

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TIN HỌC

DÀNH CHO TRUNG HỌC CƠ SỞ

SÁCH GIÁO VIÊN

QUYỂN

3



THƯ VIỆN GIÁO DỤC VIỆT NAM

PHẠM THẾ LONG (Chủ biên)
BÙI VIỆT HÀ - BÙI VĂN THANH

TIN HỌC

DÀNH CHO TRUNG HỌC CƠ SỞ

SÁCH GIÁO VIÊN

(Tái bản lần thứ bảy, có chỉnh lí và bổ sung)

QUYỂN

3



NHÀ XUẤT BẢN GIÁO DỤC VIỆT NAM

PHẦN MỘT. NHỮNG VẤN ĐỀ CHUNG

I. VÀI NÉT CHUNG VỀ MÔN TIN HỌC VÀ SÁCH GIÁO KHOA CHÍNH LÍ

1. Vai trò của môn Tin học

Ở nhà trường phổ thông, môn Tin học có vai trò quan trọng giúp cho học sinh (HS) hình thành và phát triển năng lực sử dụng công nghệ thông tin và truyền thông (ICT). Cụ thể hơn, môn Tin học góp phần hình thành và phát triển các năng lực sau ở HS:

- Năng lực sử dụng, quản lí các công cụ của ICT, khai thác các ứng dụng thông dụng khác của ICT;
- Năng lực nhận biết và ứng xử trong sử dụng ICT phù hợp với chuẩn mực đạo đức, văn hoá của xã hội Việt Nam;
- Năng lực phát hiện và giải quyết vấn đề một cách sáng tạo với sự hỗ trợ của các công cụ ICT, bao gồm tư duy thuật toán, lập trình, điều khiển và tự động hoá;
- Năng lực khai thác các ứng dụng, các dịch vụ của công nghệ kĩ thuật số trong môi trường ICT để học tập có hiệu quả ở các lĩnh vực khác nhau;
- Năng lực sử dụng các công cụ và môi trường ICT để chia sẻ thông tin, hợp tác với mọi người.

Đối với các môn học khác, Tin học cung cấp công cụ, tạo môi trường hỗ trợ giảng dạy, giúp cập nhật những tri thức mới, góp phần nâng cao hiệu quả giáo dục.

Tin học tạo ra môi trường thuận lợi cho học tập suốt đời và học từ xa, làm cho việc trang bị kiến thức, kĩ năng và hình thành nhân cách cho HS không chỉ được thực hiện trong khuôn khổ của nhà trường mà có thể được thực hiện ở mọi lúc mọi nơi.

2. Đặc điểm của môn Tin học

a) Thực hành trên máy tính là yêu cầu bắt buộc trong dạy học bộ môn

Môn Tin học sẽ rất khó dạy khi giáo viên (GV) hoàn toàn không được dùng máy tính để minh họa hay thực hành các thao tác mẫu của bài học. Theo thiết kế của chương trình, mặc dù tập thể tác giả sách giáo khoa (SGK) trong chừng mực cho phép đã cố gắng trình bày các kiến thức của bài học độc lập tối đa với các thao tác cụ thể trên máy tính, song việc học tập của HS vẫn phải phụ thuộc nhiều vào việc minh họa hay trình diễn trên máy tính, nhiều bài học vẫn phải diễn đạt hoàn toàn thông qua các thao tác cụ thể với phần mềm. Do vậy, khi dạy học GV cần chú ý đặc điểm này để chủ động trong việc diễn đạt bài học trong trường hợp không có máy tính trình diễn trên lớp.

b) Kiến thức môn học gắn liền với công nghệ và thay đổi rất nhanh

Đặc thù này làm cho Tin học khác hẳn so với tất cả các môn học có liên quan đến công nghệ hay học nghề khác. Công nghệ thông tin (CNTT), cụ thể là máy tính đã và đang phát triển nhanh chóng, len lỏi vào mọi ngõ ngách của cuộc sống hằng ngày. Điều này đòi hỏi GV phải không ngừng nâng cao trình độ cá nhân của mình để cập nhật những thay đổi của bộ môn nói chung và các phần mềm được đề cập trong SGK nói riêng.

c) Môi trường thực hành rất đa dạng và không thống nhất

Đây cũng là một đặc thù nổi bật của bộ môn Tin học. Chỉ nói riêng hệ điều hành Windows cũng đã có nhiều phiên bản khác nhau hiện đang được dùng tại Việt Nam, ví dụ: Windows XP, Windows Vista, Windows 7, 8, 10. Tương tự như vậy, phần mềm Microsoft Office cũng đang phổ biến nhiều phiên bản khác nhau như Office 2003, 2007, 2010,... Hệ thống cấu hình đĩa đi kèm tại các máy tính cũng rất đa dạng. Máy tính có thể có một, hai hay nhiều ổ đĩa cứng. Trên các máy tính thậm chí có thể cài đặt song song nhiều hệ điều hành khác nhau. Do vậy, GV cần chủ động và hết sức linh hoạt khi giảng dạy. Thông tin trong các tài liệu học tập chỉ mang tính định hướng về kiến thức môn học chứ không áp đặt quy trình thao tác trên máy tính hay một phần mềm cụ thể. Với mỗi bài học, tùy vào điều kiện thực tế mà GV có thể hoàn

toàn chủ động trong việc trình bày khái niệm hay minh họa thao tác trên máy tính sao cho dễ hiểu nhất đối với HS.

d) Tin học là môn học được đưa vào giảng dạy trong nhà trường phổ thông cách đây chưa lâu

Từ các đặc thù trên, khi tổ chức giảng dạy môn học cần lưu ý một số điểm sau:

- (1) Việc giảng dạy môn Tin học trong nhà trường đòi hỏi GV cần phải linh hoạt, do vậy không nên áp đặt các tiêu chuẩn đánh giá cứng nhắc về phương pháp, tiến độ giảng dạy.
- (2) Các nhà trường cần ưu tiên tối đa trang thiết bị cho GV khi giảng dạy môn học này.
- (3) GV dạy môn Tin học cần cập nhật kiến thức thường xuyên. Nhà trường cần tạo điều kiện cho các GV tin học học tập, nâng cao kiến thức và kinh nghiệm.
- (4) Phương pháp giảng dạy cũng cần phải đổi mới và tuân theo các quy chế linh hoạt. Các phương pháp dạy học chủ yếu là phương pháp dạy học tích cực, thực hành; dạy học theo dự án; các hoạt động trải nghiệm sáng tạo.
- (5) Trong việc đánh giá HS cần chú trọng đánh giá năng lực, kỹ năng HS dựa trên kết quả hoạt động, sản phẩm. Do vậy GV nên phối hợp nhiều phương pháp, kỹ thuật đánh giá HS.
- (6) GV có thể lựa chọn các phần mềm học tập khác để dạy cho HS, không bắt buộc phải dạy theo các phần mềm học tập được trình bày trong SGK.

