được cách dùng kiểu dữ liệu danh sách (list) của Python để biểu diễn hàng đợi. Các thao tác sau trên hàng đợi cần được mô tả bằng lệnh tương ứng trên list của Python.

Thao tác trên hàng đợi	**Lệnh tương ứng trên list của Python**
Tạo hàng đợi rỗng	Tạo list rỗng.
enqueue(Q,x)	Bổ sung x vào list bằng lệnh append().
dequeue(Q)	Phương thức pop(0).
front – đầu của Q	Phần tử đầu của list.
back, rear, tail – đuôi của Q	Phần tử cuối cùng của list.

GV chốt kiến thức, khẳng định rằng có thể biểu diễn hàng đợi bằng list của Python.

Lưu ý: GV nhấn mạnh rằng đây chỉ là một cách và không phải là cách duy nhất. Có rất nhiều cách biểu diễn khác nhưng trong phạm vi Bài 4 chỉ dùng cách này.

Hoạt động củng cố kiến thức

1. Số phần tử của hàng đợi Q chính là giá trị len(Q).

2. Sau khi thực hiện tuần tự các phép toán enqueue(Q, 2); enqueue(Q, 10); dequeue(Q); enqueue(Q, 6); dequeue(Q); enqueue(Q, 9); enqueue(Q, 1) thì phần tử đầu của hàng đợi Q là: 6, phần tử đuôi là 1.

Hoạt động 2: Tìm hiểu các hàm của kiểu dữ liệu hàng đợi

Mục đích: HS hiểu và thực hiện được cài đặt các hàm chính của kiểu dữ liệu hàng đợi bằng list của Python.

Tổ chức hoạt động: GV chia lớp thành các nhóm, mỗi nhóm được giao nhiệm vụ viết các hàm cơ bản của hàng đợi trên giấy hoặc trực tiếp trên máy tính, sau đó GV cùng HS phân tích đúng sai và rút ra kết luận. Các hàm cần viết bao gồm:

STT	Hàm	Ý nghĩa hàm
1	Q = Queue()	Tạo hàng đợi rỗng Q.
2	enqueue(Q, x)	Đưa phần tử x vào đuôi của hàng đợi Q.
3	isEmptyQueue(Q)	Kiểm tra hàng đợi Q là rỗng hay không.
4	dequeue(Q)	Lấy ra phần tử ở đầu hàng đợi.
5	front(Q)	Trả về giá trị phần tử ở đầu hàng đợi.

GV chốt kiến thức.

Hoạt động củng cố kiến thức

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.

So sánh với ngăn xếp được biểu diễn bằng mảng (list) của Python ta có:

Chỉ số đầu (front) của hàng đợi chính là chỉ số đáy (bottom) của ngăn xếp.

Chỉ số đuôi (back, rear, tail) của hàng đợi là chỉ số đỉnh (top) của ngăn xếp.

2. Bảng sau mô tả sự giống nhau và khác nhau giữa các hàm cơ bản của ngăn xếp và hàng đợi khi các kiểu dữ liệu này được biểu diễn bằng danh sách (list) của Python.

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: giá trị phần tử cuối cùng. Hàng đợi: giá trị phần tử đầu tiên.

Hoạt động luyện tập

1. Sửa lại hàm dequeue(Q) và front(Q) như sau:

Hàm dequeue(Q):

```
1       def dequeue(Q):
2           if isEmptyQueue(Q):
3               print("Hàng đợi rỗng không thể thực hiện được lệnh này.")
4           else:
5               return Q.pop(0)
```

Hàm front(Q):

```
1       def front(Q):
2           if isEmptyQueue(Q):
```

```
            print("Hàng đợi rỗng không thể thực hiện được lệnh này.")
        else:
            return Q[0]
```

2. Hàm length(Q) có thể viết như sau:

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

Hoạt động vận dụng

1. Lệnh dequeue(Q) chính là phương thức Q.pop(0), hàm này tương đương với đoạn chương trình sau bao gồm một vòng lặp do vậy có độ phức tạp thời gian O(n).

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

2. Chương trình thiết lập kiểu dữ liệu hàng đợi dựa trên mảng dữ liệu cho trước có độ dài cố định N có thể như sau:

Queue_segment.py

```
# Thiết lập mô hình hàng đợi tổng quát từ mảng một chiều với độ dài cố định N
# Hàng đợi có 2 thuộc tính là mảng T và biến backIdx
# Các biến chung mô tả hàng đợi: N,T,backIdx
N = 100
T = [None]*N
backIdx = -1

def Queue():
    global backIdx
    backIdx = -1
    return T

def isEmptyQueue(Q):
    return backIdx == -1

def isFullQueue(Q):
    return backIdx == N-1

def enqueue(Q,x):
    global backIdx
    if isFullQueue(Q):
        raise ValueError("Hàng đợi đầy, không thể thực hiện được.")
    else:
        backIdx = backIdx + 1
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
25              Q[backIdx] = x
26
27      def dequeue(Q):
28          global backIdx
29          if isEmptyQueue(Q):
30              raise ValueError("Hàng đợi rỗng, không thể thực hiện được.")
31          else:
32              x = Q[0]
33              backIdx = backIdx - 1
34              Q.pop(0)
35              return x
36
37      def front(Q):
38          if isEmptyQueue(Q):
39              raise ValueError("Hàng đợi rỗng, không thể thực hiện được.")
40          else:
41              return Q[0]
```

D THÔNG TIN BỔ SUNG

Với cách thiết lập hàng đợi sử dụng list của Python thì độ phức tạp thời gian của các hàm cơ bản được cho trong bảng sau:

STT	Hàm	Độ phức tạp thời gian
1	Q = Queue()	O(1)
2	enqueue(Q,x)	O(1)
3	isEmptyQueue(Q)	O(1)
4	dequeue(Q)	O(n)
5	front(Q)	O(1)

Như vậy mô hình cài đặt này của hàng đợi không thoả mãn tất cả các điều kiện về thời gian của mô hình kiểu dữ liệu hàng đợi. Tất cả các hàm/thao tác cơ bản trên đều có độ phức tạp thời gian O(1), trừ ra lệnh dequeue(Q) có thời gian O(n).

Do vậy việc biểu diễn, cài đặt hàng đợi bằng list của Python theo cách đã mô tả trong SCĐ là chưa thật tối ưu. Vấn đề đặt ra là có thể có một cách cài đặt khác sao cho tất cả các hàm cơ bản trên đều có thời gian chạy hằng số, tức là O(1) được không. Đây là một vấn đề hay, GV có thể đưa ra cho đối tượng HS khá, giỏi hoặc các lớp chuyên để HS nghiên cứu tìm ra cách cài đặt khác cho kiểu dữ liệu hàng đợi thoả mãn tất cả các yêu cầu về độ phức tạp thời gian của các thao tác cơ bản trên hàng đợi là O(1).

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

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

1. Kiến thức

Cách sử dụng cấu trúc dữ liệu ngăn xếp và hàng đợi.

2. Năng lực

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

Hình thành tư duy lôgic phân tích yêu cầu cần đạt để thực hiện một nhiệm vụ cụ thể.

B YÊU CẦU CHUẨN BỊ

GV có thể thực hiện bài giảng tuỳ theo điều kiện vật chất cụ thể của nhà trường. Trừ các tiết học thực hành HS bắt buộc học và thực hành trên phòng máy, còn tiết lí thuyết thì có nhiều lựa chọn, ví dụ:

– Nếu lớp học không có máy tính thì GV sẽ dạy hoàn toàn trên bảng (viết lệnh mà không cần máy tính).

– Nếu GV có máy tính và máy chiếu thì có thể giảng dạy và thực hiện lệnh trực tiếp trên máy tính, chiếu cho cả lớp theo dõi.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động khởi động

Mục đích: GV nhắc lại các kiến thức đã học về kiểu dữ liệu ngăn xếp và hàng đợi, lưu ý cho HS khi nào nên sử dụng kiểu dữ liệu ngăn xếp, khi nào nên sử dụng kiểu dữ liệu hàng đợi.

Hoạt động thực hành: Viết chương trình mô phỏng bếp ăn tập thể

Tổ chức hoạt động:

– GV giao HS đọc và phân tích yêu cầu bài toán trước.

– GV có thể chia HS thành các nhóm để thảo luận nhóm, sau đó gọi một số nhóm lên trình bày phương án của mình.

– GV cho HS mô phỏng lại việc chuẩn bị đồ ăn của bếp ăn và cho một số HS tiến hành đăng kí món ăn, xếp hàng để nhận suất ăn.

– GV hướng dẫn HS viết và chạy chương trình mô phỏng bếp ăn.

Hoạt động luyện tập

Mục đích: GV hướng dẫn HS thực hiện các yêu cầu trong phần luyện tập để đảm bảo HS thực sự hiểu được chương trình của nhiệm vụ thực hành chứ không dừng lại ở mức sao chép và chạy chương trình.

1. 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 tuỳ theo số suất mà bếp ăn đã chuẩn bị.

2. Tính toán thời gian trung bình để nhận được suất ăn của mỗi người. Để làm được bài toán này, chúng ta chú ý quy luật như sau: thời gian chờ đợi trung bình của một người để nhận được suất ăn chính là thời gian trung bình một suất ăn được đưa đến tay người lao động, trong đó:

– Thời gian để suất ăn đầu tiên đến tay người lao động là 1 giây.

– Thời gian để suất ăn đầu tiên đến tay người lao động là 2 giây,...

Yêu cầu này được thực hiện như sau:

thoigian.py

```
from Stack import *
suatan  = Stack()
file2 = open("input2.inp", encoding ="utf8")
data = file2.read()
for item in data.split(' '): # đọc thông tin suất ăn
    push(suatan,item) # đưa suất ăn vào ngăn xếp
tong_thoi_gian = 0 # tổng thời gian chờ đợi của mỗi suất ăn
so_suat_an = 0
while not isEmptyStack(suatan): #lấy suất ăn ra khỏi ngăn xếp
    tmp = pop(suatan)
    so_suat_an = so_suat_an + 1
    tong_thoi_gian = tong_thoi_gian + so_suat_an
print("Thời gian chờ đợi trung bình của mỗi người là:
",tong_thoi_gian/so_suat_an)
```

Hoạt động vận dụng

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:

bepan.py

```
1       from Queue import *
2       from Stack import *
3       def bepan(dangki, suatan):
4           while not isEmptyStack(suatan):
5               tmp1 = pop(suatan) #lấy suất ăn ra khỏi ngăn xếp
6               phat_thanh_cong = 0
7               for i in range(0,len(dangki)):
8                   tmp2 = dequeue(dangki)
9                   if tmp1 == tmp2: # người dùng lấy được suất ăn
10                      phat_thanh_cong = 1
11                      break
12                  else:
13                      enqueue(dangki,tmp2)
14              if phat_thanh_cong == 0: # không phát được suất ăn
15                  break
16          return len(dangki)
17      dangki = Queue()
18      suatan  = Stack()
19      file1= open("input1.inp",encoding="utf8")
20      data1 = file1.read()
21      for item in data1.split(' '): # đọc thông tin suất ăn
22          enqueue(dangki,item)
23      file2 = open("input2.inp", encoding ="utf8")
24      file1.close()
25      data2 = file2.read()
26      for item in data2.split(' '): # đọc thông tin suất ăn
27          push(suatan,item)
28      file2.close()
29      print("Số lượng người không nhận được suất ăn là: ",bepan(dangki, suatan))
```

D THÔNG TIN BỔ SUNG

Khi sử dụng dữ liệu list trong Python làm hàng đợi thì các thao tác cơ bản trên hàng đợi có thể được mô tả bằng các lệnh hoặc phương thức đơn giản đối với list như sau:

STT	Thao tác chuẩn với hàng đợi	Phương thức trên list tương ứng
1	Q = Stack()	Q = []
2	isEmptyQueue(Q)	Q == [] hoặc len(Q) == 0
3	front(Q)	Q[0]
4	back(Q)	Q[-1] hoặc Q[len(Q)-1]
4	dequeue(Q)	Q.pop(0)
5	enqueue(Q,x)	Q.append(x)

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

BÀI 6. CÂY NHỊ PHÂN

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. GV có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2	Hoạt động 3
Tìm hiểu cấu trúc cây và cây nhị phân.	Tìm hiểu cây nhị phân và cách biểu diễn cây nhị phân.	Tìm hiểu một số thuật toán duyệt cây nhị phân.

Hoạt động khởi động

Mục đích: HS nhận ra được một mô hình dữ liệu mới khác với kiểu dữ liệu mảng đã học.

GV chia lớp thành các nhóm và yêu cầu quan sát các sơ đồ biểu diễn thông tin, sau đó trao đổi, thảo luận và cùng trả lời các câu hỏi, ví dụ:

– Em có nhận xét gì về mô hình dữ liệu này? Mô hình này có gì khác so với kiểu dữ liệu mảng đã biết?

– Nêu các đặc thù hay đặc điểm chung giữa các mô hình này.

– Hãy nêu một số ví dụ thực tế khác có mô hình tương tự.

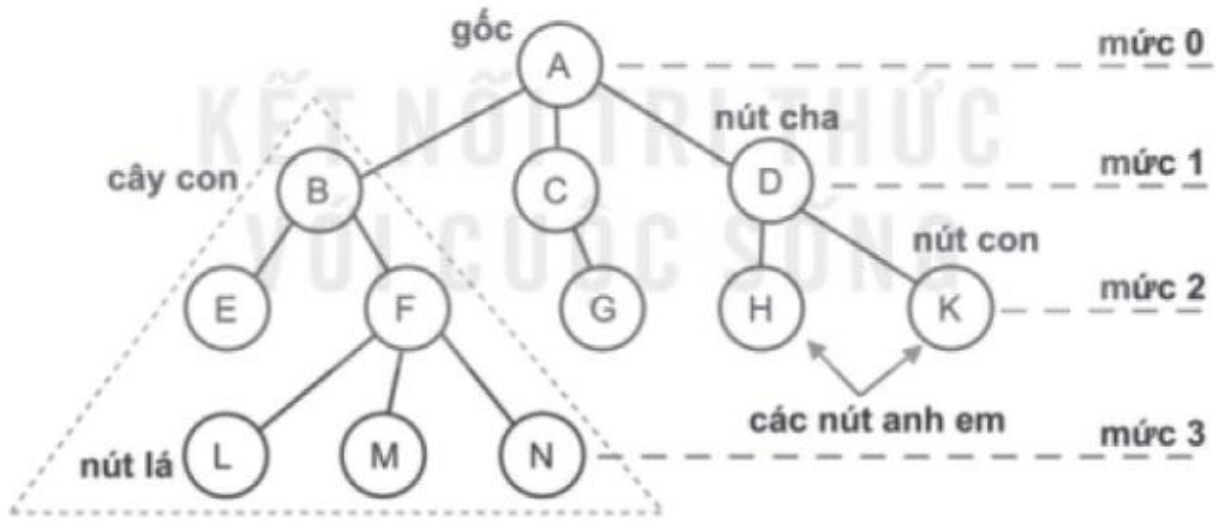
Chú ý không yêu cầu HS phải trả lời đúng các câu hỏi này. Mục đích là gợi sự tò mò của HS với một mô hình dữ liệu mới chưa được học bao giờ.

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

Mục đích: HS nắm được khái niệm và một số định nghĩa của cấu trúc cây tổng quát.

Tổ chức hoạt động: GV có thể tổ chức hoạt động này như sau:

Bước 1: GV trực tiếp giới thiệu hoặc yêu cầu một nhóm HS lên giới thiệu bằng sơ đồ các khái niệm, định nghĩa chính liên quan đến cấu trúc cây. Các khái niệm cần giới thiệu được mô tả trong sơ đồ như Hình 6.2 SCĐ:



Hình 6.1. Mô hình cây (Hình 6.2 SCĐ)

Bước 2: GV chia lớp thành các nhóm, mỗi nhóm có nhiệm vụ giới thiệu, so sánh và chỉ ra các khái niệm chính của cây tương ứng với các ví dụ mô hình cây đã biết trong Hoạt động khởi động. Mỗi nhóm sẽ trình bày theo một ví dụ cụ thể. Mỗi nhóm cần chỉ ra trên ví dụ đó:

– 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.

Cuối cùng GV chốt kiến thức.

Hoạt động củng cố kiến thức

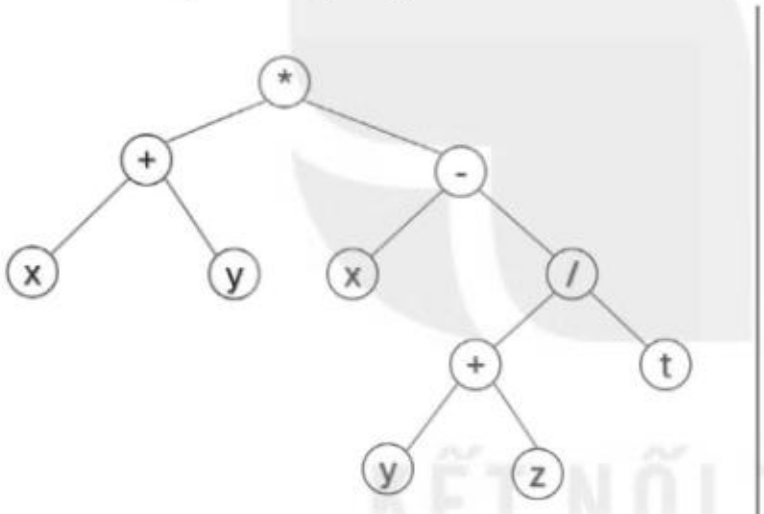
1. Một số ví dụ khác của cây có thể là:

– Mô hình gia phả dòng họ.