3. Những thay đổi trong lần phát hành này

Với lần sửa chữa, nâng cấp này, các tác giả đã có những thay đổi như sau:

- (1) Thay thế toàn bộ những nội dung liên quan đến các phần mềm phiên bản cũ bằng những phiên bản phần mềm mới hơn đang được dùng phổ biến hiện nay. Tuy nhiên, các phần mềm được sử dụng trong tài liệu này chỉ có tính minh họa cho các chức năng mà HS cần phải học. Do vậy, GV có thể sử dụng những phiên bản phần mềm khác, miễn là phù hợp với điều kiện thực tế dạy

học ở địa phương. Đặc biệt, lưu ý là GV cần căn cứ điều kiện cụ thể để tổ chức việc giảng dạy, nhất là phải cập nhật thường xuyên những thay đổi trong các phiên bản phần mềm để chủ động trong việc truyền tải kiến thức. Các hình ảnh giao diện và trình tự thao tác trong các phiên bản khác nhau của cùng một phần mềm có thể khác so với SGK.

(2) Tại đầu mỗi bài học các tác giả đã bổ sung thêm một tình huống dạy học (tạm gọi là “khởi động”), với mục tiêu tạo tâm thế vui vẻ, kích thích trí tò mò, khơi gợi động cơ giúp HS mong muốn tham gia vào quá trình học tập. GV có thể tổ chức dạy học theo các nội dung đã hướng dẫn hoặc có thể thay thế bằng các nội dung khác phù hợp hơn với điều kiện của mình.

(3) Nội dung chính của mỗi bài học theo chuẩn kiến thức, kỹ năng của Bộ Giáo dục và Đào tạo (GD & ĐT) được trình bày theo trật tự logic, tạo điều kiện để GV đổi mới phương pháp dạy học. Tại mỗi phần, các nội dung quan trọng cần khắc sâu được trình bày dưới dạng chữ in nghiêng để GV có thể lưu ý thêm cho HS. Để dạy những nội dung kiến thức này, GV nên tổ chức giảng dạy tại phòng máy tính. Tuy nhiên, với các trường không đủ máy tính, GV có thể sử dụng phương pháp làm mẫu để HS dễ hình dung và thực hành lại trên máy tính khi có điều kiện.

(4) Phần câu hỏi và bài tập, GV có thể hướng dẫn để các em thực hành ngay trên lớp hoặc ngoài thời gian học trên lớp.

(5) Tại cuối mỗi bài học, các tác giả bổ sung thêm mục “Tìm hiểu mở rộng” nhằm giúp các em HS tìm hiểu và mở rộng thêm kiến thức khi có nhu cầu. Các nội dung này không bắt buộc với tất cả các em. Do vậy, GV có thể hướng dẫn để các em thực hiện các nhiệm vụ này ngoài thời gian học trên lớp.

(6) Bổ sung thêm mục Index ở cuối sách để tiện cho việc tra cứu các từ khóa trong SGK.

4. Phương tiện và thiết bị dạy học

- Sách dành cho HS.
- Máy tính để dành cho thực hành. Ít nhất mỗi nhóm 01 chiếc.

- Máy chiếu (Projector) hoặc ti vi có thể kết nối với máy tính.
- Các phần mềm cần cài đặt trên máy tính:
 - + Phần mềm soạn thảo văn bản **Microsoft Word**;
 - + Phần mềm bảng tính **Microsoft Excel**;
 - + Phần mềm trình chiếu **Microsoft PowerPoint**;
 - + Phần mềm lập trình **Free Pascal**;
 - + Phần mềm luyện tập chuột **Mouse Skills**;
 - + Phần mềm luyện gõ phím **Rapid Typing** và **Typing Master**;
 - + Phần mềm học toán **GeoGebra**;
 - + Phần mềm gõ tiếng Việt **Unikey**;
 - + Phần mềm quan sát Hệ Mặt Trời **Solar System**;
 - + Phần mềm làm quen với giải phẫu người **Anatomy**;
 - + Phần mềm biên soạn âm thanh **Audacity**;
 - + Phần mềm thiết kế phim **Movie Maker**;
- Quy định thư mục, ổ đĩa để lưu bài tập thực hành và các tệp phục vụ học tập.

II. GIỚI THIỆU CHƯƠNG TRÌNH MÔN TIN HỌC

Môn Tin học ở trường phổ thông trang bị cho HS những hiểu biết cơ bản về công nghệ thông tin và vai trò của nó trong xã hội hiện đại. Môn học này giúp HS bước đầu làm quen với phương pháp giải quyết vấn đề theo quy trình công nghệ và kĩ năng sử dụng máy tính phục vụ học tập và cuộc sống. Tin học có vai trò quan trọng trong sự phát triển trí tuệ, tư duy thuật toán, góp phần hình thành học vấn phổ thông cho HS.

Trong hệ thống các môn học ở trường phổ thông, Tin học hỗ trợ cho hoạt động học tập của HS, góp phần làm tăng hiệu quả giáo dục. Tin học tạo ra môi trường thuận lợi cho học tập suốt đời và học từ xa, làm cho việc trang bị kiến thức, kĩ năng và hình thành nhân cách HS không chỉ được thực hiện trong khuôn khổ của nhà trường và các tổ chức đoàn thể, chính trị mà còn có

thể thực hiện ở mọi nơi, mọi lúc. Các kiến thức và kỹ năng trong môi trường học tập này thường xuyên được cập nhật làm cho HS có khả năng đáp ứng những đòi hỏi mới nhất của xã hội.

1. Quan điểm xây dựng chương trình

Tin học là môn học được chính thức đưa vào dạy học ở trường phổ thông chưa lâu nên cần được định hướng và xây dựng chương trình một cách tổng thể về nội dung, phương pháp dạy học, kiểm tra - đánh giá của môn học. Tiếp theo đó, tiến hành xây dựng chương trình cho từng cấp học, lớp học, nhằm đảm bảo tính khoa học, tính sư phạm, đồng thời tránh được lãng phí và tình trạng chồng chéo giữa các cấp học, giữa các môn học của cùng cấp học. Cùng với việc xây dựng chương trình dạy học cần triển khai các hoạt động đồng bộ về chính sách, biên chế GV, phòng máy, xây dựng mạng giáo dục, kết nối Internet, nghiên cứu phương pháp dạy học, đào tạo GV, thiết bị dạy học.

Cũng giống như các môn học khác, việc xây dựng chương trình môn Tin học cần theo đúng quy trình và đảm bảo đầy đủ các thành tố (mục tiêu dạy học, nội dung và chuẩn cần đạt tới, phương pháp và phương tiện dạy học, cách thức đánh giá kết quả).

Tin học là ngành khoa học phát triển rất nhanh, phần cứng và phần mềm thường xuyên thay đổi và được nâng cấp. Vì vậy cần phải trang bị cho HS những kiến thức phổ thông và kỹ năng cơ bản để chương trình không bị nhanh chóng lạc hậu. Tránh cả hai khuynh hướng khi xác định nội dung: hoặc chỉ thiên về lý thuyết mang tính hệ thống chặt chẽ hoặc chỉ thuần túy chú ý tới việc hình thành, phát triển những kỹ năng và thao tác. Tuy nhiên, căn cứ vào đặc trưng của tin học, cần coi trọng thực hành và phát triển kỹ năng, đặc biệt là đối với HS ở các bậc, cấp học dưới.

Cần xuất phát từ điều kiện thực tế của từng địa phương và đặc trưng của môn học để tiến hành tổ chức dạy học một cách linh hoạt, với những hình thức đa dạng để đảm bảo được yêu cầu phổ cập của môn học và nâng cao nếu có điều kiện. Khuyến khích học ngoại khóa.

Chương trình phải có tính “mở”: có phần bắt buộc và phần tự chọn nhằm linh hoạt khi triển khai và dễ dàng cập nhật với thực tế phát triển của môn học.

Một số đặc thù riêng của môn Tin học ở cấp Trung học cơ sở

- a)** Tin học là môn tự chọn bắt buộc dành cho các đối tượng HS Trung học cơ sở (THCS), được dạy cho cả bốn lớp 6, 7, 8 và 9 với thời lượng mỗi tuần hai tiết.
- b)** Môn Tin học đã được đưa vào dạy ở cấp Tiểu học, nhưng dưới hình thức tự chọn không bắt buộc. Vì vậy nội dung môn Tin học ở cấp THCS được xây dựng trên giả thiết là môn học mới.
- c)** Ngoài nội dung lí thuyết, để học môn Tin học HS cần được rèn luyện kĩ năng thông qua thực hành trên máy tính; thậm chí ở lứa tuổi HS THCS phần thực hành còn chiếm thời lượng nhiều hơn. Vì vậy máy tính và phần mềm máy tính (kể cả mạng máy tính) là những dụng cụ học tập không thể thiếu trong giảng dạy tin học. Tại các địa phương, nếu số lượng máy tính còn thiếu, kết nối Internet còn hạn chế, GV nên chủ động tìm các giải pháp tổ chức dạy học sáng tạo để khắc phục.
- d)** Chất lượng đội ngũ GV dạy tin học ở một số địa phương còn có những hạn chế nhất định, nhất là về phương pháp dạy học. Do đó cần chấp nhận sự đầu tư ưu tiên so với các môn học khác trong việc đào tạo, bồi dưỡng GV, trang bị các phương tiện cần thiết cho việc dạy học tin học.
- e)** Có thể khuyến khích hình thức kết hợp với các cơ sở tin học ngoài xã hội, các tổ chức kinh tế, các dự án về tin học, các phương tiện truyền thông đại chúng, tiếp tục phát huy vai trò chủ động, tích cực của các địa phương, các trường để mở rộng khả năng đáp ứng nhu cầu về dạy và học tin học.

2. Mục tiêu

Việc giảng dạy môn Tin học trong nhà trường phổ thông nhằm đạt những mục tiêu sau:

a) Kiến thức

- Trang bị cho HS một cách tương đối có hệ thống các kiến thức cơ bản nhất ở mức phổ thông của khoa học tin học: các kiến thức nhập môn về tin học, hệ thống, thuật toán và ngôn ngữ lập trình, cơ sở dữ liệu, hệ quản trị cơ sở

dữ liệu,... năng lực sử dụng các thành tựu của ngành khoa học này trong học tập và trong các lĩnh vực hoạt động sau này.

- Làm cho HS biết được các lợi ích của công nghệ thông tin cũng như những ứng dụng phổ biến của công nghệ thông tin trong các lĩnh vực khác nhau của đời sống.
- Bước đầu làm quen với cách giải quyết vấn đề có sử dụng công cụ tin học.

b) Kỹ năng

HS có khả năng sử dụng máy tính, phần mềm máy tính và mạng máy tính phục vụ học tập và bước đầu vận dụng vào cuộc sống.

c) Thái độ

- Có tác phong suy nghĩ và làm việc hợp lý, chính xác.
- Có hiểu biết một số vấn đề xã hội, kinh tế, đạo đức liên quan đến tin học.
- Có thái độ đúng đắn và có ý thức ứng dụng tin học trong học tập và cuộc sống.

3. Nội dung chương trình phần III, Tin học dành cho Trung học cơ sở

Phần III

CHỦ ĐỀ	MỨC ĐỘ CẦN ĐẠT	GHI CHÚ
Lập trình đơn giản		
<i>1. Thuật toán và ngôn ngữ lập trình</i>		
Kiến thức <ul style="list-style-type: none"> • Biết được khái niệm bài toán, thuật toán. • Biết rằng có thể mô tả thuật toán bằng cách liệt kê các bước hoặc sơ đồ khối. • Biết được một chương trình là mô tả của một thuật toán trên một ngôn ngữ cụ thể. Kỹ năng <ul style="list-style-type: none"> • Mô tả được thuật toán đơn giản bằng liệt kê các bước. 		– Nên chọn thuật toán của bài toán gần gũi, quen thuộc với HS.
<i>2. Chương trình Pascal đơn giản</i>		
Kiến thức		– Có thể sử dụng ngôn ngữ lập