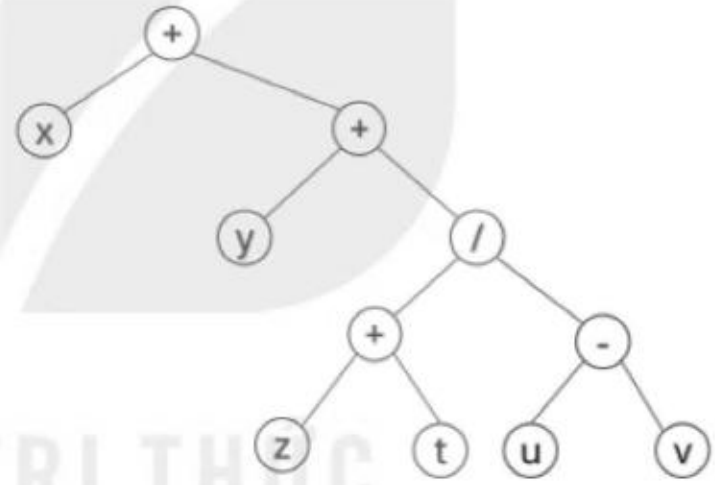
– Mô hình tổ chức của cơ quan.

– Mô hình các chương, mục, bài học của một cuốn sách.

2. Sơ đồ cây tương ứng như sau:



a) (x + y) * (x – (y + z)/t)



b) x + (y + (z + t)/(u – v))

3. Chiều cao của cây: a) 4; b) 5.

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

Mục đích: 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.

Tổ chức hoạt động: Hoạt động này sẽ bao gồm hai bước.

Bước 1: Phân loại, phân biệt cây nhị phân hoàn hảo và hoàn chỉnh.

Có thể hiểu đơn giản: 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.

Tại bước này, cách tốt nhất là GV trực tiếp giới thiệu và giải thích kĩ cho HS hiểu để phân biệt được giữa cây nhị phân hoàn chỉnh và không hoàn chỉnh.

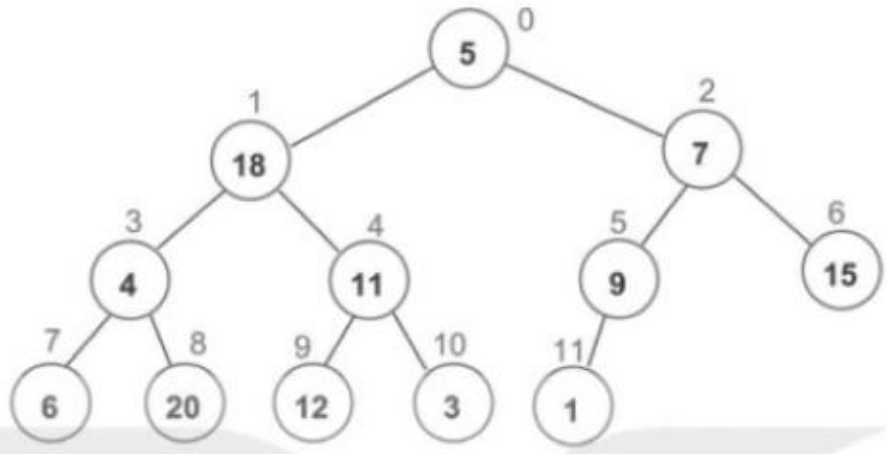
KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

Bước 2: Tìm hiểu cách biểu diễn cây nhị phân hoàn chỉnh bằng mảng một chiều.

Tại bước này GV có thể chia lớp thành các nhóm và mỗi nhóm sẽ thực hiện việc biểu diễn cây nhị phân hoàn chỉnh với mảng một chiều. Các nhóm có thể trình bày biểu diễn này theo hai cách:

Cách 1: Cho trước cây nhị phân hoàn chỉnh, chỉ ra mảng một chiều tương ứng.

Cách 2: Cho trước mảng một chiều, cần vẽ cây nhị phân hoàn chỉnh tương ứng.



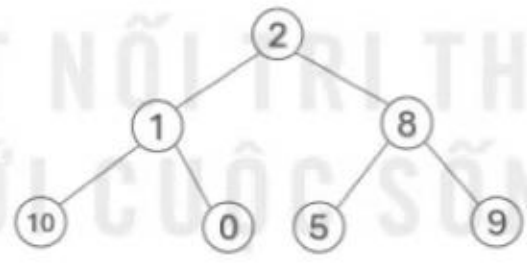
Hình 6.2. Đánh chỉ số cây nhị phân hoàn chỉnh (Hình 6.6 SCĐ)

Cuối cùng GV chốt kiến thức.

Lưu ý: Phần lưu ý của SCĐ về cây nhị phân hoàn chỉnh đã biến đổi có thể bỏ qua không cần giới thiệu cho HS, đây là phần kiến thức nâng cao sẽ được học trong bài tiếp theo.

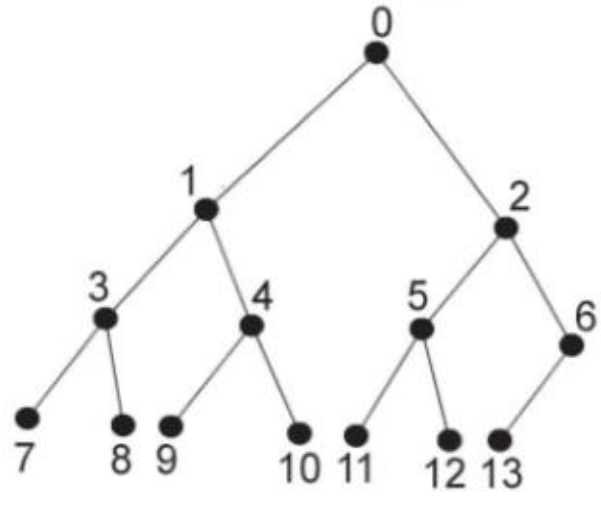
Hoạt động củng cố kiến thức

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.

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



Cây nhị phân này có chiều cao bằng 3.

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

Mục đích: HS biết và thực hiện thủ công và bằng lập trình các thuật toán duyệt trước, duyệt giữa và duyệt sau trên cây nhị phân hoàn chỉnh.

Tổ chức hoạt động: GV chia lớp thành các nhóm, 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.

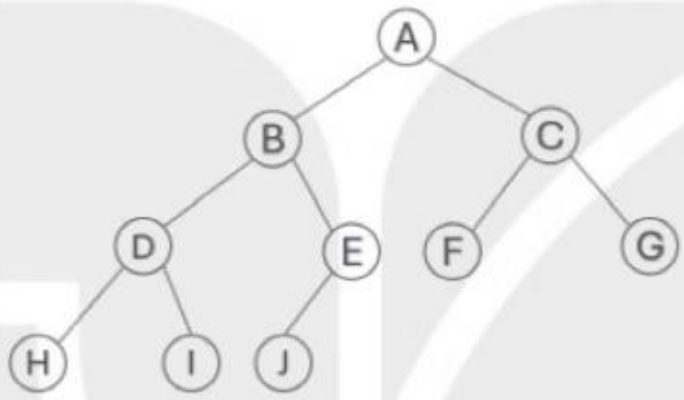
– Mô tả cách duyệt trên một ví dụ cụ thể.

– Viết đoạn chương trình mô tả thuật toán duyệt,

Cuối cùng GV chốt lại ba thuật toán duyệt chính đã được học của bài học.

Hoạt động củng cố kiến thức

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 ở Câu 1 như sau: H, I, D, J, E, B, F, G, C, A.

Hoạt động luyện tập

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

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 Duyet_truoc(), Duyet_giua() và Duyet_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 60 của thế kỉ XX 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.

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

– 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 lí 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 (Interative Dichotomister) 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	– 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.	– 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	– Đơn giản. – Chèn và xoá dữ liệu dễ dàng. – Tiết kiệm bộ nhớ.	– Tìm kiếm chậm: O(n). – Sắp xếp chậm: $O(n^2)$. – Không lưu trữ dữ liệu có thứ tự.
Ngăn xếp	– Cơ chế 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.	– Không hỗ trợ truy cập ngẫu nhiên. – Không hiệu quả cho các ứng dụng cần truy cập dữ liệu theo thứ tự.
Hàng đợi	– Cơ chế FIFO. – Dễ dàng thực hiện các thao tác enqueue và dequeue. – Hiệu quả cho các ứng dụng cần truy cập dữ liệu theo thứ tự.	– Không hỗ trợ truy cập ngẫu nhiên. – 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

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. GV có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2	Hoạt động 3
Tìm hiểu cấu trúc cây tìm kiếm nhị phân.	Thuật toán chèn khoá mới vào cây tìm kiếm nhị phân.	Tìm hiểu thuật toán tìm kiếm trên cây tìm kiếm nhị phân.

Hoạt động khởi động

Mục đích: HS nhận biết được một dạng đặc biệt của cây nhị phân: Cây tìm kiếm nhị phân (BST – Binary Search Tree).

GV gợi ý, hướng dẫn HS quan sát và nhận ra quy luật của quan hệ giữa các khoá trong mô hình cây nhị phân trong SCĐ. Chú ý không yêu cầu HS trả lời chính xác hoàn toàn.

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

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

Tổ chức hoạt động: GV chia lớp thành các nhóm. Các nhóm tiến hành trao đổi, đọc, thảo luận dưới sự hướng dẫn của GV để từng bước nắm bắt kiến thức, kĩ năng chính của hoạt động này.

Hoạt động này được chia làm hai bước với các mục đích khác nhau.

Bước 1: Biểu diễn của cây nhị phân tổng quát. Có hai cách biểu diễn cây nhị phân tổng quát.

Cách 1: Sử dụng cấu trúc nút liên kết. Cách biểu diễn này sẽ cần hai cấu trúc:

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.

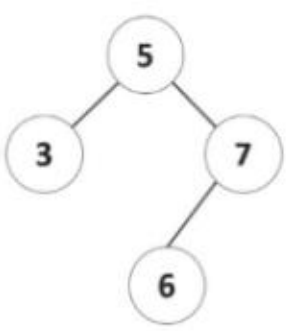


a) Cấu trúc Node *b) Cấu trúc Tree*

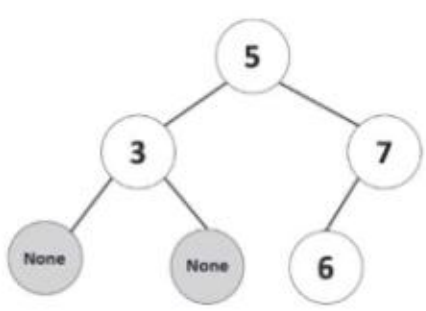
Hình 7.1. Mô hình nút liên kết cây nhị phân (Hình 7.2 SCĐ)

Cách 2: Sử dụng mảng một chiều.

Với cách này, 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 này (đã học ở Bài 6) để biểu diễn.



Cây nhị phân tổng quát



Bổ sung nút giả None để thành cây nhị phân hoàn chỉnh

[5, 3, 7, None, None, 6]

Biểu diễn bằng mảng một chiều

Bước 2: Định nghĩa cây tìm kiếm nhị phân và biểu diễn cây tìm kiếm nhị phân.

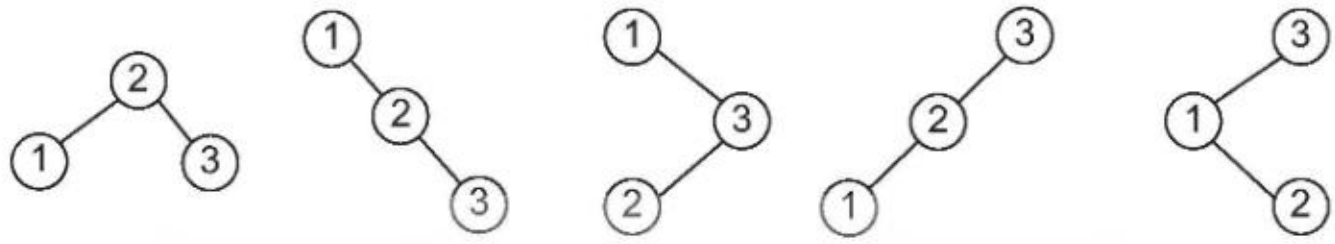
Tại bước này, GV đưa ra định nghĩa của cây tìm kiếm nhị phân. Thời gian còn lại dành cho các nhóm HS trao đổi và nhận biết được sự khác biệt giữa cây tìm kiếm nhị phân và cây nhị phân tổng quát.

Cuối cùng GV chốt kiến thức.

Hoạt động củng cố kiến thức

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.



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

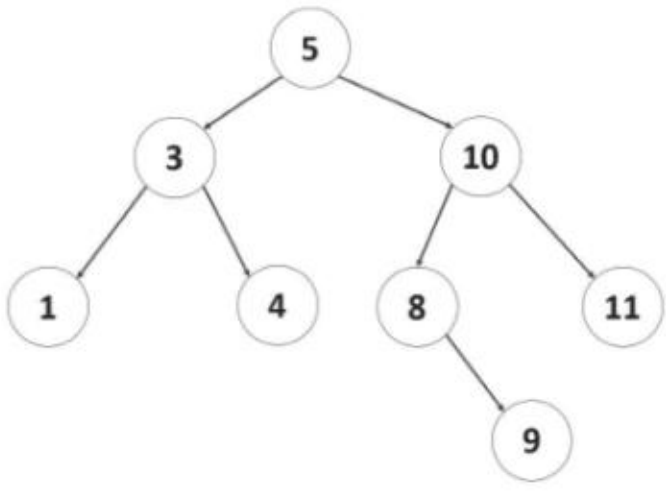
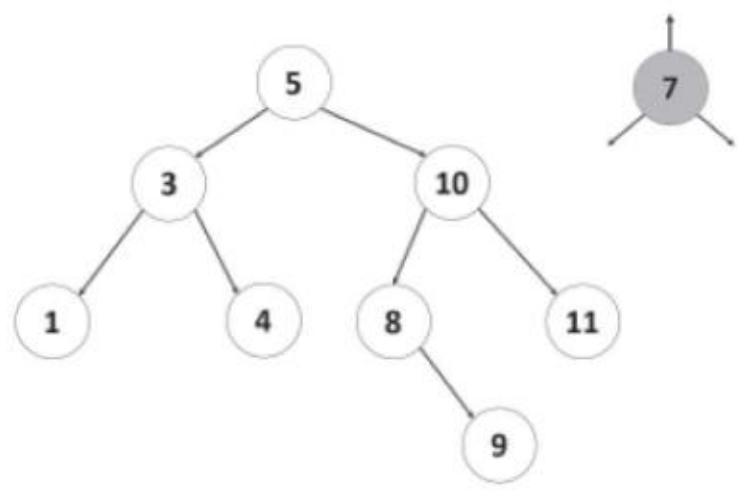
Mục đích: HS hiểu được thuật toán chèn một nút có khoá mới vào một cây tìm kiếm nhị phân T cho trước, sao cho sau khi chèn cây T vẫn là cây tìm kiếm nhị phân.

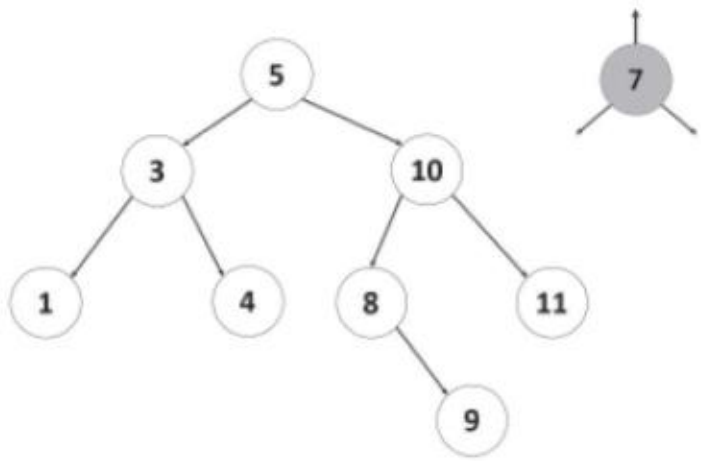
Tổ chức hoạt động:

Hoạt động này rất quan trọng vì thông qua thuật toán chèn nút mới vào cây tìm kiếm nhị phân thì HS mới có thể hiểu được thuật toán chèn, đồng thời hiểu được ý nghĩa của mô hình cây tìm kiếm nhị phân.

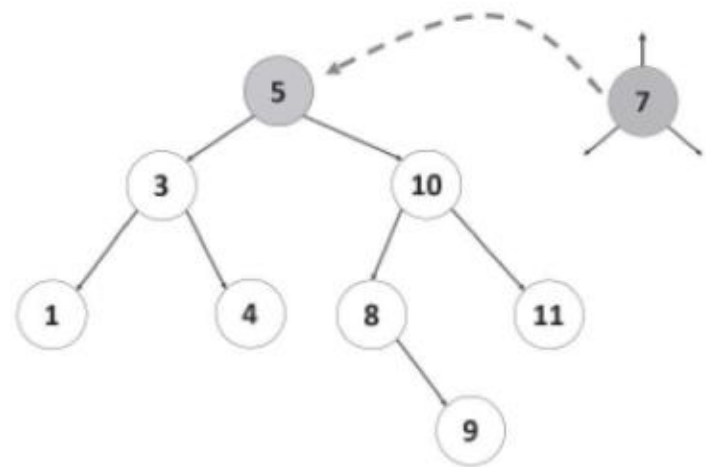
GV có thể chia lớp thành nhóm, mỗi nhóm tiến hành trao đổi và thực hiện thao tác chèn nút mới vào cây BST theo các ví dụ khác nhau. GV cũng có thể chỉ dùng một ví dụ và vẽ mô hình để HS dễ quan sát.

Có thể GV thực hiện hoạt động này như sau: GV trình bày một lần, sau đó gọi từng nhóm HS lên trình bày lại, nhưng với các bộ dữ liệu khác nhau, hoặc sử dụng nút với khoá khác. Thông qua các hoạt động này HS sẽ hiểu bài.

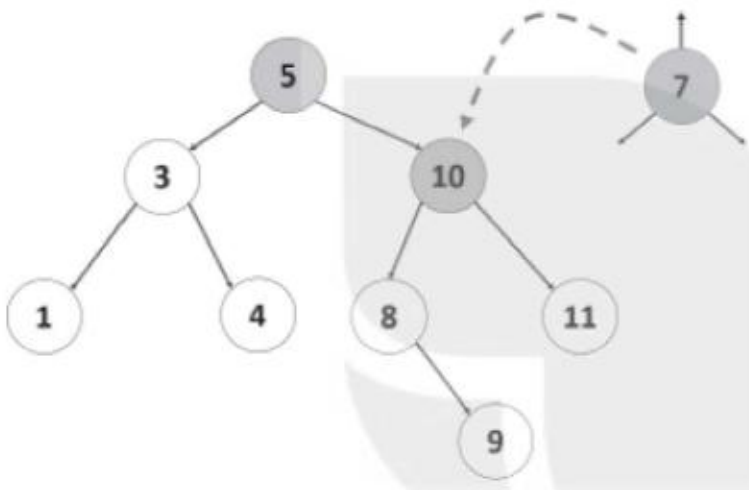
	
Đây là cây tìm kiếm nhị phân gốc.	Giả sử cần chèn vào nút mới với khoá 7.