CHU ĐỀ	MỨC ĐỘ CẦN ĐẠT	GHI CHÚ
<ul style="list-style-type: none"> • Biết sơ bộ về ngôn ngữ lập trình Pascal. • Biết cấu trúc của một chương trình Pascal: cấu trúc chung và các thành phần. • Biết các thành phần cơ sở của ngôn ngữ Pascal. • Hiểu được một số kiểu dữ liệu chuẩn. • Hiểu được cách khai báo biến. • Biết được các khái niệm: phép toán, biểu thức số học, hàm số học chuẩn, biểu thức quan hệ. • Hiểu được lệnh gán. • Biết các câu lệnh vào/ra đơn giản để nhập thông tin từ bàn phím và đưa thông tin ra màn hình. <p>Kĩ năng</p> <ul style="list-style-type: none"> • Viết được chương trình Pascal đơn giản, khai báo đúng biến, câu lệnh vào/ra để nhập thông tin từ bàn phím hoặc đưa thông tin ra màn hình. 		<p>trình khác theo hướng dẫn thực hiện chương trình.</p> <p>– Minh hoạ các khái niệm bằng một chương trình Pascal đơn giản.</p> <p>– Cần xây dựng các bài thực hành và tổ chức thực hiện tại phòng máy để HS đạt được những kĩ năng theo yêu cầu.</p>
3. Tổ chức rẽ nhánh		
<p>Kiến thức</p> <ul style="list-style-type: none"> • Hiểu được câu lệnh rẽ nhánh (dạng thiếu và dạng đủ). • Hiểu được câu lệnh ghép. <p>Kĩ năng</p> <ul style="list-style-type: none"> • Viết đúng các lệnh rẽ nhánh khuyết, rẽ nhánh đầy đủ. • Biết sử dụng đúng và có hiệu quả câu lệnh rẽ nhánh. 		<p>– Nhấn mạnh ba cấu trúc điều khiển là tuần tự, rẽ nhánh và lặp.</p> <p>– Trình bày được thuật toán của một số bài toán rẽ nhánh thường gặp, chẳng hạn giải phương trình bậc nhất.</p>
4. Tổ chức lặp		
<p>Kiến thức</p> <ul style="list-style-type: none"> • Hiểu được câu lệnh lặp kiểm tra điều kiện trước, vòng lặp với số lần định trước. • Biết được các tình huống sử dụng từng loại lệnh lặp. <p>Kĩ năng</p> <ul style="list-style-type: none"> • Viết đúng lệnh lặp với số lần định trước. 		<p>– Kĩ năng chỉ yêu cầu sử dụng lệnh lặp với số lần định trước.</p>
5. Kiểu mảng và biến có chỉ số		
<p>Kiến thức</p> <ul style="list-style-type: none"> • Biết được khái niệm mảng một chiều. 		<p>– Yêu cầu HS viết được chương trình của một số bài toán sau: nhập giá trị phần tử của mảng, in,</p>

CHỦ ĐỀ	MỨC ĐỘ CẦN ĐẠT	GHI CHÚ
<ul style="list-style-type: none"> • Biết cách khai báo mảng, truy cập các phần tử của mảng. Kĩ năng <ul style="list-style-type: none"> • Thực hiện được khai báo mảng, truy cập phần tử mảng, sử dụng các phần tử của mảng trong biểu thức tính toán. 		tính tổng các phần tử.
6. Một số thuật toán tiêu biểu		
Kiến thức <ul style="list-style-type: none"> • Hiểu thuật toán của một số bài toán thường gặp như: tìm số lớn nhất, số nhỏ nhất; kiểm tra ba số cho trước có phải là độ dài ba cạnh của tam giác không. 		
Khai thác phần mềm học tập		
Kiến thức <ul style="list-style-type: none"> • Biết cách sử dụng phần mềm học tập đã lựa chọn. Kĩ năng <ul style="list-style-type: none"> • Thực hiện được các công việc khởi động/ra khỏi, sử dụng bảng chọn, các thao tác tương tác với phần mềm. 		– Lựa chọn phần mềm học tập theo hướng dẫn thực hiện chương trình.

III. GIỚI THIỆU SÁCH GIÁO KHOA TIN HỌC DÀNH CHO TRUNG HỌC CƠ SỞ, QUYỀN 3

1. Định hướng biên soạn

Sách giáo khoa Tin học dành cho Trung học cơ sở – Quyền 3 được biên soạn theo một số định hướng cụ thể sau:

- Thể hiện đúng các nội dung, yêu cầu của chương trình đã được Bộ Giáo dục và Đào tạo phê duyệt là cung cấp cho HS những kiến thức, kĩ năng cơ bản, thiết thực và có hệ thống ban đầu về thuật toán và kĩ thuật lập trình.
- Tiếp cận được trình độ giáo dục phổ thông của các nước tiên tiến trong khu vực và trên thế giới.
- Nội dung sách giáo khoa tập trung vào những kiến thức định hướng để từ đó HS có thể phát huy những yếu tố tích cực của các thành tựu công nghệ thông tin và tăng cường khả năng tự học.

- Nội dung, cách trình bày và diễn đạt ngắn gọn, dễ hiểu thông qua mô tả và các ví dụ minh họa cụ thể.

2. Cấu trúc, nội dung

a) Cấu trúc

Tương ứng với Chương trình giáo dục phổ thông môn Tin học, cấp Trung học cơ sở, phần III, sách giáo khoa gồm hai chương:

Chương I – Lập trình đơn giản: gồm 9 bài lý thuyết, 7 bài thực hành;

Chương II – Phần mềm học tập: gồm 3 bài lý thuyết kết hợp với thực hành.

b) Nội dung

TIN HỌC DÀNH CHO TRUNG HỌC CƠ SỞ, QUYỀN 3

Chương I – Lập trình đơn giản

Bài 1. Máy tính và chương trình máy tính

Bài 2. Làm quen với chương trình và ngôn ngữ lập trình

Bài thực hành 1. Làm quen với Free Pascal

Bài 3. Chương trình máy tính và dữ liệu

Bài thực hành 2. Viết chương trình để tính toán

Bài 4. Sử dụng biến và hằng trong chương trình

Bài thực hành 3. Khai báo và sử dụng biến

Bài 5. Từ bài toán đến chương trình

Bài 6. Câu lệnh điều kiện

Bài thực hành 4. Sử dụng câu lệnh điều kiện

Bài 7. Câu lệnh lặp

Bài thực hành 5. Sử dụng lệnh lặp *for...do*

Bài 8. Lặp với số lần chưa biết trước

Bài thực hành 6. Sử dụng lệnh lặp *while...do*

Bài 9. Làm việc với dãy số

Bài thực hành 7. Xử lý dãy số trong chương trình

Chương II – Phần mềm học tập

Bài 10. Làm quen với giải phẫu cơ thể người bằng phần mềm Anatomy

Bài 11. Giải toán và vẽ hình phẳng với GeoGebra

Bài 12. Vẽ hình không gian với GeoGebra

3. Phân bổ thời lượng

Nội dung	Bài lí thuyết hoặc lí thuyết kết hợp thực hành	Bài thực hành	Tổng số tiết
Chương I. Lập trình đơn giản	9	7	38
Chương II. Phần mềm học tập	3		12
Bài tập			8
Ôn tập			6
Kiểm tra			6
Tổng cộng	12	7	70

- Chương I gồm 9 bài lí thuyết, các bài 1, 2, 3, 4, 6 và 8 dạy trong 2 tiết, bài 5 dạy trong 4 tiết, bài 7 và 9 dạy trong 3 tiết; Có 7 bài thực hành, trong đó các bài thực hành 1, 2, 3, 4 và 6 được dạy trong 2 tiết, bài thực hành 5 và 7 dạy trong 3 tiết.
- Chương II gồm 3 bài lí thuyết kết hợp thực hành, mỗi bài được dạy trong 4 tiết.
- Về cơ bản, 8 tiết bài tập dành cho việc làm bài tập chương I (Lập trình đơn giản), chương II (Phần mềm học tập) không cần tiết bài tập.
- Thời lượng dành cho ôn tập cuối kì là 6 tiết, mỗi học kì 3 tiết.
- Thời lượng dành cho các bài kiểm tra định kì là 6 tiết, mỗi học kì 3 tiết.

Việc phân bổ thời lượng trên đây chỉ là tương đối, trong quá trình dạy học GV có thể điều chỉnh để phù hợp với tình hình thực tiễn.

4. Một số giải thích

a) Sách giáo khoa Tin học dành cho THCS- Quyển 3 được biên soạn bám sát theo nội dung, yêu cầu của Chương trình giáo dục phổ thông môn Tin học, cấp THCS, phần III đã được ban hành kèm theo Quyết định số 16/2006/QĐ-BGDĐT ngày 05/5/2006 của Bộ trưởng Bộ Giáo dục và Đào tạo.

b) Cấu trúc của mỗi bài lí thuyết được xây dựng một cách nhất quán như sau:
Mỗi bài lí thuyết đều được bắt đầu bằng những ví dụ cụ thể dẫn dắt đến cách thức giải quyết các vấn đề trong thực tiễn bằng chương trình máy tính. Bằng cách này HS sẽ dễ thấy hơn mối liên hệ giữa việc lập trình và cuộc sống, cũng như lợi ích của việc lập trình để giải quyết các bài toán bằng máy tính.

Phần nội dung tiếp theo trình bày các thành phần cơ bản hoặc cấu trúc tương ứng của ngôn ngữ lập trình nói chung. Các nội dung này được trình bày ở mức tổng quát nhất có thể, nhưng vẫn đảm bảo HS có thể tiếp thu được. Do đó, khi sử dụng Pascal để minh hoạ trong các mục tiếp theo, SGK không trình bày cú pháp và ngữ nghĩa của các câu lệnh Pascal một cách đầy đủ và chi tiết. GV cần lưu ý điều này để bổ sung cho HS trong các bài thực hành tiếp ngay sau đó.

Sau giới thiệu lí thuyết là một số ví dụ về chương trình để minh hoạ tình huống sử dụng. Về các ví dụ, các tác giả đã hạn chế đến mức tối đa việc lập trình giải quyết các bài toán có nội dung toán học, tránh gây quá tải cho HS.

c) Những nội dung kiến thức, kĩ năng trọng tâm của mỗi bài lí thuyết trình bày khác biệt về màu, kiểu chữ để học sinh dễ ghi nhớ.

d) Mỗi bài lí thuyết đều có phần *Câu hỏi và bài tập* nhằm mục đích cho HS ôn luyện các kiến thức, kĩ năng của bài học lí thuyết và chuẩn bị cho bài thực hành ngay sau đó. Một phần của các câu hỏi, bài tập này nhằm ôn các kiến thức của bài học lí thuyết, vì vậy GV cần hướng dẫn HS làm ngay trên lớp. Phần còn lại dành để giúp HS nắm vững hơn cú pháp và ngữ nghĩa của các câu lệnh Pascal. Với những câu hỏi và bài tập này, GV nên hướng dẫn HS giải và trả lời trong tiết bài tập. Tùy tình hình tiếp thu kiến thức của HS, GV có thể lựa chọn chỉ làm một số bài hoặc chủ động ra các câu hỏi, bài tập phù hợp hơn với trình độ HS, không nhất thiết phải làm hết các câu

hỏi, bài tập trong SGK. Trong phân bổ thời lượng, số tiết bài tập là khá nhiều (8 tiết). Điều này thể hiện câu hỏi, bài tập là một phần quan trọng trong việc giúp HS tiếp thu kiến thức, rèn luyện kỹ năng.

- e) Ngay sau bài lý thuyết (trừ bài 1 và bài 5) là bài thực hành tương ứng với kiến thức lý thuyết đã học. Mục đích chính của các bài thực hành là rèn luyện kỹ năng thực hành trên máy tính cho HS, qua đó củng cố, hiểu sâu hơn các nội dung đã được học ở bài lý thuyết. Các bài thực hành này về cơ bản là để HS thực hành vận dụng những nội dung vừa học ở phần lý thuyết. Tuy nhiên, càng về sau các bài thực hành không chỉ phục vụ cho việc củng cố, thực hành nội dung của bài học lý thuyết tương ứng mà còn giúp ôn luyện những kiến thức, kỹ năng đã được học ở các bài trước đó. Ngoài ra, nội dung *Bài thực hành* còn giới thiệu một số khái niệm mới, kiến thức mới. Một số câu lệnh, thủ tục, hàm được giới thiệu trong bài thực hành. Kiến thức về câu lệnh được rút ra sau khi HS đã được thực hành về câu lệnh.
- f) Mục *Tổng kết* cuối mỗi bài thực hành tóm tắt các kiến thức, kỹ năng cơ bản HS cần tiếp thu được của bài thực hành, chủ yếu là các bước đã thực hiện, cú pháp, ngữ nghĩa cũng như cách sử dụng của Pascal. Phần này giúp HS hệ thống lại những kiến thức, kỹ năng của bài thực hành và là nội dung giúp HS tra cứu nhanh trong quá trình học tập. Tuy nhiên, không nên yêu cầu HS thuộc lòng phần nội dung này. Trong quá trình thực hành, HS sẽ từng bước ghi nhớ cú pháp và ngữ nghĩa các câu lệnh.
- g) Thuật toán có thể biểu diễn bằng sơ đồ khối hoặc bằng cách liệt kê. Tuy nhiên, SGK lựa chọn giới thiệu cách biểu diễn thuật toán bằng cách liệt kê. Về cơ bản, cách liệt kê gần gũi với cách suy nghĩ của HS THCS hơn. Việc so sánh, đối chiếu giữa thuật toán được mô tả bằng cách liệt kê và chương trình tương ứng có thể là dễ dàng hơn với HS. SGK chỉ sử dụng sơ đồ khối để biểu diễn hoạt động của cấu trúc điều khiển (rẽ nhánh, lặp). GV không cần giới thiệu thêm về cách biểu diễn thuật toán bằng sơ đồ khối.
- h) Ngoài việc đảm bảo giới thiệu các kiến thức, kỹ năng theo yêu cầu của chương trình, SGK còn giới thiệu thêm một số nội dung, ví dụ một số công cụ lập trình (câu lệnh, hàm chuẩn, thủ tục chuẩn) nhằm tạo thêm hứng thú cho HS, một số bài toán, thuật toán phổ thông, đơn giản để HS mở rộng thêm kiến thức, kỹ năng. Một số câu hỏi, bài tập, bài thực hành có yêu cầu

cao hơn dành cho các vùng, miền có điều kiện và dành cho đối tượng HS có khả năng tiếp thu tốt.