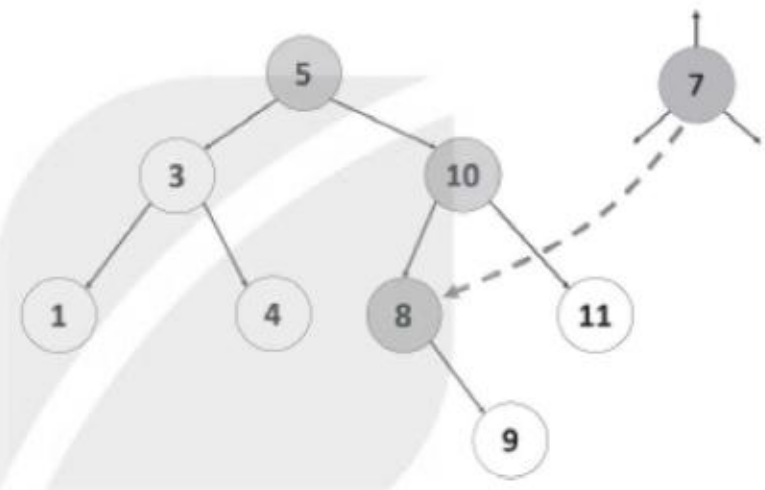
Bước 1: Thiết lập cấu trúc nút mới với khoá 7 sẵn sàng chèn vào cây tìm kiếm nhị phân T.



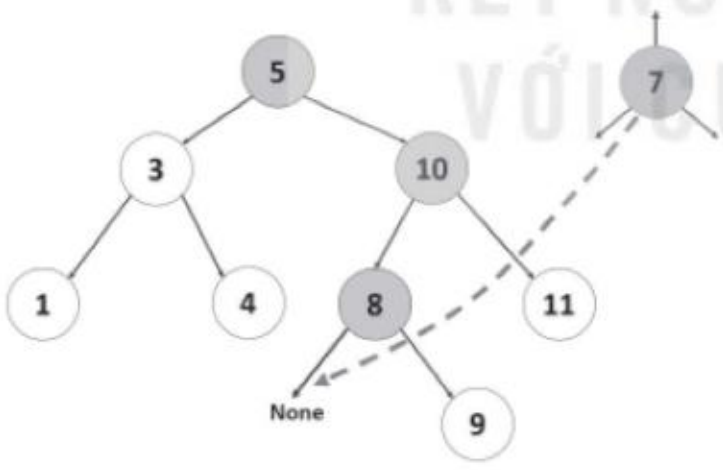
Bước 2: Công việc chèn bắt đầu bằng việc tìm kiếm vị trí để chèn nút mới. Luôn bắt đầu từ nút gốc.



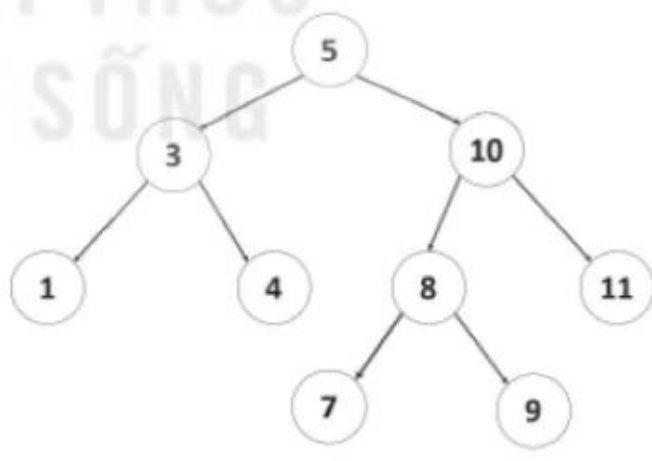
Bước 3: Vì nút gốc có khoá 5 < 7, nên sẽ chuyển tìm tiếp sang nút con phải của nút gốc (với khoá 10).



Bước 4: Nút hiện thời có khoá 10 > 7 nên sẽ chuyển tìm tiếp sang nút con trái của nút hiện thời (với khoá 8).



Bước 5: Nút hiện thời là 8 > 7 nên sẽ chuyển tìm tiếp sang nút con trái của nút hiện thời. Nhưng nút con trái này là None, do vậy đây chính là vị trí cần chèn nút mới.



Bước 6: Chèn nút mới với khoá 7 vào vị trí là nút con trái của nút với khoá 8. Hoàn thành công việc chèn nút mới vào cây T mà vẫn bảo toàn tính chất của cây BST.

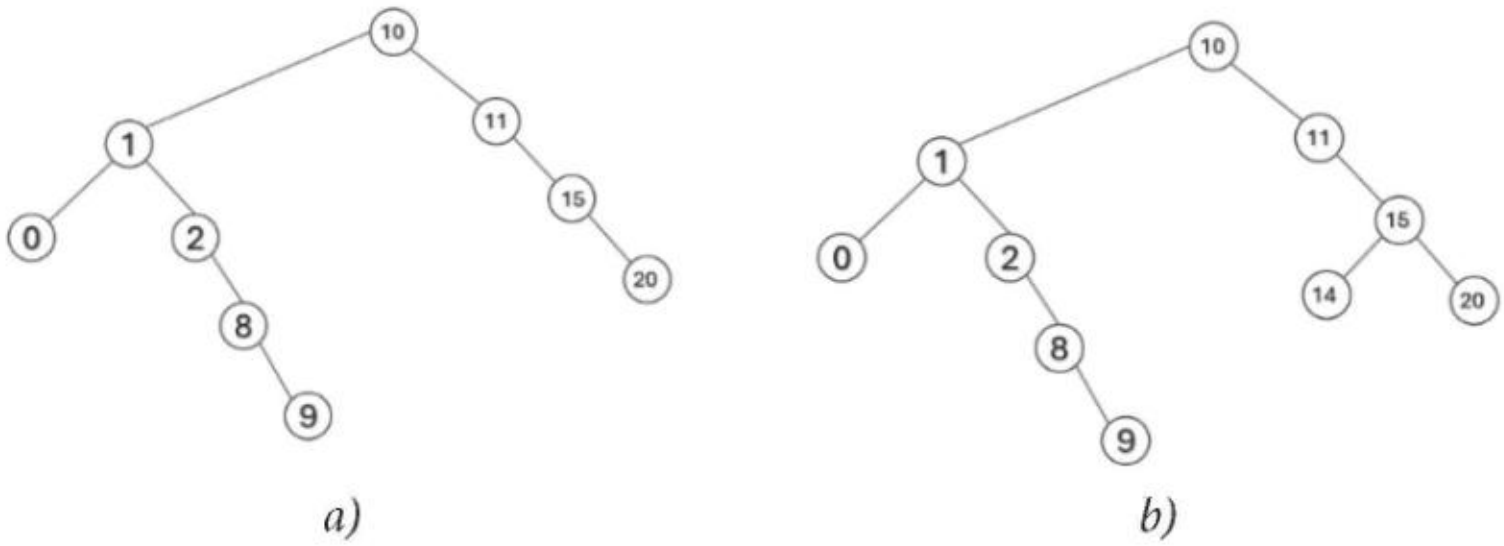
Sau khi hiểu được các bước của lệnh chèn nút mới vào cây BST, GV giới thiệu đoạn chương trình mô tả thao tác này.

Cuối cùng GV chốt kiến thức.

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

Hoạt động củng cố kiến thức

Kết quả cây tìm kiếm nhị phân Hình a (Câu 1) thu được như Hình b (Câu 2).



a) *b)*

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

Mục đích: HS hiểu được thuật toán tìm kiếm khoá trên cây tìm kiếm nhị phân, hiểu được tính ưu việt của việc tìm kiếm trên cây BST.

Tổ chức hoạt động:

GV chia lớp thành các nhóm. GV đưa một số ví dụ và bài toán tìm kiếm. Mỗi nhóm sẽ tiến hành công việc tìm kiếm theo cách đã mô tả trong SCĐ.

Các nhóm sẽ trình bày kết quả tìm kiếm của mình và đề xuất thuật toán, chương trình.

Theo phân công của GV, một số nhóm sẽ tiến hành mô tả thuật toán theo cách đệ quy, một số nhóm sẽ tiến hành mô tả thuật toán theo cách không đệ quy.

Cuối cùng GV chốt kiến thức bằng cách mô tả lại hai chương trình tìm kiếm theo đệ quy và không đệ quy.

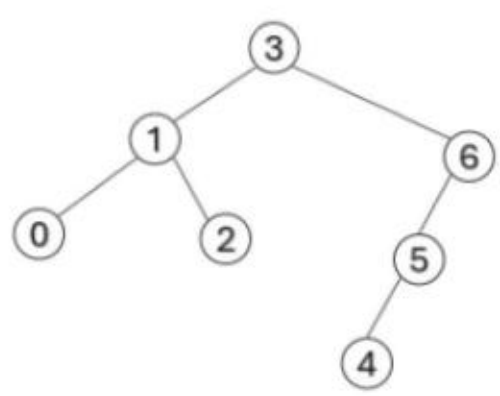
GV cùng thảo luận với các nhóm để chốt độ phức tạp thời gian của các thuật toán tìm kiếm này, so sánh với các thuật toán tìm kiếm đã học ở lớp 11, từ đó rút ra kết luận về tính ưu việt của thuật toán tìm kiếm trên mô hình cây tìm kiếm nhị phân.

Hoạt động củng cố kiến thức

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:

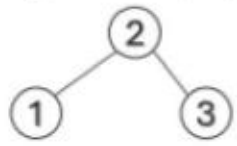
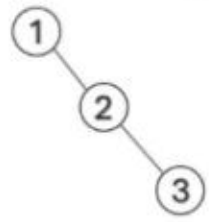
a) $3 \rightarrow 6 \rightarrow 5 \rightarrow 4$, tìm thấy.

b) $3 \rightarrow 6 \rightarrow$ None, không tìm thấy.

c) $3 \rightarrow 1 \rightarrow 0$, tìm thấy.

Hoạt động luyện tập

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

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

qlHS.py

```
fname = "Data.inp"
def NhapDL(fname):
    f = open(fname,encoding = "UTF-8")
    A = []
    for st in f:
        line = st.split()
        diem = float(line[-1])
        hoten = " ".join(line[ : -1])
        A.append((hoten,diem))
    f.close()
    return A
# Các hàm mô tả cây BST
def left(k):
    return 2*k+1
def right(k):
    return 2*k+2
def parent(k):
    return (k-1)//2
def Tree():
    return []
def Tree_Insert(T,v,d):
    k = 0
    while k < len(T) and T[k] != None:
        if v < T[k][0]:
            k = left(k)
        elif v > T[k][0]:
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
            k = right(k)
        else:
            return
    if k >= len(T):
        T.extend([None]*(k - len(T) + 1))
    T[k] = (v,d)
def search(T,k,v):
    if k >= len(T):
        return -1
    elif T[k] == None:
        return -1
    else:
        if v == T[k][0]:
            return k
        elif v < T[k][0]:
            return search(T,left(k),v)
        else:
            return search(T,right(k),v)
#Chương trình chính
def Menu():
    print("Chọn chức năng:\n1. Nhập thêm \n2. Tìm kiếm \n0. Thoát ")
    ch = int(input("Chọn chức năng (1,2,0): "))
    return ch

A = NhapDL(fname)
T = Tree()
for (ht,diem) in A:
    Tree_Insert(T,ht,diem)
while True:
    ch = Menu()
    if ch == 1:
        ht = input("Nhập thêm họ tên HS: ")
        diem = float(input("Nhập điểm HS trên: "))
        Tree_Insert(T,ht,diem)
    if ch == 2:
        ht = input("Nhập họ tên HS cần tìm điểm: ")
        k = search(T,0,ht)
        if k >= 0:
            print("Học sinh",T[k][0], "có điểm số:",T[k][1])
        else:
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
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 là biểu diễn của một cây tìm kiếm nhị phân 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[left(k)] < A[k] < A[right(k)].

(iib) 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[leftmost(k)] < A[k] < A[rightmost(k)].

Hàm leftmost(A,k) thiết lập như sau:

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

Hàm rightmost(A,k) xác định như sau:

```
1     def rightmost(T,k):  # phần tử gần nhất bên phải k
2         if right(k) < len(T) and T[right(k)] != None:
3             Idx = right(k)
4             while left(Idx) < len(T) and T[left(Idx)] != None:
5                 Idx = left(Idx)
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 hay không 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             delf.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 SCĐ. Cần phân tích kĩ hơn dòng lệnh 10, 11 khi bổ sung thêm các phần tử None vào cuối dãy A.

– Trong trường hợp trung bình, giả sử các nút của cây BST phân bổ 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 bằng chiều cao của cây. Do vậy cạnh đáy của cây sẽ là 2^{n-1}, 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(logn) 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 không là tối ưu.

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

<table>
<tr><th>Khái niệm</th><th colspan="6">Định nghĩa và phân loại</th></tr>
<tr><td>Định nghĩa chung BST</td><td colspan="6">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 các khoá của các nút thuộc cây con phải của x phải có khoá lớn hơn hoặc bằng khoá của nút x. y.key ≤ x.key ≤ z.key.</td></tr>
<tr><td>Phân loại theo yêu cầu khoá trong BST</td><td colspan="3">Cho phép khoá trùng nhau</td><td colspan="3">Các khoá phải khác nhau từng đôi một</td></tr>
<tr><td>Phân loại theo kiểu cài đặt cây BST</td><td>Nút liên kết</td><td colspan="2">Mảng một chiều</td><td>Nút liên kết</td><td colspan="2">Mảng một chiều</td></tr>
<tr><td>Phân loại chi tiết cho từng kiểu cài đặt</td><td>Tại mọi nút x, mọi nút y trong cây con trái và nút z trong cây con phải của x ta có:
y.key < x.key ≤ z.key
hoặc:
y.key ≤ x.key < z.key.</td><td colspan="2">T[i] < T[k] ≤ T[j]
hoặc:
T[i] ≤ T[k] < T[j].</td><td>Với mọi nút x, mọi nút y trong cây con trái và nút z trong cây con phải của x ta có:
y.key < x.key < z.key.</td><td colspan="2">T[i] < T[k] < T[j]</td></tr>
<tr><td>Kiểu nút liên kết</td><td>Nút có 3 thuộc tính (left, right, p)</td><td colspan="2">Nút có 2 thuộc tính (left, right)</td><td>Nút có 3 thuộc tính (left, right, p)</td><td>Nút có 2 thuộc tính (left, right)</td><td></td></tr>
<tr><td>Phân loại</td><td>(1)</td><td colspan="2">(2)</td><td>(4)</td><td>(5)</td><td>(6)</td></tr>
</table>

BÀI 8. THỰC HÀNH CÂY TÌM KIẾM NHỊ PHÂN

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. Hoặc có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

1. GV giới thiệu tổng quan mục đích của bài thực hành, các nhiệm vụ cần thực hiện.

2. Với mỗi nhiệm vụ, GV đọc đề bài, phân tích cách làm, sau đó yêu cầu HS thực hiện dựa trên SCĐ đã có.

3. Trong quá trình thực hiện nếu HS gặp khó khăn, GV hướng dẫn và giải quyết các khó khăn đó.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động khởi động

Phần khởi động GV sẽ cùng 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.

Ví dụ các câu trả lời có thể như sau, không yêu cầu HS trả lời chính xác:

1. Cây nhị phân hoàn hảo có n nút thì chiều cao của nó là bao nhiêu?

Trả lời: Chiều cao của nó là $\log_2 (n+1)$.

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?

Trả lời: Trong số các cây tìm kiếm nhị phân thì cây tìm kiếm nhị phân 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).

3. Em hãy đề xuất thuật toán xoá một phần tử của cây tìm kiếm nhị phân được cài đặt bằng mảng.

Trả lời: Để xoá một phần tử của cây tìm kiếm nhị phân, 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 thành None bằng lệnh T[k] = None.

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

Mục đích: 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 lí các đối tượng có các thuộc tính khác ngoài khoá.

Tổ chức hoạt động:

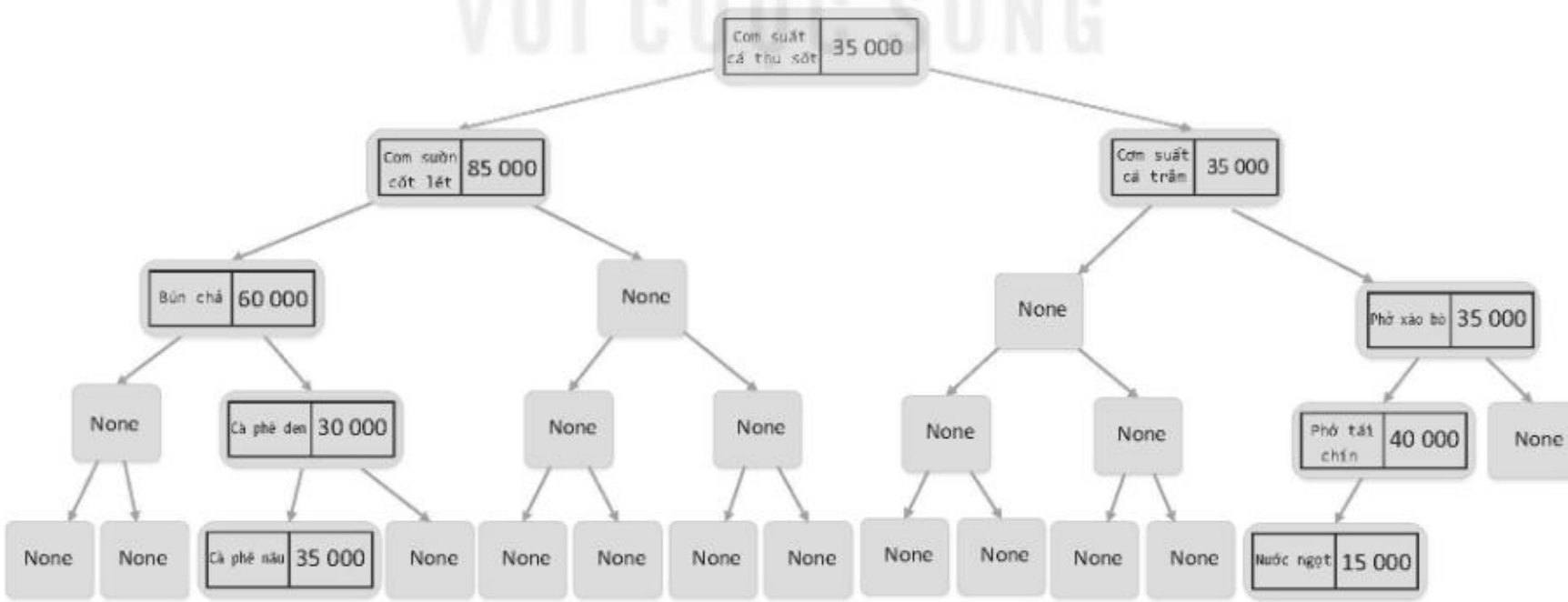
GV tổ chức cho các nhóm tìm hiểu nhiệm vụ, thảo luận đề xuất ý tưởng, triển khai thành mã nguồn, kiểm tra với dữ liệu sách đã cho và minh hoạ trước lớp.

Hoạt động luyện tập

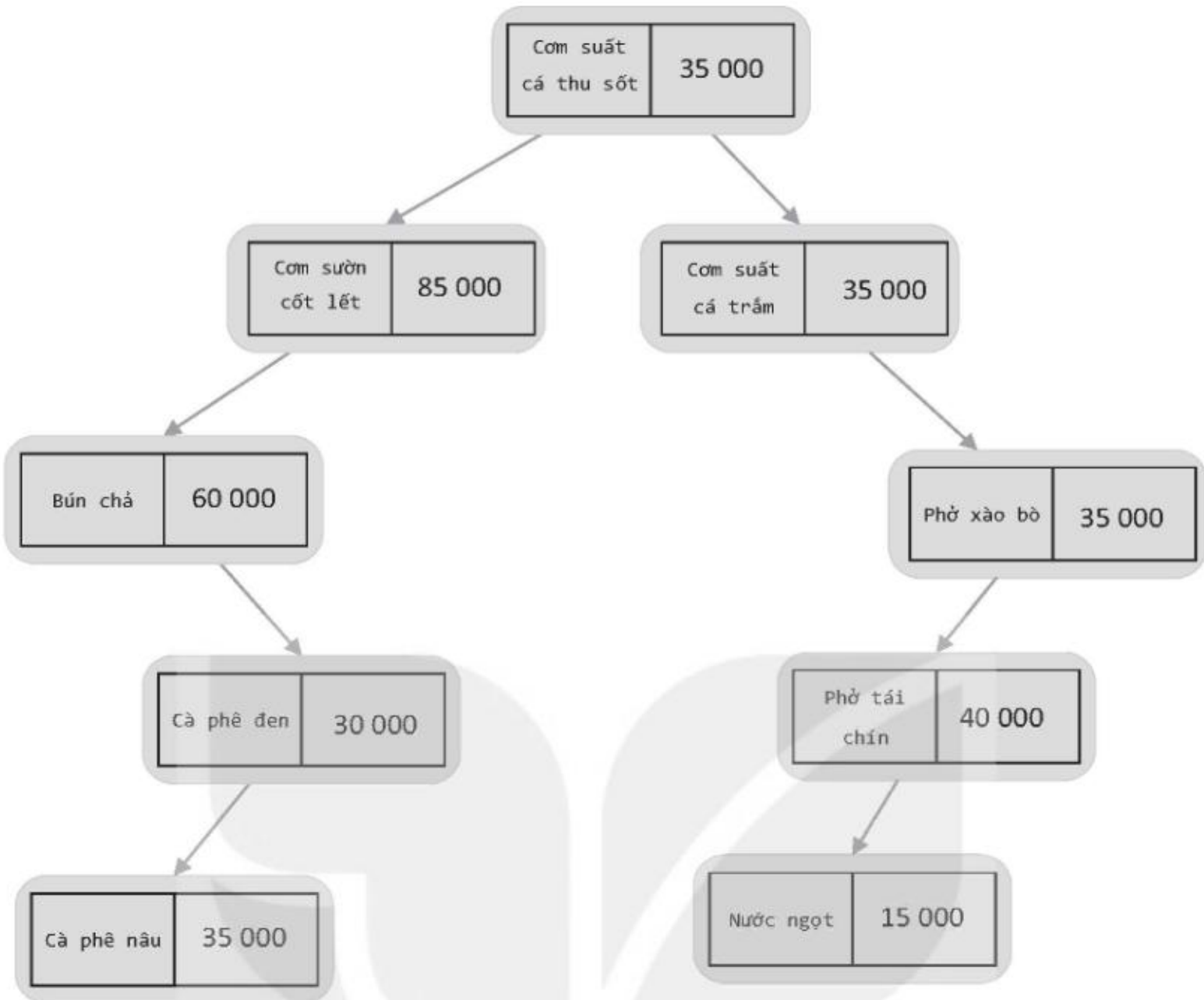
1. Để vẽ lại cây tìm kiếm nhị phân ứng với tệp *menu.inp*, đầu tiên cần sắp xếp tên các món ăn theo thứ tự từ điển để dễ so sánh các khoá khi chèn các nút vào cây. Thứ tự như sau:

Bún chả
Cà phê đen
Cà phê nâu
Cơm sườn cốt lết
Cơm suất cá thu sốt
Cơm suất 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 có 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:



Cây nhị phân thực sự sau khi bỏ các nút giả None có thể được hình dung như sau:



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

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

Ở lần gọi hàm search thứ ba, search(T, k = 3, name = "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 60 000.

Hoạt động vận dụng

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 tử 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
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

Account_management.py

```
1   from BST_account import *  #Khai báo thư viện cây tìm kiếm nhị phân
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 lí
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 Vận dụng 1 nhưng sử dụng thêm hàm inorder để in danh sách các HS 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 #Thể 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 #Thể hiện đã cập nhật
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

Pupil_management.py

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

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

D THÔNG TIN BỔ SUNG

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

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
            else:
                y.right = z
    def Tree_Insert_Update(T, name, price):
        n = search(T.root, name)
        if n is not None:
            n.price = price
            return 1 #Thể hiện đã cập nhật giá
        else:
            newNode = Node(name, price)
            Tree_Insert(T, newNode)
            return 0 #Thể hiện đã thêm món mới
```

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

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. GV có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2
Tìm hiểu các thuật toán duyệt cây tìm kiếm nhị phân.	Tìm hiểu thuật toán sắp xếp dãy số bằng cây tìm kiếm nhị phân.

Hoạt động khởi động

Mục đích: 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 HS trả lời và nhận ra được các điểm mới khi áp dụng các thuật toán duyệt đã học cho mô hình cây BST.

Không yêu cầu HS trả lời đúng và chính xác các câu hỏi.

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

Mục đích: 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 6, biết được ý nghĩa thuật toán duyệt giữa và duyệt ngược trên cây tìm kiếm nhị phân.

Tổ chức hoạt động:

GV chia nhóm HS. Các nhóm trao đổi, thảo luận về các thuật toán duyệt đã biết nhưng bây giờ sẽ áp dụng cho cây tìm kiếm nhị phân.

Cần chú ý đến hai bước sau:

Bước 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. Điểm khác biệt duy nhất như sau:

Trong Bài 6, mô hình cây được học là cây nhị phân hoàn chỉnh, còn trong bài học này thì cây tìm kiếm nhị phân 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.

Bước 2: 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.

Cuối cùng GV chốt kiến thức.

Hoạt động củng cố kiến thức

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.

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

Mục đích: HS hiểu được ý tưởng và cách thực hiện thuật toán sắp xếp dãy số mới dựa trên cây tìm kiếm nhị phân và so sánh với các thuật toán sắp xếp đã biết.

Tổ chức hoạt động:

GV chia nhóm HS, GV cùng các nhóm trao đổi về ý tưởng thiết kế thuật toán sắp xếp dãy dựa trên thuật toán duyệt giữa của cây tìm kiếm nhị phân.

GV đưa ra hai cách cài đặt cụ thể dựa trên hai khả năng:

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.

Cuối cùng GV chốt kiến thức.

Hoạt động củng cố kiến thức

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:
4               insert(T,x,0)
5           inorder(T,0)
```

Hoạt động luyện tập

1. 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

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           while left(k) < len(T) and T[left(k)] != None:
3               k = left(k)
4           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] = 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              elif v > T[k]:
16                  k = right(k)
17              else:
18                  C[k] = C[k] + 1
19          if k >= len(T):
20              T.extend([None]*(k - len(T) + 1))
21              C.extend([None]*(k - len(C) + 1))
22          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 2 phần Luyện tập, ta đã biết thuật toán sắp xếp dãy số sử dụng cây tìm kiếm nhị phân sử dụng mảng 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(nlogn), 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.

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

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 có khoá nhỏ hơn hoặc bằng khoá của x, tất các khoá của các nút thuộc cây con phải của x 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 và 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] ≤ T[j] hoặc: T[i] ≤ T[k] < T[j]	Với mọi nút x, mọi nút y trong cây con trái và 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(logn)	O(logn)	O(n), xấu nhất: $O(2^n)$	O(logn)	O(logn)	O(n), xấu nhất: $O(2^n)$
Thiết lập cây từ dãy	O(nlogn)	O(nlogn)	$O(n^2)$, xấu nhất: $O(2^n)$	O(nlogn)	O(nlogn)	$O(n^2)$, xấu nhất: $O(2^n)$
Duyệt cây	O(n)	O(n)	O(n)	O(n)	O(n)	O(n)
Sắp xếp	O(nlogn)	O(nlogn)	$O(n^2)$, xấu nhất: $O(2^n)$	O(nlogn)	O(nlogn)	$O(n^2)$, xấu nhất: $O(2^n)$

BÀI 10. THỰC HÀNH TỔNG HỢP VỚI CÂY TÌM KIẾM NHỊ PHÂN

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản hoặc có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động khởi động

Phần khởi động GV sẽ cùng HS ôn lại các kiến thức đã biết từ các bài học trước. HS cùng trao đổi và trả lời các câu hỏi của GV đưa ra.

Hoạt động thực hành: Viết chương trình quản lí danh bạ điện thoại

Mục đích: 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.

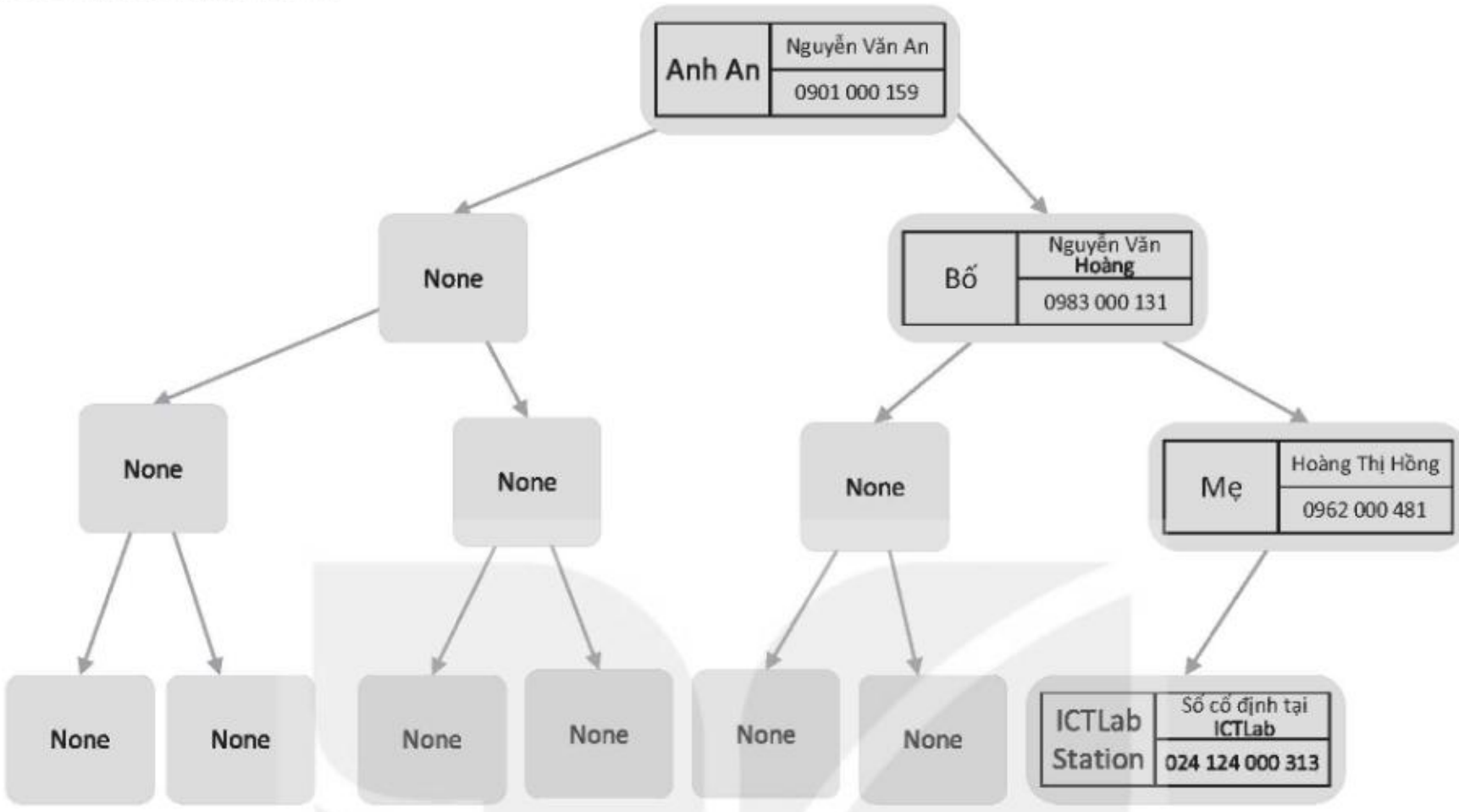
Tổ chức hoạt động:

GV tổ chức cho 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 hoạ trước lớp.

Hoạt động luyện tập

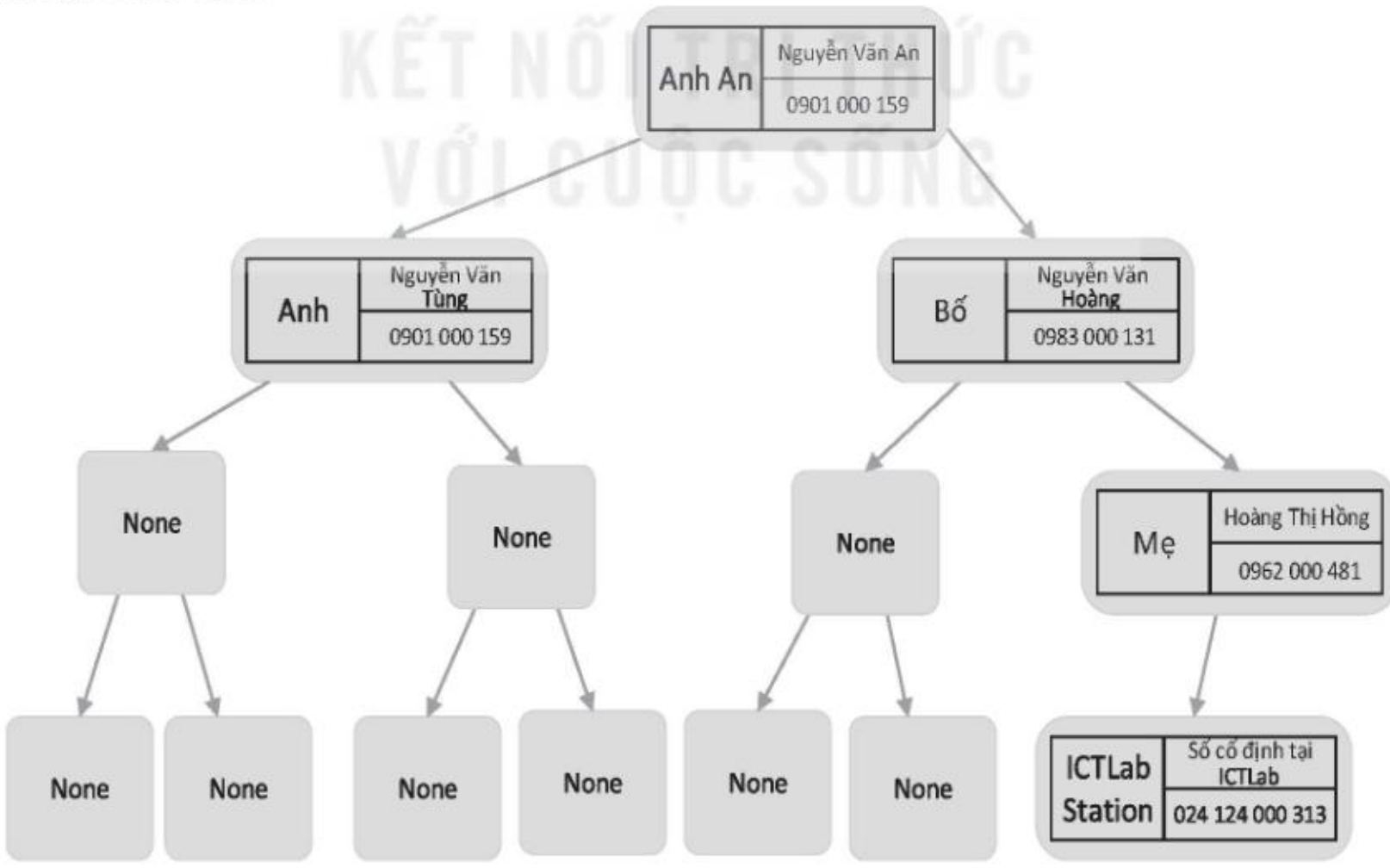
1. a) Để vẽ lại cây tìm kiếm nhị phân ứng với tệp contacts.inp hãy thực hiện dần các bước của thuật toán chèn cho từng liên hệ trong tệp contacts.inp. Do dữ liệu chỉ gồm 4 liên hệ trong đó 3 liên hệ đầu tiên được chèn vào lại đúng với thứ tự khoá của chúng, nút "Bố"

và "Mẹ" đều nằm ở nhánh phải của cây. Vì "ICTLab Station" < "Mẹ" nên "ICTLab Station" là nút con trái của "Mẹ". Cây tìm kiếm nhị phân thu được, không may thay, có dạng "cây suy biến" như Hình 10.1.



Hình 10.1

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 10.2.



Hình 10.2

Để in được các danh bạ ở một trang, cần phải đếm thứ tự các liên hệ khi duyệt cây. GV có thể gợi ý HS 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

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

#Khởi tạo cây tìm kiếm và đọc dữ liệu từ tệp
T = Tree()
f = open("contacts_paging.inp", "r", encoding="utf8") #File chứa nhiều liên hệ
lines = f.readlines()
for line in lines:
    l = line.strip()
    l = l.split(",", maxsplit=3)
    Tree_Insert(T, l[0], l[1], l[2])
f.close()

count = 0
starting = 0
ending = 0
def inorder_paging(T, k):
    global count
    global starting
    global ending
    if(count>ending):
        return
    if k < len(T) and T[k] != None:
        inorder_paging(T,left(k))
        count += 1
        if(count>= starting):
            print(T[k][0], "|", T[k][1], "|", T[k][2])
        inorder_paging(T,right(k))

#Hiển thị các chức năng chương trình và xử lí theo lựa chọn
options = ["Thoát chương trình", "Hiển thị các liên hệ theo thứ tự alphabet",
"Tra cứu liên hệ", "Cập nhật hoặc thêm một liên hệ"]
selected = -1 #Chức năng người dùng chọn lựa từ menu, được khởi tạo bằng -1
while selected != 0:
    for choice in range(len(options)):
        print(choice, "-", options[choice] )
    selected = int(input("Chọn chức năng: "))
    if selected < 0 or selected > 4:
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
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

1. Bài tập này luyện kĩ năng duyệt cây tìm kiếm nhị phân 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