- i) Các nội dung *Tìm hiểu mở rộng* ở cuối bài là không bắt buộc, tránh yêu cầu tất cả HS phải đọc, hiểu, gây quá tải. GV có thể chọn lựa, giới thiệu, giải thích đôi chút để gây hứng thú cho những em ham thích, đọc thêm.
- j) Về cơ bản, SGK trình bày theo cách tiếp cận các kiến thức, khái niệm cơ bản về lập trình từ khái quát đến cụ thể, sử dụng Pascal làm ngôn ngữ cụ thể để minh họa. Cách tiếp cận này thể hiện rõ mục tiêu chính của chương trình là dạy kiến thức, kỹ năng về lập trình, ngôn ngữ lập trình nói chung, không phải dạy một ngôn ngữ lập trình cụ thể. Khi giảng dạy GV cần lưu ý truyền đạt kiến thức về lập trình là chính, tránh việc sa đà trình bày quá nhiều chi tiết về các thủ thuật với ngôn ngữ Pascal.
- k) Do chỉ là một ngôn ngữ đóng vai trò minh họa cho các kiến thức bắt đầu về lập trình nên các nội dung cụ thể gắn liền với Pascal trong SGK đã được trình bày một cách cô đọng, tăng tính trực quan và giảm tối đa tính hình thức theo nguyên tắc cần đến đâu thì giới thiệu đến đó. SGK không nhằm mục đích giới thiệu các thành phần, kiểu dữ liệu, cú pháp, ngữ nghĩa của các câu lệnh và các đặc trưng khác của Pascal *một cách đầy đủ* như là cẩm nang về lập trình. Chẳng hạn, rất nhiều kiểu dữ liệu, câu lệnh điều kiện *case*, các câu lệnh lặp khác,... không được giới thiệu trong SGK, nhiều câu lệnh chỉ được giới thiệu ngắn gọn mà không đi sâu vào giải thích cú pháp và ngữ nghĩa,...

Các ví dụ và chương trình Pascal cũng được lựa chọn và trình bày theo nguyên tắc trên. Các ví dụ này có thể chưa phải là những chương trình đã được viết một cách gọn nhất hoặc tối ưu nhất. Tuy nhiên, chúng được mô tả và trình bày một cách phù hợp với sự phát triển tư duy của HS sau khi đã được giới thiệu phần kiến thức tương ứng về ngôn ngữ lập trình. Trong quá trình học tập, với sự hướng dẫn của GV, HS có thể chỉnh sửa để có các chương trình tốt hơn, qua đó phát triển tốt hơn các kỹ năng lập trình.

- l) Về thứ tự trình bày các nội dung trong chương I, cần lưu ý một vài điểm sau đây. Trước hết, cần nhấn mạnh rằng việc *xác định bài toán và xây dựng thuật toán là bước quan trọng nhất trong việc viết chương trình*. Chương trình chỉ là thể hiện một thuật toán cụ thể bằng ngôn ngữ lập trình. Chương trình chỉ hoạt động có hiệu quả khi có thuật toán đúng, tối ưu và do vậy chỉ có thể viết được chương trình sau khi đã xây dựng, mô tả thuật

toán. Xét theo thứ tự thời gian và tư duy logic thì nội dung giới thiệu về bài toán và thuật toán cần được giới thiệu ngay từ bài 1.

Tuy nhiên, các tác giả cho rằng nội dung về thuật toán và mô tả thuật toán là vấn đề khó nhất trong toàn bộ nội dung của SGK. Nếu trình bày về thuật toán ngay trong bài 1, khi HS mới bắt đầu một năm học mới, sẽ gây cảm giác quá tải. Hơn thế nữa, việc giới thiệu như vậy có thể làm cho HS hiểu rằng lập trình chỉ là giải toán.

Mặt khác, với tâm sinh lí HS THCS, trước khi giới thiệu những nội dung khó cần bắt đầu dẫn dắt từ những nội dung nhẹ nhàng hơn và dễ gây hứng thú cho HS. Nội dung của các bài từ 1 đến 4 phục vụ mục đích này.

Có một lí do khác là nếu chưa được giới thiệu và chưa hiểu được bản chất của phép gán, HS sẽ rất khó hiểu các nội dung về thuật toán. Trong khi đó, để HS có được khái niệm về phép gán thì cách tốt nhất là giới thiệu các nội dung liên quan đến biến trong lập trình.

Cuối cùng, để HS hiểu được nội dung của các bài từ 1 đến 4 không cần thiết phải có những kiến thức về thuật toán. Các vấn đề giới thiệu trong các bài đó và trong các bài thực hành có thể giải quyết được với những "thuật toán hiển nhiên", hầu như HS nào cũng có thể nhận biết và áp dụng. Tuy nhiên, trước khi đề cập tới các cấu trúc điều khiển trong chương trình thì lại cần thiết phải giới thiệu trước về khái niệm thuật toán và mô tả thuật toán.

Đó chính là lí do nội dung về bài toán và thuật toán được trình bày trong bài 5.