```
1   #Tiếp đoạn mã nguồn file BST_pupil ở Bài 8
2   def inorder_lef2right(T,k):
3       if k < len(T) and T[k] != None:
4           inorder_lef2right(T, left(k))
5             print("Mã", T[k][0], "Họ tên:", T[k][1], "Ngày sinh:", T[k][2].
    strftime("%d-%m-%Y"))
6           inorder_lef2right(T, right(k))
```

```
7
8       def inorder_right2left(T,k):
9           if k < len(T) and T[k] != None:
10              inorder_right2left(T, right(k))
11                  print("Mã", T[k][0], "Họ tên:", T[k][1], "Ngày sinh:", T[k][2].
        strftime("%d-%m-%Y"))
12              inorder_right2left(T, left(k))
```

Pupil_management.py

```
1       from BST_pupil import *  #Khai báo thư viện cây tìm kiếm nhị phân
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 lí
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)
32            except:
33                print("Ngày tháng không hợp lệ")
34                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_lef2right(T,0)
41        elif selected == 4:
42            inorder_right2left(T,0)
```

2. Do yêu cầu của đề bài là sắp xếp các món ăn theo giá bằng cách 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))
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

menu_management.py

```
1     from BST_menu import * #Khai báo thư viện cây tìm kiếm nhị phân
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
13    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 điện thoại được cài đặt bằng cấu trúc nút liên kết thay vì mảng tránh để 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
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
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)
```

KẾT NỐI TRI THỨC
VỚI CUỘC SỐNG

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

BÀI 11. KHÁI NIỆM ĐỒ THỊ

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

1. Kiến thức

– Khái niệm đồ thị.

– Các định nghĩa và khái niệm liên quan đến đồ thị: Bậc của đỉnh đồ thị, đường đi, chu trình, tính liên thông, ma trận kề, danh sách kề.

– Cấu trúc và ý nghĩa của ma trận kề, danh sách kề.

2. Năng lực

– 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

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. Hoặc có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2	Hoạt động 3
Tìm hiểu khái niệm đồ thị.	Tìm hiểu một số khái niệm, định nghĩa liên quan đến đồ thị.	Tìm hiểu cách biểu diễn dữ liệu đồ thị trên máy tính.

Hoạt động khởi động

Mục đích: HS làm quen với một bài toán mới, qua đó gợi tò mò muốn biết về một mô hình dữ liệu mới.

GV giới thiệu bài toán 7 cây cầu Königsberg. HS làm quen, trao đổi nhanh xem có cách giải nào không.

Thông qua ví dụ này, GV hỏi HS về mô hình dữ liệu cần để mô tả bài toán. Không yêu cầu HS trả lời đúng.

Hoạt động 1: Tìm hiểu khái niệm đồ thị

Mục đích: HS biết và nắm được khái niệm đồ thị và một số định nghĩa, khái niệm trực tiếp có liên quan đến đồ thị.

Tổ chức hoạt động:

GV chia nhóm HS để trao đổi, thảo luận một số vấn đề, khái niệm, định nghĩa có trong sách. Sau mỗi phần GV cần chốt lại để HS nắm được và tìm thêm các ví dụ thực tế để minh hoạ cho các định nghĩa này.

– Khái niệm và định nghĩa đồ thị. Các khái niệm: đỉnh, cạnh đồ thị.

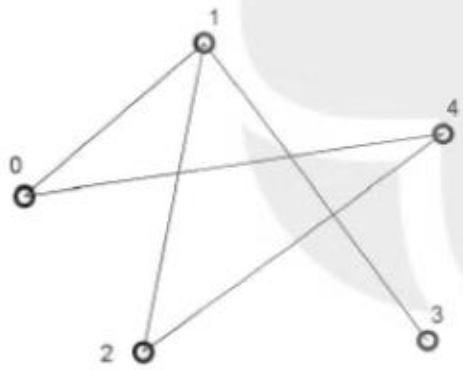
– Đồ thị vô hướng và có hướng.

– Khái niệm và định nghĩa đơn đồ thị.

Hoạt động củng cố kiến thức

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

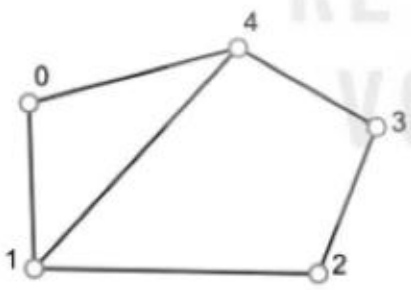
2. Đồ thị này có thể vẽ như sau:



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

Mục đích: HS biết và nắm được một số khái niệm, định nghĩa quan trọng liên quan đến đồ thị.

Tổ chức hoạt động:

GV chia nhóm HS để trao đổi, thảo luận một số vấn đề, khái niệm, định nghĩa có trong sách. Sau mỗi phần GV cần chốt lại để HS nắm được và tìm thêm các ví dụ thực tế để minh hoạ cho các định nghĩa này.

– Bậc của các đỉnh của đồ thị.

– Đường đi và chu trình trong đồ thị.

– Đồ thị liên thông. Thành phần liên thông của đồ thị.

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

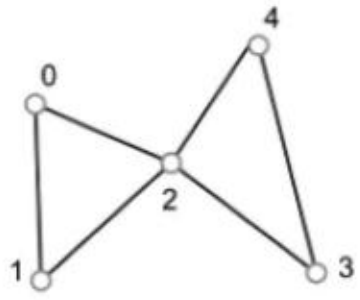
– Ma trận kề.

– Danh sách kề.

Hoạt động 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.

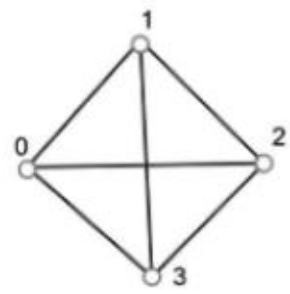
2. a)



Ma trận kề và danh sách kề:

$$A=\begin{pmatrix}01100\\10100\\11011\\00101\\00110\end{pmatrix}\quad Adj=\begin{pmatrix}0\to 1,2\\1\to 0,2\\2\to 0,1,3,4\\3\to 2,4\\4\to 2,3\end{pmatrix}$$

b)



Ma trận kề và danh sách kề:

$$A=\begin{pmatrix}0111\\1011\\1101\\1110\end{pmatrix}\quad Adj=\begin{pmatrix}0\to 1,2,3\\1\to 0,2,3\\2\to 0,1,3\\3\to 0,1,2\end{pmatrix}$$

Hoạt động 3: Tìm hiểu cách biểu diễn dữ liệu đồ thị trên máy tính

Mục đích: HS nắm được cách tổ chức và biểu diễn các mô hình đồ thị trên máy tính.

Tổ chức hoạt động:

GV chia lớp thành các nhóm cùng trao đổi, thảo luận. Sau đó từng nhóm lên trình bày và mô tả cách biểu diễn, hay cách thiết lập mô hình dữ liệu đồ thị trên máy tính. Các khái niệm cần mô tả cụ thể bao gồm:

– Đỉnh đồ thị. Danh sách các đỉnh.

– Cạnh. Danh sách các cạnh.

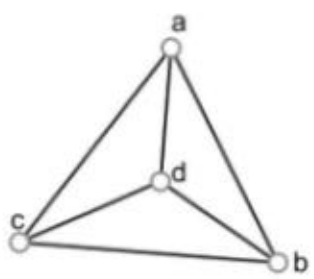
– Ma trận kề.

– Danh sách kề.

Lưu ý: Ma trận kề và danh sách kề đều được mô tả bằng kiểu dữ liệu list của Python.

Hoạt động củng cố kiến thức

a)

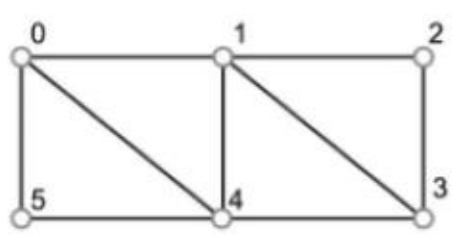


n = 4

V = [a, b, c, d]

E = [{a, b}, {a, c}, {a, d}, {b, c}, {b, d}, {c, d}]

b)



n = 6

V = [0, 1, 2, 3, 4, 5]

E = [{0, 1}, {0, 4}, {0, 5}, {1, 2}, {1, 3}, {1, 4}, {2, 3}, {3, 4}, {4, 5}]

Ma trận kề và danh sách kề:

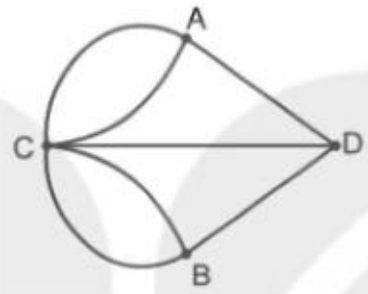
$$A = \begin{pmatrix} 0111 \\ 1011 \\ 1101 \\ 1110 \end{pmatrix} \quad Adj = \begin{pmatrix} a \to b,c,d \\ b \to a,c,d \\ c \to a,b,d \\ d \to a,b,c \end{pmatrix}$$

Ma trận kề và danh sách kề:

$$A = \begin{pmatrix} 010011 \\ 101110 \\ 010100 \\ 011010 \\ 110101 \\ 100010 \end{pmatrix} \quad Adj = \begin{pmatrix} 0 \to 1,4,5 \\ 1 \to 0,2,3,4 \\ 2 \to 1,3 \\ 3 \to 1,2,4 \\ 4 \to 0,1,3,5 \\ 5 \to 0,4 \end{pmatrix}$$

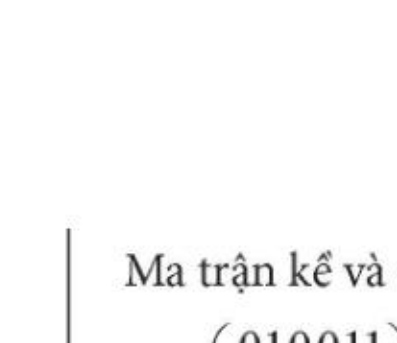
Hoạt động luyện tập

1. Đồ thị ứng với bài toán 7 cây cầu không phải là đồ thị đơn. Bậc của các đỉnh của đồ thị này như sau: deg(A) = 3, deg(B) = 3, deg(C) = 5, deg(D) = 3.



2. Đồ thị có dạng như hình sau:

$$A = \begin{pmatrix} 0110 \\ 1001 \\ 1001 \\ 0110 \end{pmatrix}$$



Hoạt động vận dụng

1. Các ma trận kề của đồ thị đầy đủ tương ứng n = 2, 3, 4 lần lượt như sau:

$$A = \begin{pmatrix} 01 \\ 10 \end{pmatrix} \qquad A = \begin{pmatrix} 011 \\ 101 \\ 110 \end{pmatrix} \qquad A = \begin{pmatrix} 0111 \\ 1011 \\ 1101 \\ 1110 \end{pmatrix}$$

2. Đáp số: 3 thành phần liên thông.

D THÔNG TIN BỔ SUNG

Đồ thị là một mô hình dữ liệu mạnh mẽ được sử dụng trong nhiều lĩnh vực khác nhau để biểu diễn và giải quyết các bài toán liên quan đến mối quan hệ giữa các đối tượng. Dưới đây là một số ứng dụng phổ biến của đồ thị:

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

1. Mạng xã hội:

– Mạng xã hội như Facebook, Twitter được biểu diễn bằng đồ thị, với các nút là người dùng và các cạnh là mối quan hệ kết bạn.

– Đồ thị giúp thực hiện các chức năng như tìm kiếm bạn bè, đề xuất kết bạn, lan truyền thông tin,...

2. Bản đồ:

– Bản đồ giao thông được biểu diễn bằng đồ thị, với các nút là giao lộ và các cạnh là đường.

– Đồ thị giúp tìm đường đi ngắn nhất, tính toán thời gian di chuyển,...

3. Lập kế hoạch:

– Lập kế hoạch dự án được biểu diễn bằng đồ thị, với các nút là các nhiệm vụ và các cạnh là mối quan hệ phụ thuộc.

– Đồ thị giúp xác định thứ tự thực hiện các nhiệm vụ, tính toán thời gian hoàn thành dự án,...

4. Học máy: Học máy sử dụng đồ thị để biểu diễn dữ liệu và mô hình học. Ví dụ, đồ thị Bayes được sử dụng để phân loại dữ liệu, mạng neuron được sử dụng để học tập và dự đoán.

5. Khoa học máy tính: Sử dụng đồ thị để biểu diễn và giải quyết các bài toán như: Tìm kiếm và sắp xếp; Tối ưu hoá; Xác định chu trình; Phân tích mạng; Mô hình hoá mối quan hệ giữa các bản ghi dữ liệu trong cơ sở dữ liệu quan hệ hoặc cơ sở dữ liệu đồ thị; Mô phỏng quá trình ra quyết định và giải các bài toán lôgic phức tạp trong Trí tuệ nhân tạo,...

6. Sinh học và chăm sóc sức khoẻ: Sử dụng đồ thị để biểu diễn cấu trúc protein, mạng tương tác protein,... Đồ thị giúp nghiên cứu các tương tác giữa các protein và dự đoán chức năng của protein, mô hình hoá và dự đoán sự lây lan của dịch bệnh trong dân số.

7. Hoá học: Sử dụng đồ thị để biểu diễn cấu trúc phân tử và mạng phản ứng hoá học, giúp nghiên cứu các phản ứng hoá học và thiết kế các phân tử mới.

8. Khoa học xã hội: Sử dụng để nghiên cứu mối quan hệ và tương tác giữa các cá nhân trong một mạng xã hội, như Facebook hoặc Twitter,... Đồ thị được sử dụng để biểu diễn mối quan hệ xã hội, giúp nghiên cứu về cấu trúc xã hội và các nhóm xã hội.

9. Kinh tế và quản lí: Mô hình hoá và phân tích các mạng lưới kinh tế, như mạng lưới ngân hàng hoặc thị trường tài chính. Sử dụng đồ thị để tối ưu hoá chuỗi cung ứng và quản lí dòng chảy của hàng hoá và dịch vụ.

10. Giao thông: Mô hình hoá và tối ưu hoá mạng lưới giao thông, từ mạng đường bộ đến mạng hàng không và đường sắt.

BÀI 12. BIỂU DIỄN ĐỒ THỊ

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

1. Kiến thức

– Các dạng tệp dữ liệu đồ thị.

– Biểu diễn đồ thị bằng ma trận kề.

– Biểu diễn đồ thị bằng danh sách kề.

2. Năng lực

Thiết lập và biểu diễn được đồ thị bằng ma trận kề và danh sách kề.

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. Hoặc có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2	Hoạt động 3
Tìm hiểu các mô hình dữ liệu đồ thị (vô hướng hoặc có hướng).	Tìm hiểu cách thiết lập đồ thị từ ma trận kề và danh sách kề.	Tìm hiểu cách thiết lập dữ liệu đồ thị từ tệp dữ liệu danh sách các cạnh.

Hoạt động khởi động

Mục đích: HS được làm quen và nhận biết khuôn dạng dữ liệu đầu vào của đồ thị dưới dạng các tệp dữ liệu.

GV cho HS quan sát hình ảnh đồ thị và các tệp dữ liệu, HS dự đoán dữ liệu nào có trong các tệp này.

Hoạt động 1: Tìm hiểu các mô hình dữ liệu đồ thị (vô hướng hoặc có hướng)

Mục đích: HS nhận biết và nắm vững 3 dạng tệp dữ liệu đầu vào thường gặp nhất của đồ thị.

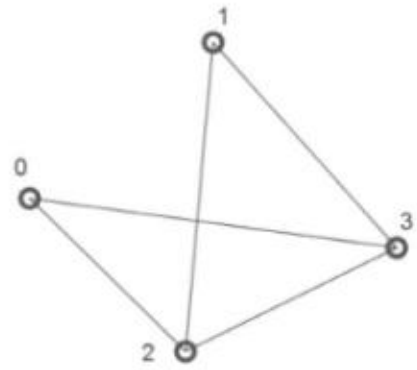
Tổ chức hoạt động:

GV chia nhóm HS và cho các nhóm lần lượt nhận biết, sau đó mô tả chi tiết 3 dạng dữ liệu đầu vào của đồ thị. Chú ý các tệp dữ liệu này đều có thể là dữ liệu của đồ thị vô hướng và có hướng. Ba dạng tệp dữ liệu của đồ thị bao gồm:

– Tệp dữ liệu danh sách các cạnh.

– Tệp dữ liệu ma trận kề.

– Tệp dữ liệu danh sách kề.



Hình 12.1

Hoạt động củng cố kiến thức

1. Đồ thị có dạng như Hình 12.1.

2. Có. Vì tệp dữ liệu danh sách các cạnh không quy định thứ tự các cạnh, nên hai tệp khác nhau vẫn biểu diễn cùng một đồ thị.

Hoạt động 2: Tìm hiểu cách thiết lập đồ thị từ ma trận kề và danh sách kề

Mục đích: HS biết và nắm được cách thiết lập bộ dữ liệu chính của đồ thị từ tệp dữ liệu ma trận kề và danh sách kề.

Tổ chức hoạt động:

GV tổ chức lớp để tiếp nhận, thử nghiệm và hiểu các chương trình thiết lập dữ liệu đã có trong SCĐ. Cụ thể như sau:

a) Dữ liệu đầu vào là tệp ma trận kề. Khi đó chương trình tương ứng sẽ áp dụng chung cho cả đồ thị vô hướng và có hướng.

Input: Tệp ma trận kề. Output: V, A.

b) Dữ liệu đầu vào là tệp danh sách kề. Khi đó chương trình tương ứng sẽ áp dụng chung cho cả đồ thị vô hướng và có hướng.

Input: Tệp danh sách kề. Output: V, Adj.

Hoạt động củng cố kiến thức

1. Khẳng định đúng: Adj[i] bằng số lượng các phần tử 1 của hàng thứ i của ma trận kề A.

2. Ma trận kề A gồm toàn số 0 nếu đồ thị không có bất kì cạnh nào, tức là số cạnh bằng 0.

Hoạt động 3: Tìm hiểu cách thiết lập dữ liệu đồ thị từ tệp dữ liệu danh sách các cạnh

Mục đích: HS biết và nắm được cách thiết lập bộ dữ liệu chính của đồ thị từ tệp dữ liệu danh sách các cạnh.

Tổ chức hoạt động:

GV tổ chức lớp để tiếp nhận, thử nghiệm và hiểu các chương trình thiết lập dữ liệu đã có trong SCĐ. Cụ thể như sau:

a) Trường hợp đồ thị vô hướng. Sẽ có hai chương trình:

1. Input: Danh sách các cạnh. Output: V, A.

2. Input: Danh sách các cạnh. Output: V, Adj.

b) Trường hợp đồ thị có hướng. Sẽ có hai chương trình:

1. Input: Danh sách các cạnh. Output: V, A.

2. Input: Danh sách các cạnh. Output: V, Adj.

Hoạt động củng cố kiến thức

1. Đồ thị đơn, vô hướng, n đỉnh thì số cạnh lớn nhất là tổ hợp chập 2 của n, tức là bằng $\frac{n(n-1)}{2}$.

2. Nếu đồ thị không có bất kì cạnh nào thì tất cả các phần tử của Adj đều rỗng.

Hoạt động luyện tập

1. Ví dụ chương trình nhập danh sách các cạnh và trả về V, Adj cho đồ thị có hướng có thể sửa lại như sau:

```
1     def BuildGraph(fname):
2         f = open(fname)
3         n = int(f.readline())
4         V = [i for i in range(n)]
5         Adj = [[] for i in range(n)]
6         for line in f:
7             edge = [int(ch) for ch in line.split()]
8             i,j = edge[0],edge[1]
9             if i != j:
10                Adj[i].append(j)
11                Adj[j].append(i)
12        f.close()
13        return V,Adj
```

2. Tính được:

– 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.

Hoạt động vận dụng

1. Hàm GraphEdge(A) của đồ thị vô hướng có thể được thiết lập như sau:

```
1     def GraphEdge(A):
2         n = len(A)
3         E = []
4         for i in range(n):
```

```
5               for j in range(i+1,n):
6                   if A[i][j] == 1:
7                       E.append({i,j})
8           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:

```
1       def GraphEdge(Adj):
2           n = len(Adj)
3           E = []
4           for i in range(n):
5               for j in Adj[i]:
6                   if {i,j} not in E:
7                       E.append({i,j})
8           return E
```

b) Đồ thị có hướng, hàm GraphEdge(Adj) được thiết lập như sau:

```
1       def GraphEdge(Adj):
2           n = len(Adj)
3           E = []
4           for i in range(n):
5               for j in Adj[i]:
6                   E.append((i,j))
7           return E
```

D THÔNG TIN BỔ SUNG

Để giúp HS hiểu rõ và vận dụng tốt hơn về cách biểu diễn dữ liệu của đồ thị thông qua ma trận kề, danh sách cạnh và danh sách kề, GV có thể cung cấp thêm thông tin về các ứng dụng cụ thể của mỗi cách biểu diễn. Dưới đây là một vài gợi ý:

1. Ma trận kề

– Mạng lưới điện: Ma trận kề có thể được sử dụng để biểu diễn mạng lưới điện, với các đỉnh đại diện cho các trạm biến áp và các cạnh đại diện cho các đường dây điện. Ma trận kề sẽ cho biết có đường dây điện nối giữa các trạm biến áp hay không.

– Ma trận liên thông: Ma trận liên thông có thể được sử dụng để biểu diễn mối quan hệ liên thông giữa các thành phố trong một quốc gia. Ma trận liên thông sẽ cho biết có đường bộ hoặc đường sắt nối giữa các thành phố hay không.

– Bảng xếp hạng: Ma trận kề có thể được sử dụng để biểu diễn bảng xếp hạng trong các cuộc thi đấu. Ma trận kề sẽ cho biết hai đội đã thi đấu với nhau hay chưa.

2. Danh sách cạnh

– Mạng xã hội: Danh sách bạn bè của mỗi người dùng trong mạng xã hội có thể được biểu diễn bằng danh sách cạnh. Mỗi cạnh đại diện cho mối quan hệ kết bạn giữa hai người dùng.

– Bản đồ giao thông: Danh sách các tuyến đường nối giữa các giao lộ có thể được biểu diễn bằng danh sách cạnh. Mỗi cạnh đại diện cho một tuyến đường và có thể bao gồm thông tin về chiều dài, thời gian di chuyển,...

– Lịch trình học tập: Danh sách các môn học được học trong một học kì có thể được biểu diễn bằng danh sách cạnh. Mỗi cạnh đại diện cho mối quan hệ phụ thuộc giữa hai môn học, ví dụ như môn học A phải được học trước môn học B.

3. Danh sách kề

– Danh sách liên lạc: Danh sách liên lạc trong điện thoại có thể được biểu diễn bằng danh sách kề. Mỗi đỉnh đại diện cho một người liên lạc và danh sách kề sẽ liệt kê tất cả các số điện thoại của người liên lạc đó.

– Cấu trúc phân tử: Cấu trúc phân tử có thể được biểu diễn bằng danh sách kề. Mỗi đỉnh đại diện cho một nguyên tử và danh sách kề sẽ liệt kê tất cả các nguyên tử liên kết với nguyên tử đó.

– Mạng máy tính: Danh sách kề có thể được sử dụng để biểu diễn mạng máy tính, với các đỉnh đại diện cho các máy tính và các cạnh đại diện cho các đường cáp nối giữa các máy tính.

Lưu ý: Một đồ thị có thể có nhiều cách biểu diễn khác nhau. Việc lựa chọn cách biểu diễn đồ thị phù hợp sẽ phụ thuộc vào nhiều yếu tố, một trong các yếu tố quan trọng nhất đó là thuật toán được sử dụng để xử lí đồ thị trong từng bài toán cụ thể.

BÀI 13. THỰC HÀNH THIẾT LẬP ĐỒ THỊ

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

1. Kiến thức

Biểu diễn đồ thị bằng ma trận kề và danh sách kề.

2. Năng lực

Thực hành thiết lập đồ thị theo các bộ dữ liệu khác nhau.

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. GV có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Nhiệm vụ 1	Nhiệm vụ 2
Viết chương trình hiển thị danh sách kề.	Viết chương trình hiển thị ma trận kề, danh sách kề và bậc của đồ thị.

Hoạt động khởi động

Phần khởi động GV sẽ cùng 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.

Ví dụ các câu trả lời có thể như sau, không yêu cầu HS trả lời chính xác:

1. Nếu đồ thị là vô hướng thì ma trận kề có đặc điểm gì?

Trả lời: Nếu đồ thị vô hướng thì ma trận kề sẽ là ma trận đối xứng, nghĩa là với mọi đỉnh i, j ta có $A_{ij} = A_{ji}$.

2. Phân biệt sự giống nhau và khác nhau giữa ma trận kề và danh sách kề.

Trả lời: HS có thể nêu sự khác biệt giữa ma trận kề A và danh sách kề Adj:

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?

Trả lời: 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.

Hoạt động thực hành: Nhiệm vụ 1: Viết chương trình hiển thị danh sách kề

Mục đích: Thiết lập được chương trình tính danh sách kề từ dữ liệu đầu vào là ma trận kề.

Tổ chức hoạt động:

GV tổ chức cho các nhóm trình bày yêu cầu, trình bày ý tưởng giải, thuật toán và chương trình giải chi tiết.

Hoạt động thực hành: Nhiệm vụ 2: Viết chương trình hiển thị ma trận kề, danh sách kề và bậc của đồ thị

Mục đích: Thiết lập được chương trình từ đầu vào là danh sách các cạnh, cần tính ma trận kề A, danh sách kề Adj và chi tiết bậc của các đỉnh của đồ thị.

Tổ chức hoạt động:

GV tổ chức cho các nhóm trình bày yêu cầu, trình bày ý tưởng giải, thuật toán và chương trình giải chi tiết.

Hoạt động luyện tập

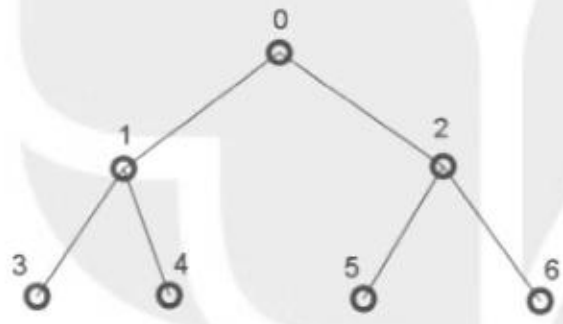
1. Hàm In_danh_sach_dinh_ke() có thể viết lại với dữ liệu đầu vào Adj như sau:

```
1     def In_danh_sach_dinh_ke(Adj):
2         n = len(A)
3         for i in range(n):
4             print("Đỉnh kề với",i,":", end = " ")
5             for j in Adj[i]:
6                 print(j, end = " ")
7             print()
```

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.

Hoạt động vận dụng

1. Ma trận kề của đồ thị tương ứng như sau:



$$A = \begin{pmatrix} 0110000 \\ 1001100 \\ 1000011 \\ 0100000 \\ 0100000 \\ 0010000 \\ 0010000 \end{pmatrix}$$

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

```
1     def BuildGraph(fname):
2         f = open(fname)
3         n = int(f.readline())
4         A = [[0]*n for i in range(n)]
5         V = [i for i in range(n)]
6         E = []
7         Adj = [[] for i in range(n)]
8         for line in f:
9             edge = [int(ch) for ch in line.split()]
10            i,j = edge[0],edge[1]
11            A[i][j] = 1
12            A[j][i] = 1
13            Adj[i].append(j)
```

```
14              Adj[j].append(i)
15              E.append({i,j})
16          f.close()
17          return V,E,A,Adj
```

b) Trường hợp đồ thị có hướng

```
1       def BuildGraph(fname):
2           f = open(fname)
3           n = int(f.readline())
4           A = [[0]*n for i in range(n)]
5           V = [i for i in range(n)]
6           E = []
7           Adj = [[] for i in range(n)]
8           for line in f:
9               edge = [int(ch) for ch in line.split()]
10              i,j = edge[0],edge[1]
11              A[i][j] = 1
12              Adj[i].append(j)
13              E.append((i,j))
14          f.close()
15          return V,E,A,Adj
```

D THÔNG TIN BỔ SUNG

1. Trong bài học này, SCĐ đã đưa ra các phương án sau thiết lập đồ thị từ một tệp dữ liệu ban đầu. GV chú ý với mỗi trường hợp cần xác định rõ phần Input (đầu vào) và Output (yêu cầu dữ liệu đầu ra) và cần phân biệt sự khác biệt giữa kiểu đồ thị vô hướng hay có hướng.

Dữ liệu vào (Input)	Dữ liệu ra (Output)	Phân biệt đồ thị vô hướng và có hướng
Ma trận kề	V, A	Không phân biệt.
Danh sách kề	V, Adj	Không phân biệt.
Danh sách các cạnh	V, A hoặc V, Adj	Phân biệt đồ thị vô hướng và có hướng.

2. Trên thực tế, hay trong các bài tập thực hành, một bộ dữ liệu đầy đủ các thông tin của đồ thị G = (V, E) là bộ gồm các thông tin sau:

n – số lượng đỉnh của đồ thị.

V – danh sách các đỉnh.

E – danh sách các cạnh.

A – ma trận kề.

Adj – danh sách kề.

Trong Python thì các dữ liệu trên thường được mô tả như sau:

V là list các chỉ số đỉnh, đánh số từ 0 đến n – 1. Nếu các đỉnh của đồ thị có tên riêng (ví dụ a, b, c,...) thì cần thêm một mảng nữa để lưu các tên đỉnh này.

E là list các cạnh. Nếu G là đồ thị vô hướng thì mỗi cạnh được biểu diễn bằng kiểu tập hợp 2 đỉnh kề của cạnh đó, ví dụ e = {1, 3}. Nếu G là đồ thị có hướng thì mỗi cạnh được biểu diễn bằng cặp có thứ tự các đỉnh xuất phát và đỉnh kết thúc của cạnh này, ví dụ có thể dùng kiểu dữ liệu list hoặc tuple để mô tả một cạnh. Ví dụ e = [1, 3] hoặc e = (1, 3).

A được biểu diễn bằng mảng 2 chiều kích thước n × n trong Python, mỗi phần tử của ma trận này là 0 hoặc 1. Nếu đồ thị G vô hướng thì A sẽ là ma trận đối xứng.

Adj được biểu diễn bằng một list of list trong Python. Với mỗi đỉnh i, Adj[i] là danh sách các đỉnh kề của đỉnh i, như vậy Adj[i] có dạng là một list.

BÀI 14. KĨ THUẬT DUYỆT ĐỒ THỊ THEO CHIỀU SÂU

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

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

– Trình bày được ý tưởng của duyệt đồ thị theo chiều sâu.

– Mô phỏng được thuật toán duyệt theo chiều sâu.

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. GV có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2	Hoạt động 3
Tìm hiểu ý tưởng của thuật toán duyệt đồ thị theo chiều sâu.	Tìm hiểu thuật toán duyệt DFS.	Tìm hiểu thuật toán duyệt không đệ quy DFS.

Hoạt động khởi động

Mục đích: HS làm quen với bài toán duyệt các đỉnh của đồ thị, có liên hệ với các thuật toán duyệt (tìm kiếm) đã biết trên kiểu dữ liệu mảng (một chiều) và dữ liệu cây.

GV đưa ra bài toán duyệt từng đỉnh của đồ thị cho trước (ví dụ hai đồ thị có hình trong SCĐ) và yêu cầu HS thảo luận, so sánh với các thuật toán duyệt đã biết. GV yêu cầu HS làm thử công việc duyệt trên các đồ thị này và đưa ra nhận xét của mình. Ở phần này chưa yêu cầu HS trả lời chính xác, đây chỉ là phần khởi động.

Hoạt động 1: Tìm hiểu ý tưởng của thuật toán duyệt đồ thị theo chiều sâu

Mục đích: Thông qua một ví dụ cụ thể HS làm quen và biết được cách duyệt đầu tiên: duyệt theo chiều sâu (DFS).

Tổ chức hoạt động:

GV mô tả ý tưởng của thuật toán, sau đó chia lớp thành các nhóm, mỗi nhóm sẽ tiến hành quá trình duyệt "bằng tay" theo một trong hai cách sau:

1. Thực hiện trên sơ đồ, hình ảnh (như Hình 14.2 SCĐ).

2. Thực hiện bằng cách kẻ bảng (theo Bảng 14.1 SCĐ).

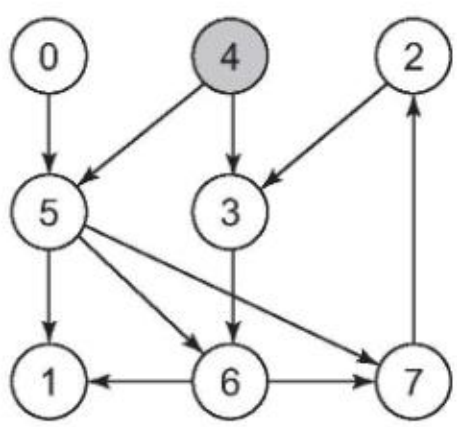
Sau đó các nhóm so sánh kết quả với nhau.

Cuối cùng GV chốt kiến thức và nhấn mạnh thêm ý tưởng "duyệt sâu" của thuật toán.

Hoạt động củng cố kiến thức

1. Có.

2. Trong đồ thị ở Hình 14.1, nếu bắt đầu duyệt theo chiều sâu từ đỉnh 4 thì thứ tự các đỉnh được duyệt sẽ là: 4 3 6 1 7 2 5 0.



Hình 14.1

Hoạt động 2: Tìm hiểu thuật toán duyệt DFS

Mục đích: HS biết và nắm được kĩ thuật cài đặt cụ thể thuật toán duyệt đồ thị theo chiều sâu DFS bằng đệ quy.

Tổ chức hoạt động: GV chia lớp thành các nhóm để đọc, trao đổi, thảo luận cách cài đặt ý tưởng thuật toán DFS trên ngôn ngữ Python.