- m)** Nội dung về bài toán và thuật toán (bài 5) được phân bổ thời lượng tương đối nhiều (4 tiết lí thuyết và 2 tiết bài tập). Theo các tác giả, đây là một phần nội dung quan trọng, nếu không nói là quan trọng nhất. Nếu đã nắm vững cách thức mô tả thuật toán để giải quyết bài toán, HS sẽ dễ dàng tiếp thu được các kiến thức trình bày trong những bài tiếp theo.
- n)** Một số kiến thức như cú pháp câu lệnh, cú pháp khai báo biến, kiểu dữ liệu,... được giới thiệu dần dần. Do cách giới thiệu như vậy nên ban đầu có thể chưa đủ, chưa bao quát hết nhưng đảm bảo HS cảm nhận đúng khi mới tiếp cận. Khai báo biến, cú pháp câu lệnh điều kiện, câu lệnh lặp,... được khái quát hoá sau tiết thực hành.
- o)** Đối với HS THCS, để HS dễ tiếp thu, việc trình bày về ngôn ngữ lập trình cần thông qua một ngôn ngữ lập trình cụ thể để minh hoạ, giải thích. Hơn nữa, các kĩ năng lập trình như viết, chỉnh sửa, dịch, chạy và kiểm thử

chương trình đòi hỏi phải sử dụng một ngôn ngữ lập trình bậc cao cụ thể. Khó có thể lựa chọn một ngôn ngữ lập trình cụ thể nào đó đáp ứng cùng lúc được các tiêu chí như: hiện đại, cập nhật, dễ hiểu, dễ dùng, giá thành rẻ (hoặc miễn phí) và tính sư phạm cao nên cách sử dụng Pascal để minh họa trong SGK chỉ là một phương án. Các tác giả rất ủng hộ phương án đa dạng hoá ngôn ngữ lập trình trong quá trình triển khai dạy học, cho phép địa phương, GV tự lựa chọn ngôn ngữ lập trình bậc cao thích hợp với GV, HS, nhà trường.

- p)** Trong SGK lựa chọn ngôn ngữ Pascal để minh họa bởi một số lí do chính sau đây: Ngôn ngữ Pascal là ngôn ngữ lập trình cấu trúc, có tính sư phạm cao; Phần lớn GV Tin học ở cấp THCS hiện nay đã được học và thực hành ngôn ngữ lập trình Pascal là chính; Ngôn ngữ Pascal có nhiều phiên bản chạy được trên hệ điều hành khác nhau hiện có trong trường THCS; Ngôn ngữ lập trình Pascal chạy được trên hầu hết tất cả các máy đã được trang bị ở các trường THCS từ trước đến nay; Việc cài đặt Pascal là dễ dàng và ngôn ngữ Pascal có thể được sử dụng miễn phí.
- q)** Việc phân bổ thời lượng cho các bài lí thuyết, thực hành là tương đối, GV có thể phối hợp với các tiết bài tập, ôn tập để tự cân đối thời lượng cho phù hợp với tình hình giảng dạy thực tiễn. Điều quan trọng là đảm bảo truyền đạt đúng, đủ kiến thức, kĩ năng theo yêu cầu của Chương trình.

IV. GỢI Ý VỀ TỔ CHỨC DẠY HỌC

1. Phương pháp dạy học

- a)** Về cơ bản, SGK lựa chọn phương án trình bày kiến thức, kĩ năng chung về lập trình và sử dụng ngôn ngữ Pascal để minh họa. Cách tiếp cận này thể hiện rõ việc dạy lập trình nói chung mà không phải là dạy ngôn ngữ lập trình cụ thể Pascal. Tuy nhiên, khi giảng dạy GV không nhất thiết phải trình bày theo cách tiếp cận này. Có thể tiếp cận bằng cách đi từ ngôn ngữ lập trình cụ thể Pascal rồi khái quát thành những kiến thức, kĩ năng của lập trình nói chung. Cách tiếp cận từ cụ thể đến khái quát có thể sẽ phù hợp hơn với phần lớn HS THCS. Trong sách giáo viên (SGV), nội dung của từng bài cụ thể được gợi ý về cách dạy học theo hướng từ cụ thể đến khái quát.
- b)** Do sử dụng ngôn ngữ Pascal để minh họa, thời lượng làm việc với các câu lệnh, chương trình, phần mềm Free Pascal (FP) là khá nhiều nên để cảm

nhận là đang học ngôn ngữ Pascal. Vì vậy, trong quá trình dạy học GV cần lưu ý tiến hành khái quát đúng lúc, đúng chỗ để HS vượt ra khỏi một ngôn ngữ cụ thể, rút ra được những kiến thức, kỹ năng, nguyên lý của lập trình nói chung. Trong SGK có hướng dẫn thời điểm khái quát hoá kiến thức, kỹ năng ở một số bài học cụ thể.

- c) Các chương trình được viết khi học ở tiết lý thuyết, tiết bài tập cần để HS chạy thử ở bài thực hành ngay sau đó. Làm như vậy sẽ giúp HS củng cố, hiểu rõ hơn về nội dung lý thuyết vừa học. Hơn nữa, việc này sẽ giúp tạo hứng thú, củng cố niềm tin cho HS, gắn kết tốt hơn giữa học với hành. Để tránh HS mất nhiều thời gian vào việc gõ chương trình, GV nên sử dụng các chương trình được viết trong giờ lý thuyết, giờ bài tập để HS chỉnh sửa, chạy thử, tìm hiểu trong giờ thực hành, không nên yêu cầu HS gõ tất cả các chương trình này trong tiết thực hành.
- d) Trong phân bổ thời lượng dành 8 tiết bài tập, 6 tiết để ôn tập. Các tiết này chưa được định nội dung cụ thể, GV hoàn toàn chủ động đưa ra nội dung cho tiết bài tập, ôn tập. Tuy nhiên, tiết bài tập nên dành thời gian để hướng dẫn HS làm một số bài tập trong SGK (nếu trong tiết lý thuyết chưa làm hết), chuẩn bị cho những bài thực hành sau đó. Tùy mức độ tiếp thu của HS, GV có thể ra thêm các bài tập, bổ sung bài thực hành trên máy tính để HS ôn luyện kiến thức, kỹ năng. Các tiết ôn tập nên được bố trí vào cuối kì (ngay trước hoặc ngay sau bài kiểm tra cuối học kì), trong tiết ôn tập cần tổng kết, khái quát những kiến thức, kỹ năng trọng tâm của chương trình để HS khắc sâu, ghi nhớ. Đặc biệt, tiết ôn tập cần khái quát hoá để thể hiện được tư tưởng dạy lập trình mà không dạy ngôn ngữ lập trình cụ thể.
- e) Trong SGK có gợi ý mô tả một số thuật toán theo cách biểu diễn gần với câu lệnh mà HS cần viết hoặc cần tìm hiểu trong chương trình tương ứng. GV có thể tham khảo, lựa chọn cách mô tả này để giảng dạy phù hợp với đối tượng HS của mình.
- f) Các bài toán được giới thiệu trong SGK nói chung là đơn giản, có thể viết chương trình mà không gặp nhiều khó khăn. Đối với một bài toán cụ thể, nhiệm vụ của HS là viết được chương trình. Tuy nhiên, qua các bài toán HS cần hiểu và thực hiện được các bước giải bài toán trên máy tính: Xác định bài toán, mô tả thuật toán và viết chương trình. Do vậy, cần thực hiện đầy đủ các bước đi từ bài toán đến chương trình: Xác định input, output của bài toán, xây dựng, mô tả thuật toán bằng cách liệt kê và viết chương trình.