Các bước mô tả cách cài đặt cụ thể thuật toán như sau:

Bước 1: Thiết lập hàm duyệt DFS bắt đầu từ một đỉnh cụ thể, sử dụng mảng mark[] để đánh dấu, hay duyệt từng đỉnh của đồ thị, bắt đầu từ đỉnh s.

Bước 2: Thiết lập hàm hay chương trình mô tả thuật toán DFS trên toàn bộ đồ thị G.

Bước 3: Phân tích độ phức tạp thời gian của thuật toán trên.

Hoạt động củng cố kiến thức

1. Nếu đồ thị G chỉ bao gồm các đỉnh biệt lập thì tất cả các lệnh DFS(v) sẽ chỉ đánh dấu đúng một đỉnh v. Do vậy thuật toán duyệt theo chiều sâu DFSTraversal(G) sẽ đánh dấu từng đỉnh của G bằng lời gọi hàm DFS(v), mỗi lệnh như vậy đánh dấu đúng một đỉnh.

2. Nếu kiểm tra thì sẽ thấy kết quả dãy các đỉnh đã được duyệt trùng khớp với kết quả duyệt thủ công trong Hoạt động 1.

Hoạt động 3: Tìm hiểu thuật toán duyệt không đệ quy DFS

Mục đích: HS biết và cài đặt được thuật toán duyệt theo chiều sâu không đệ quy sử dụng ngăn xếp.

Tổ chức hoạt động: GV chia nhóm HS và tổ chức cho các nhóm trao đổi, thảo luận và viết chương trình cho thuật toán DFS mới sử dụng ngăn xếp.

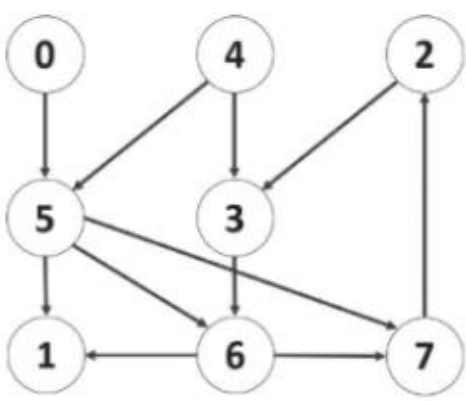
Thuật toán mới này chỉ khác biệt ở hàm duyệt DFS(s) bắt đầu từ đỉnh s cụ thể.

Cuối cùng GV chốt kiến thức.

Lưu ý: Độ phức tạp thời gian của thuật toán DFS không đệ quy giống như thuật toán DFS đệ quy.

Hoạt động 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.



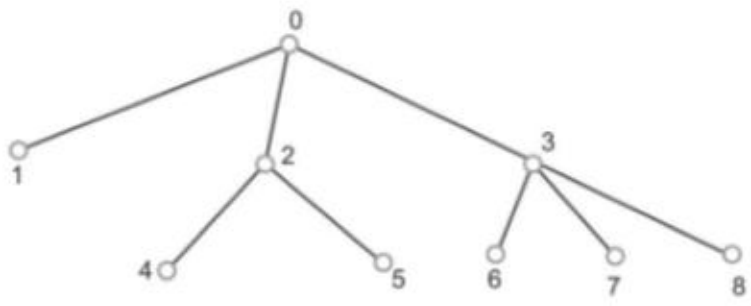

```
8
0 5
1
2 3
3 6
4 3 5
5 1 6 7
6 1 7
7 2
```

Danh sách kề

```
8
0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 1 0 1 0 0
0 1 0 0 0 0 1 1
0 0 0 0 0 0 0 1
0 0 1 0 0 0 0 0
```

Ma trận kề

2. Thứ tự các đỉnh được duyệt theo thuật toán DFS đệ quy 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.




```
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
```

Hoạt động luyện tập

1. Chương trình như sau:

```
from Stack import *
fname = "Graph14.1a_adj.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()]
        line = line[1: ]
        Adj[i].extend(line)
    f.close()
    return V,Adj

def DFS(s):
    S = Stack() # khởi tạo Stack rỗng
    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
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
27                print(v,end = " ")
28                for u in Adj[v]:
29                    push(S,u)
30            else:
31                print("not found",end = "")
32            print()
33
34    # Chương trình chính
35    V,Adj = BuildGraph(fname)
36    n = len(V)
37    mark = [False]*n
38    print("Ngăn xếp                          pop(S)    phần tử đánh dấu")
39    print("-----------------------------------------------------------")
40    for s in V:
41        if not mark[s]:
42            DFS(s)
```

Kết quả chạy chương trình trên với đồ thị Hình 14.1a SCĐ như sau:

```
Ngăn xếp                          pop(S)    phần tử đánh dấu
-----------------------------------------------------------
[0]                               0         0
[5]                               5         5
[0, 1, 6, 7]                      7         7
[0, 1, 6, 2, 5, 6]                6         6
[0, 1, 6, 2, 5, 1, 3, 5, 7]       7         not found
[0, 1, 6, 2, 5, 1, 3, 5]          5         not found
[0, 1, 6, 2, 5, 1, 3]             3         3
[0, 1, 6, 2, 5, 1, 2, 4, 6]       6         not found
[0, 1, 6, 2, 5, 1, 2, 4]          4         4
[0, 1, 6, 2, 5, 1, 2, 3, 5]       5         not found
[0, 1, 6, 2, 5, 1, 2, 3]          3         not found
[0, 1, 6, 2, 5, 1, 2]             2         2
[0, 1, 6, 2, 5, 1, 3, 7]          7         not found
[0, 1, 6, 2, 5, 1, 3]             3         not found
[0, 1, 6, 2, 5, 1]                1         1
[0, 1, 6, 2, 5, 5, 6]             6         not found
[0, 1, 6, 2, 5, 5]                5         not found
[0, 1, 6, 2, 5]                   5         not found
[0, 1, 6, 2]                      2         not found
[0, 1, 6]                         6         not found
[0, 1]                            1         not found
[0]                               0         not found
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

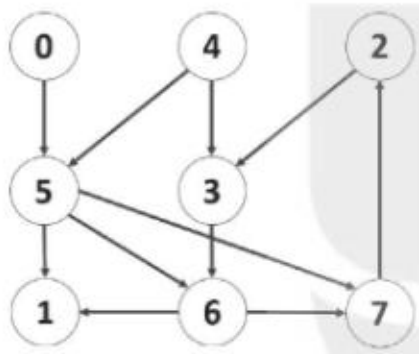
2. a) Chương trình chính trong trường hợp này cần viết lại như sau:

```
1       V,Adj = BuildGraph(fname)
2       n = len(V)
3       mark = [False]*n
4       for s in V:
5               DFS(s)
```

b) 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 SCĐ. Như vậy thuật toán DFS "mới" này hoàn toàn tương đương với thuật toán duyệt DFS đã mô tả trong SCĐ.

Hoạt động vận dụng

1. Đồ thị Hình 14.1b SCĐ có dữ liệu danh sách kề như sau:




```
8
0 5
1
2 3
3 6
4 3 5
5 1 6 7
6 1 7
7 2
```

```
8
0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 1 0
0 0 0 1 0 1 0 0 0
0 1 0 0 0 0 1 1
0 0 0 0 0 0 0 0 1
0 0 1 0 0 0 0 0 0
```

Chương trình nhập dữ liệu từ tệp danh sách kề và thực hiện thuật toán duyệt DFS theo cả hai cách cho đồ thị này có thể làm như sau:

– Hàm đọc dữ liệu từ tệp để trả về bộ V, Adj cho đồ thị có hướng như sau:

```
1       def BuildGraph(fname):
2           """ Thiết lập đồ thị có hướng theo dữ liệu danh sách đỉnh kề """
3           f = open(fname)
4           n = int(f.readline())
5           V = [i for i in range(n)]
6           Adj = [[] for i in range(n)]
7           for i in range(n):
8               line = [int(x) for x in f.readline().split()]
9               Adj[i].extend(line[1:])
10          f.close()
11          return V,Adj
```

Hàm chính của thuật toán duyệt DFS() đệ quy như sau, có in ra các đỉnh đã duyệt:

```
1       def DFS(s): # DFS đệ quy
2           mark[s] = True
3           print(s,end = " ")
```

```
4           for v in Adj[s]:
5               if not mark[v]:
6                   DFS(v)
```

– Hàm chính của thuật toán duyệt DFS() không đệ quy như sau, có in ra các đỉnh đã duyệt:

```
1       def DFS(s):  # DFS không đệ quy
2           S = Stack()
3           push(S,s)
4           while not isEmptyStack(S):
5               v = pop(S)
6               if not mark[v]:
7                   mark[v] = True
8                   print(v,end = " ")
9                   for u in Adj[v]:
10                      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 kề:

```
1       from Stack import *
2       fname = "Graph14.1b_adj.inp"
3       V,Adj = BuildGraph(fname)
4       n = len(V)
5       mark = [False]*n
6       for s in V:
7           if not mark[s]:
8               DFS(s)
```

2. Các thuật toán DFS() nếu viết lại theo tham số là ma trận kề A sẽ được viết như sau:

a) Thuật toán DFS() đệ quy.

```
1       def DFS(A,s):
2           mark[s] = True
3           for j in range(len(A)):
4               if A[s][j] == 1:
5                   if not mark[j]:
6                       DFS(j)
```

b) Thuật toán DFS() không đệ quy (sử dụng Stack).

```
1       def DFS(A,u):
2           S = Stack() # khởi tạo Stack rỗng
3           push(S,u)
4           while not isStackEmpty(S):
5               v = pop(S)
6               if not mark[v]:
```

```
7                   mark[v] = True
8                   for j in range(len(A)):
9                       if A[v][j] == 1:
10                          if not mark[j]
11                              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, tức là 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.

$$p = v_0 = s, v_1, v_2, ..., v_{k-1}, v_k = t.$$

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 là điều phải chứng minh.

D THÔNG TIN BỔ SUNG

1. Toàn bộ phần mô tả thuật toán duyệt đồ thị theo chiều sâu trong SCĐ là mô tả theo bộ dữ liệu V, Adj. Hoàn toàn tương tự, có thể mô tả thuật toán này nếu đồ thị được biểu diễn bằng ma trận kề A. Bài tập vận dụng 2 đã mô tả cách cài đặt DFS theo ma trận kề A. Chú ý hai cách mô tả này hoàn toàn tương tự nhau nhưng độ phức tạp thời gian thực hiện có sự khác biệt và được mô tả trong bảng sau:

Thuật toán	Đồ thị vô hướng	Đồ thị có hướng
DFS sử dụng danh sách kề Adj	O(V + 2E)	O(V + E)
DFS sử dụng ma trận kề A	$O(V^2)$	$O(V^2)$

2. Thuật toán duyệt đồ thị theo chiều sâu DFS trên đồ thị, phương án đệ quy, chính là kĩ thuật duyệt quay lui mà HS đã được học trong sách Chuyên đề học tập Tin học 11 – Định hướng Khoa học máy tính. Khi dạy đến phần này, GV có thể gợi ý và nhắc lại thuật toán duyệt vét cạn quay lui đã học để HS có thể so sánh và tìm ra được sự tương tự giữa hai thuật toán này.

BÀI 15. THỰC HÀNH DUYỆT ĐỒ THỊ THEO CHIỀU SÂU

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

1. Kiến thức

– Khái niệm chu trình.

– Kiểm tra đồ thị cho trước có tồn tại chu trình hay không.

2. Năng lực

– Thực hành duyệt đồ thị theo chiều sâu.

– Kiểm tra được đồ thị có chu trình hay không.

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 nhiệm vụ.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. GV có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Nhiệm vụ
Hệ thống chuyên đề học tập

Hoạt động khởi động

Mục đích: HS được nhắc lại khái niệm chu trình và bài toán kiểm tra một đồ thị cho trước có chu trình hay không.

GV nhắc lại khái niệm chu trình và yêu cầu HS trao đổi nhanh để đưa ra lời giải, sử dụng thuật toán duyệt DFS đã biết.

Hoạt động thực hành: Nhiệm vụ: Hệ thống chuyên đề học tập

Mục đích: 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 trong thực tế.

Tổ chức thực hiện: GV chia nhóm để các nhóm thảo luận về ý tưởng và thuật giải của bài toán kiểm tra sự tồn tại của chu trình trong đồ thị. Từ đó dẫn đến lời giải cụ thể cho nhiệm vụ thực hành của bài học.

Hoạt động luyện tập

1. Không đúng. Hàm kiểm tra trên chỉ áp dụng cho đồ thị có hướng. Với đồ thị vô hướng cần chỉnh 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.

```
1     def DFS_acyclic(Adj,s):
2         S = Stack()
3         push(S,s)
4         while not isEmptyStack(S):
5             v = top(S)
6             if status[v] == 0:
7                 status[v] = 1
8             else:
9                 pop(S)
10                status[v] = 2
11            for u in Adj[v]:
12                if status[u] == 0:
13                    push(S,u)
14                elif status[u] == 1:
15                    return False
16        return True
```

Hoạt động vận dụng

1. Chương trình được sửa lại như sau:

Sửa lại hàm kiểm tra tính không mâu thuẫn của các chuyên đề:

```
1     def DFS_acyclic(s):
2         status[s] = 1 # Đang duyệt
3         push(S,s)
4         for v in Adj[s]:
5             if status[v] == 1:
6                 prev[v] = s
7                 return False
8             elif status[v] == 0:
9                 prev[v] = s
10                if not DFS_acyclic(v):
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
                    return False
        status[s] = 2 # Đã duyệt xong
        return True
```

Bổ sung hàm in dãy các chuyên đề có mâu thuẫn, nếu có:

```
    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,"-->",sep = "",end = "")
        print(u)
```

Cuối cùng là sửa đoạn chương trình chính:

```
    fi = "Data.inp"
    V,Adj = BuildGraph(fi)
    print(Adj)
    n = len(V)
    status = [0]*n
    prev = [None]*n
    S = Stack()
    if Acyclic():
        print("Hệ thống chuyên đề học tập hợp lí.")
    else:
        print("Hệ thống chuyên đề học tập không hợp lí.")
        t = pop(S)
        print("Dãy các chuyên đề có mâu thuẫn:")
        printpath(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.

```
    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:
            edge = [int(ch) for ch in line.split()]
            i,j = edge[0],edge[1]
            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)
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
48
49      # Chương trình chính
50      fi = "graph-1.inp"
51      V,Adj = BuildGraph(fi)
52      n = len(V)
53      status = [0]*n
54      prev = [None]*n
55      S = Stack()
56      if isCyclic():
57          print("Đồ thị có chu trình:")
58          t = pop(S)
59          printpath(t)
60      else:
61          print("Đồ thị không có chu trình.")
```

D THÔNG TIN BỔ SUNG

Tuỳ theo điều kiện cụ thể, GV có thể cung cấp thêm thông tin để HS biết duyệt đồ thị theo chiều sâu (DFS) thường được sử dụng trong một số bài toán sau:

1. Tìm kiếm trên đồ thị:

– Tìm kiếm một đỉnh cụ thể trong đồ thị thoả mãn một điều kiện xác định.

– Tìm tất cả các đỉnh có thể truy cập được từ một đỉnh cho trước.

– Tìm đường đi từ một đỉnh này tới một đỉnh khác (với điều kiện không quay lại các đỉnh đã đi qua).

2. Xác định cấu trúc của đồ thị:

– Kiểm tra tính liên thông của đồ thị.

– Tìm số lượng thành phần liên thông trong đồ thị.

– Xác định chu trình trong đồ thị.

BÀI 16. KĨ THUẬT DUYỆT ĐỒ THỊ THEO CHIỀU RỘNG

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

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

– Trình bày được ý tưởng của duyệt đồ thị theo chiều rộng.

– Mô phỏng được thuật toán duyệt theo chiều rộng.

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. GV có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Hoạt động 1	Hoạt động 2
Làm quen ý tưởng của thuật toán duyệt đồ thị theo chiều rộng.	Tìm hiểu thuật toán duyệt đồ thị theo chiều rộng.

Hoạt động khởi động

Mục đích: HS làm quen với ý tưởng của thuật toán duyệt theo chiều rộng và dự đoán cách thực hiện của thuật toán duyệt đồ thị này.

GV trình bày ý tưởng ban đầu của thuật toán duyệt đồ thị theo chiều sâu và cho các nhóm trao đổi, phát biểu và thử tìm lời giải trên đồ thị của Hình 16.1 SCĐ.

Hoạt động 1: Làm quen ý tưởng của thuật toán duyệt đồ thị theo chiều rộng

Mục đích: HS biết và nắm được ý tưởng của cách duyệt đồ thị theo chiều rộng và thực hiện được việc duyệt đồ thị thủ công.

Tổ chức hoạt động:

GV chia lớp thành các nhóm, tìm thêm một số ví dụ đồ thị và biểu diễn các đồ thị này bằng hình ảnh. GV trình bày nhanh ý tưởng của duyệt theo chiều rộng, sau đó cho các nhóm thực hiện duyệt thủ công và so sánh kết quả.

GV đánh giá từng nhóm đúng hay sai. Cuối cùng GV chốt kiến thức.

Hoạt động củng cố kiến thức

1. Mệnh đề này đúng. Cách chứng minh hoàn toàn tương tự Vận dụng 3 của Bài 14.

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.

Hoạt động 2: Tìm hiểu thuật toán duyệt đồ thị theo chiều rộng

Mục đích: HS biết và thực hiện được thuật toán, chương trình duyệt đồ thị theo chiều rộng sử dụng hàng đợi.

Tổ chức hoạt động: GV chia lớp theo các nhóm. GV hướng dẫn nhanh cách thực hiện thuật toán duyệt theo chiều rộng sử dụng công cụ hàng đợi.

Sau đó, dựa trên một bộ dữ liệu cụ thể (có thể khác với ví dụ trong sách), các nhóm lần lượt điền vào bảng tương tự như Bảng 16.2 SCĐ.

Sau đó, GV so sánh kết quả giữa các nhóm và giới thiệu cách cài đặt cụ thể trên máy tính, so sánh với kết quả thực hiện thủ công.

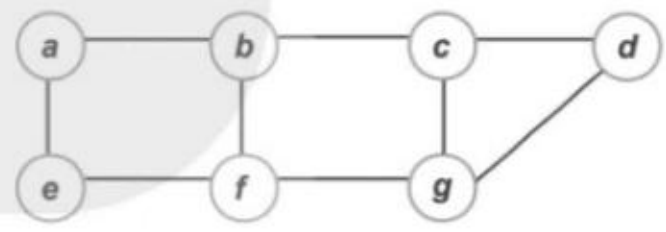
Cuối cùng GV chốt kiến thức.

Hoạt động củng cố kiến thức

1. Có. Thứ tự các đỉnh kề trong Adj sẽ ảnh hưởng đến thứ tự duyệt các đỉnh của 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.



Hoạt động luyện tập

1. Chương trình thực hiện thuật toán duyệt theo chiều rộng và in ra thông tin của Bảng 16.2 SCĐ như sau:

```
11    fname = "graph.inp"
12    from Queue import *
13
14    def BuildGraph(fname):
15        """ Thiết lập đồ thị theo dữ liệu danh sách đỉnh kề """
16        f = open(fname)
17        n = int(f.readline())
18        V = [i for i in range(n)]
19        Adj = [[] for i in range(n)]
20        for i in range(n):
21            line = [int(x) for x in f.readline().split()]
22            Adj[i].extend(line[1:])
23        f.close()
24        return V,Adj
25
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
26      def BFS(s):
27          stt = 0
28          Q = Queue()
29          enqueue(Q,s)
30          while not isEmptyQueue(Q):
31              stt = stt + 1
32              print(stt,":"," "*(3-len(str(stt))),sep = "",end = "")
33              st = str(Q)
34              print(st + " "*(40 - len(st)),end = "")
35              v = dequeue(Q)
36              print(v, " "*(10-len(str(v))), end = "")
37              if not mark[v]:
38                  mark[v] = True
39                  print(v,end = " ")
40                  for u in Adj[v]:
41                      enqueue(Q,u)
42              else:
43                  print("--",end = "")
44              print()
45          print(stt,":"," "*(3-len(str(stt))),sep = "",end = "")
46          st = str(Q)
47          print(st + " "*(40 - len(st)),end = "")
48
49      # Chương trình chính
50      V,Adj = BuildGraph(fname)
51      n = len(V)
52      mark = [False]*n
53      print("Stt Hàng đợi                                    dequeue    Đánh dấu")
54      print("-------------------------------------------------------------")
55      BFS(0)
```

Kết quả chạy trên bộ dữ liệu Hình 16.3 SCĐ cho ta kết quả sau, trùng khớp với Bảng 16.2 SCĐ.

```
Stt Hàng đợi                                    dequeue    Đánh dấu
-------------------------------------------------------------
1:  [0]                                         0          0
2:  [1, 2, 6, 9]                                1          1
3:  [2, 6, 9, 0, 7, 8]                          2          2
4:  [6, 9, 0, 7, 8, 0, 3]                       6          6
5:  [9, 0, 7, 8, 0, 3, 0, 4]                    9          9
6:  [0, 7, 8, 0, 3, 0, 4, 0]                    0          --
```

```
7:  [7, 8, 0, 3, 0, 4, 0]          7          7
8:  [8, 0, 3, 0, 4, 0, 1]          8          8
9:  [0, 3, 0, 4, 0, 1, 1]          0          --
10: [3, 0, 4, 0, 1, 1]             3          3
11: [0, 4, 0, 1, 1, 2, 4]          0          --
12: [4, 0, 1, 1, 2, 4]             4          4
13: [0, 1, 1, 2, 4, 3, 6, 10]      0          --
14: [1, 1, 2, 4, 3, 6, 10]         1          --
15: [1, 2, 4, 3, 6, 10]            1          --
16: [2, 4, 3, 6, 10]               2          --
17: [4, 3, 6, 10]                  4          --
18: [3, 6, 10]                     3          --
19: [6, 10]                        6          --
20: [10]                           10         10
21: [4, 5]                         4          --
22: [5]                            5          5
23: [10]                           10         --
24: []
```

2. Hàm BFS(s) duyệt theo chiều rộng nếu sử dụng ma trận kề sẽ có dạng như sau:

```
1       def BFS(s):
2           Q = Queue()
3           enqueue(Q,s)
4           while not isEmptyQueue(Q):
5               v = dequeue(Q)
6               if not mark[v]:
7                   mark[v] = True
8                   for u in range(len(A)):
9                       if A[v][u] == 1:
10                          enqueue(Q,u)
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

Hoạt động vận dụng

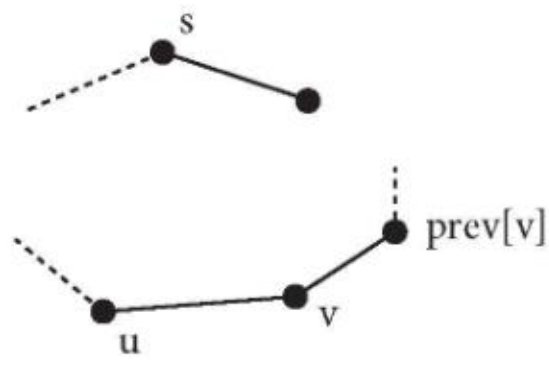
1. Ý tưởng của việc sử dụng BFS phát hiện chu trình trong đồ thị vô hướng G như sau:

Chúng ta sẽ cần sử dụng hai mảng đánh dấu 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: Giả sử chúng ta chạy thuật toán BFS(s), bắt đầu từ s. Trong quá trình duyệt, các đỉnh được duyệt sẽ được đánh dấu mark[] = True. Khi một đỉnh v được duyệt thì prev[v] chỉ ra đỉnh đứng trước v trên đường đi từ s đến v. Nếu 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] (Hình 16.1) thì suy ra đã có một đường đi khác từ s đến u. Vậy kết hợp với đường đi từ s đến u qua v, chúng ta thu được chu trình. Từ đó suy ra thuật toán tìm chu trình theo BFS.



Hình 16.1

Hàm **BFS_cycle(s)** sẽ trả về True nếu tồn tại chu trình đi qua s, ngược lại hàm trả về False:

```
1       def BFS_cycle(s):
2           Q = Queue()
3           mark[s] = True
4           enqueue(Q,s)
5           while not isEmptyQueue(Q):
6               v = dequeue(Q)
7               for u in Adj[v]:
8                   if not mark[u]:
9                       mark[u] = True
10                      prev[u] = v
11                      enqueue(Q,u)
12                  elif prev[v] != u:
13                      return True
14          return False
```

Hàm isCycle() sẽ kiểm tra sự tồn tại chu trình trên toàn bộ đồ thị vô hướng G:

```
1       def isCycle():
2        for v in V:
3        if not mark[v]:
4        if BFS_cycle(v):
5        return True
6        return False
```

Đoạn chương trình chính của lời giải bài toán như sau:

```
1       fname = "graph.inp"
2       from Queue import *
3       V,Adj = BuildGraph(fname)
```

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

```
    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")
```

2. Giả sử, không mất tổng quát, đồ thị vô hướng G = (V, E) được cho trước bởi tập dữ liệu danh sách các cạnh. Chương trình sau sẽ giải bài toán đã cho. Hàm xử lí 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.

find_path_in_undirected_graph.py

```
from Queue import *
def BuildGraph(fname):
    """ Thiết lập đồ thị 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:
        edge = [int(ch) for ch in line.split()]
        i,j = edge[0],edge[1]
        Adj[i].append(j)
        Adj[j].append(i)
    f.close()
    return V,Adj

def BFS(s):
    Q = Queue() # khởi tạo hàng đợi rỗng
    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):
    print(prev)
    if prev[t] == None:
        print("Không tồn tại đường đi từ đỉnh",s,"đến đỉnh",t)
    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(v,"-->",end = "")
            else:
                print(v)

def FindPath(s,t):
    BFS(s)
    printpath(s,t)

# Chương trình chính
fdata = "graph.inp"
V,Adj = BuildGraph(fdata)
mark = [False]*n
prev = [None]*n
s = int(input("Nhập đỉnh thứ nhất: "))
t = int(input("Nhập đỉnh thứ hai: "))
FindPath(s,t)
```

D THÔNG TIN BỔ SUNG

Nhiều bài toán trên đồ thị phải cần sử dụng các kĩ thuật duyệt đồ thị. Duyệt đồ thị theo chiều rộng thường được sử dụng trong các bài toán sau:

1. Tìm kiếm trên đồ thị

– Tìm kiếm tất cả các đỉnh có thể truy cập được từ một đỉnh cho trước.

– Tìm đường đi ngắn nhất từ một đỉnh này tới một đỉnh khác.

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

2. Xác định cấu trúc của đồ thị

– Kiểm tra tính liên thông của đồ thị.

– Tìm số lượng thành phần liên thông trong đồ thị.

So với duyệt đồ thị theo chiều sâu, duyệt đồ thị theo chiều rộng luôn tìm ra được đường đi ngắn nhất giữa hai đỉnh và nó dễ dàng triển khai hơn so với duyệt đồ thị theo chiều sâu. Tuy nhiên, duyệt đồ thị theo chiều rộng có thể cần sử dụng nhiều bộ nhớ hơn so với duyệt theo chiều sâu.

BÀI 17. THỰC HÀNH DUYỆT ĐỒ THỊ TỔNG HỢP

A MỤC TIÊU, YÊU CẦU ĐỐI VỚI HỌC SINH

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

– 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

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.

B YÊU CẦU CHUẨN BỊ

Với những bài mô tả thiết kế thuật toán, GV có thể sử dụng hình ảnh, sơ đồ minh hoạ cho các bước thực hiện trên một mẫu dữ liệu đơn giản. Hoặc có thể sử dụng các phần mềm mô phỏng thuật toán để minh hoạ thêm trong quá trình giảng dạy.

C GỢI Ý CÁC HOẠT ĐỘNG DẠY HỌC CHÍNH

Nhiệm vụ
Tìm đường đi xe đạp

Hoạt động khởi động

Mục đích: HS được gợi ý tìm hiểu các ứng dụng của thuật toán duyệt đồ thị theo chiều rộng.

GV cùng ôn tập, trao đổi với HS về một số ứng dụng của thuật toán duyệt theo chiều sâu đồ thị. Vấn đề là với duyệt theo chiều rộng thì có những ứng dụng gì?

Hoạt động thực hành: Nhiệm vụ. Tìm đường đi xe đạp

Mục đích: HS tìm hiểu, trao đổi để hiểu bài toán cần giải quyết.

Tổ chức thực hành: GV chia nhóm và hướng dẫn HS tìm hiểu bài toán, tìm hiểu cách giải, hướng HS đến tìm lời giải với mô hình dữ liệu đồ thị.

Cuối cùng HS thực hiện tìm hiểu lời giải chi tiết của bài toán.

Hoạt động luyện tập

1. Hàm nhập dữ liệu có thể thay đổi lại như sau:

```
1     def NhapDL(names):
2         ht1 = input("Nhập tên HS thứ nhất: ")
3         ht2 = input("Nhập tên HS thứ hai: ")
4         i = names.index(ht1)
5         j = names.index(ht2)
6         return i,j
```

2. Hàm BFS(s) có thể viết lại sử dụng ma trận kề A như sau:

```
1     def BFS(A,s):
2         Q = Queue() # khởi tạo hàng đợi rỗng
3         mark[s] = True
4         enqueue(Q,s)
5         while not isEmptyQueue(Q):
6             v = dequeue(Q)
7             for u in range(len(A)):
8                 if A[v][u] == 1
9                     if not mark[u]:
10                        mark[u] = True
11                        prev[u] = v
12                        enqueue(Q,u)
```

Hoạt động vận dụng

1. Bổ sung hàm sau kiểm tra và thông báo nếu tồn tại hai bạn HS không thể đi xe đạp đến nhà bạn kia.

```
    def Check():
        kq = True
        for s in range(n):
            for t in range(n):
                mark[ : ] = [False]*n
                prev[ : ] = [None]*n
                BFS(s)
                if s != t and prev[t] == None:
                  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")
```

2. Hàm printpath(s,t) viết dưới dạng không đệ quy như sau:

```
    def printpath(s,t):
        print(prev)
        if prev[t] == None:
            print("Không tồn tại đường đi xe đạp từ nhà bạn",names[s],"đến nhà
    bạn",names[t])
        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])
```

D THÔNG TIN BỔ SUNG

Bài thực hành này là một ứng dụng nhỏ sử dụng thuật toán duyệt theo chiều rộng BFS để tìm đường đi trên đồ thị. Bài toán tìm đường đi trên đồ thị có rất nhiều ứng dụng trong thực tế. GV có thể mở rộng cho HS biết được các bài toán sau liên quan đến tìm đường đi trên đồ thị. Giả sử cho trước đồ thị G = (V, E) có hướng hoặc vô hướng.

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

1. Bài toán tìm đường đi

Cho trước hai đỉnh s, t. Kiểm tra xem có đường đi từ đỉnh s đến t thông qua các cạnh của G hay không. Nếu có thì chỉ ra một cách đi như vậy. Bài toán này đã được giải trong bài thực hành.

2. Bài toán tìm và tính khoảng cách đường đi ngắn nhất

Cho trước hai đỉnh s, t. Kiểm tra xem có đường đi từ đỉnh s đến t thông qua các cạnh của G hay không. Nếu có thì chỉ ra một cách đi với số các cạnh phải đi qua là ít nhất. Gọi $d(v_1, v_2)$ là khoảng cách đường đi ngắn nhất giữa hai đỉnh v_1, v_2. Nếu không tồn tại đường đi thì quy định khoảng cách là ∞.

Như vậy bài toán tìm khoảng cách là cho trước hai đỉnh s, t, cần tính d(s, t).

3. Bài toán tính khoảng cách từ một đỉnh s đến tất cả các đỉnh còn lại của đồ thị G.

4. Bài toán tính đồng thời khoảng cách giữa tất cả các đỉnh.

Cần tính ma trận khoảng cách $D = \{D_{ij}\}$, với $D_{ij} = d(i, j)$.

KẾT NỐI TRI THỨC
VỚI CUỘC SỐNG

Nhà xuất bản Giáo dục Việt Nam xin trân trọng cảm ơn
các tác giả có tác phẩm, tư liệu được sử dụng, trích dẫn trong cuốn sách này.

Chịu trách nhiệm xuất bản:
Chủ tịch Hội đồng thành viên – Tổng Giám đốc NGUYỄN TIẾN THANH

Chịu trách nhiệm nội dung:
Tổng biên tập PHẠM VĨNH THÁI

Biên tập nội dung: LƯU THẾ SƠN – PHẠM THỊ THANH NAM – VŨ THỊ VÂN
Thiết kế sách: NGUYỄN THÀNH TRUNG
Trình bày bìa: NGUYỄN BÍCH LA
Sửa bản in: PHẠM THỊ TÌNH - TRẦN THU HÀ
Chế bản: PHÒNG CHẾ BẢN, CÔNG TY CP DVXB GIÁO DỤC HÀ NỘI

Bản quyền thuộc Nhà xuất bản Giáo dục Việt Nam.
Tất cả các phần của nội dung cuốn sách này đều không được sao chép, lưu trữ, chuyển thể dưới bất kì hình thức nào khi chưa có sự cho phép bằng văn bản của Nhà xuất bản Giáo dục Việt Nam.

CHUYÊN ĐỀ HỌC TẬP TIN HỌC 12 – ĐỊNH HƯỚNG KHOA HỌC MÁY TÍNH – SÁCH GIÁO VIÊN

Mã số:
In ..... cuốn (QĐ ..... SLK), khổ 19 x 26,5cm.
In tại Công ty cổ phần in .....
Số ĐKXB: .....
Số QĐXB: ..... / QĐ-GD ngày ... tháng ... năm ....
Nộp lưu chiểu quý ..... năm .....
Mã số ISBN: 978-604-0-.....



HUÂN CHƯƠNG HỒ CHÍ MINH

BỘ SÁCH GIÁO VIÊN LỚP 12 – KẾT NỐI TRI THỨC VỚI CUỘC SỐNG

1. Ngữ văn 12, tập một – SGV
2. Ngữ văn 12, tập hai – SGV
3. Chuyên đề học tập Ngữ văn 12 – SGV
4. Toán 12 – SGV
5. Chuyên đề học tập Toán 12 – SGV
6. Lịch sử 12 – SGV
7. Chuyên đề học tập Lịch sử 12 – SGV
8. Địa lí 12 – SGV
9. Chuyên đề học tập Địa lí 12 – SGV
10. Giáo dục kinh tế và pháp luật 12 – SGV
11. Chuyên đề học tập Giáo dục kinh tế và pháp luật 12 – SGV
12. Vật lí 12 – SGV
13. Chuyên đề học tập Vật lí 12 – SGV
14. Hoá học 12 – SGV
15. Chuyên đề học tập Hoá học 12 – SGV
16. Sinh học 12 – SGV
17. Chuyên đề học tập Sinh học 12 – SGV
18. Công nghệ 12 – Công nghệ Điện – Điện tử – SGV
19. Chuyên đề học tập Công nghệ 12 – Công nghệ Điện – Điện tử – SGV
20. Công nghệ 12 – Lâm nghiệp – Thuỷ sản – SGV
21. Chuyên đề học tập Công nghệ 12 – Lâm nghiệp – Thuỷ sản – SGV
22. Tin học 12 – SGV
23. Chuyên đề học tập Tin học 12 – Định hướng Tin học ứng dụng – SGV
24. Chuyên đề học tập Tin học 12 – Định hướng Khoa học máy tính – SGV
25. Mĩ thuật 12 – SGV
26. Chuyên đề học tập Mĩ thuật 12 – SGV
27. Âm nhạc 12 – SGV
28. Chuyên đề học tập Âm nhạc 12 – SGV
29. Hoạt động trải nghiệm, hướng nghiệp 12 – SGV
30. Giáo dục thể chất 12 – Bóng chuyền – SGV
31. Giáo dục thể chất 12 – Bóng đá – SGV
32. Giáo dục thể chất 12 – Cầu lông – SGV
33. Giáo dục thể chất 12 – Bóng rổ – SGV
34. Giáo dục quốc phòng và an ninh 12 – SGV
35. Tiếng Anh 12 – Global Success – SGV

Website: https://www.nxbgd.vn

KẾT NỐI TRI THỨC VỚI CUỘC SỐNG