- g)** Có một thực tế là một số câu lệnh (nhất là câu lệnh có cấu trúc) thường được giới thiệu gắn liền với một số bài toán, thuật toán điển hình nào đó. Cách làm này có thuận lợi là HS vừa học được câu lệnh mới vừa nắm được bài toán, thuật toán mới. Tuy nhiên, đối với một số HS việc cùng lúc phải học cả hai nội dung mới không phải lúc nào cũng dễ dàng. Để giảm bớt khó khăn cho HS, nên tách việc dạy câu lệnh mới với việc giới thiệu thuật toán mới, nghĩa là dạy xong câu lệnh rồi đến thuật toán hoặc ngược lại. Trong SGK có giới thiệu một số cách làm như vậy, bài toán sử dụng để giới thiệu hoặc áp dụng câu lệnh mới thường dễ hoặc HS đã biết bài toán, thuật toán từ trước. Khi đó, HS chỉ còn duy nhất nhiệm vụ tìm hiểu câu lệnh, không phải mất thời gian để hiểu bài toán, thuật toán. HS chỉ cần tập trung tìm hiểu câu lệnh mới. Ngược lại, khi giới thiệu thuật toán mới thì cần sử dụng câu lệnh HS đã biết sử dụng, lúc đó HS cũng chỉ tập trung vào tìm hiểu thuật toán mới. Hi vọng cách làm như vậy sẽ tạo thuận lợi để HS tiếp thu kiến thức nhẹ nhàng hơn.
- h)** SGK được in màu, hình thức đẹp, các tranh, ảnh trình bày trong SGK đã được chọn lọc, cân nhắc kỹ lưỡng. Vì vậy, cần khai thác tối đa SGK trong quá trình dạy học. Một trong những việc GV có thể thực hiện ngay trong lớp học đó là hướng dẫn HS và dành thời gian cho HS tự nghiên cứu nội dung SGK. Ban đầu việc giao bài cho HS đọc có thể mất thời gian, nhưng khi kỹ năng đọc hiểu của HS được cải thiện thì việc dành thời gian để các em tự đọc có thể sẽ không những không mất thời gian mà ngược lại sẽ tiết kiệm thời gian.
- i)** Việc dạy các phần mềm học tập sẽ hiệu quả hơn nếu tiến hành tại phòng máy tính. Nhưng khi dạy lập trình không nên lạm dụng phòng máy tính. Tiết thực hành cơ bản là để HS chạy thử chương trình, rèn luyện kỹ năng làm việc với môi trường lập trình. Không để tình trạng vào tiết thực hành HS mới biết bài toán và viết chương trình mà chưa chuẩn bị trước.
- j)** Việc giới thiệu phần mềm học tập nhằm mục đích chính là cung cấp cho HS kiến thức, rèn luyện kỹ năng khai thác phần mềm. Bên cạnh đó, việc khai thác phần mềm học tập còn nhằm mục đích tạo sự thay đổi, gây thêm hứng thú học tập. Do vậy, mặc dù SGK trình bày hai phần tách biệt nhưng không có nghĩa là phải dạy theo đúng tuần tự trình bày các bài trong SGK. Nội dung của chương II (Phần mềm học tập) có thể được dạy xen kẽ với chương I (Lập trình đơn giản).
- k)** Khuyến khích GV tăng cường sử dụng các thông tin hoặc phần mềm miễn phí trên Internet để làm phong phú thêm nội dung bài giảng. Tuy nhiên,

trong quá trình đó, không thể tránh khỏi các thông tin nhạy cảm hoặc không chính xác so với chủ trương, đường lối của Đảng và Nhà nước, do vậy GV cần chủ động phát hiện để định hướng kịp thời, tránh những nhận thức sai lầm cho HS.

2. Thiết bị dạy học

- a)* Bộ Giáo dục và Đào tạo đang tiến hành xây dựng, ban hành danh mục thiết bị dạy học tối thiểu môn Tin học cấp THCS. Theo đó, các trường THCS phải đáp ứng được danh mục thiết bị dạy học ít nhất này thì mới có thể tổ chức dạy học môn Tin học. Dự kiến danh mục thiết bị dạy học tối thiểu quy định mỗi trường THCS phải có ít nhất một phòng máy với 25 máy vi tính nối mạng và kết nối Internet. Ngoài máy tính, danh mục còn có các tranh, ảnh được phóng to để dạy học.
- b)* Phần lớn các nội dung dạy học Tin học THCS sẽ rất hiệu quả khi sử dụng các thiết bị trình chiếu, do vậy máy chiếu projector, máy chiếu overhead, máy chiếu vật thể,... là các thiết bị được khuyến khích trang bị để dạy học cho môn Tin học.
- c)* Trong SGK sử dụng phần mềm FP để minh họa. Phần mềm FP được chọn để thay thế cho phần mềm TP trong các lần xuất bản trước. Tuy nhiên trên thực tế GV vẫn có thể sử dụng phiên bản TP để dạy cho học sinh.
- d)* Những trường được trang bị hệ thống Hishare (một CPU kết nối với nhiều màn hình) thì nên sử dụng FP.
- e)* Hiện nay ở một số trường THCS có thể còn có những máy tính cấu hình thấp đã được trang bị từ trước. Những máy tính này hoàn toàn vẫn có thể được sử dụng để thực hành với phần mềm Turbo Pascal for DOS. Do vậy, cần rà soát, tận dụng các máy tính cũ để phục vụ cho các tiết thực hành.
- f)* Hiện nay có một số phần mềm hỗ trợ cho việc quản lý dạy học trên phòng máy tính như: XClass, Magic Class, NetOpSchool, E-Learning Class,... Hơn thế nữa, các phần mềm này còn giúp khai thác phòng thực hành môn Tin học như một phòng đa phương tiện để dạy học các môn học khác.

3. Ôn tập và kiểm tra

- a)* Thời lượng dành cho ôn tập là 6 tiết (3 tiết/học kì). Căn cứ vào tình hình thực tế của lớp học, GV tự xác định nội dung các tiết ôn tập. Tuy nhiên, nên dành các tiết ôn tập để ôn luyện, tổng kết kiến thức, kỹ năng trọng tâm

của chương trình. Trong các tiết ôn tập GV cần khái quát kiến thức, kỹ năng lập trình nói chung thể hiện rõ mục tiêu, trọng tâm của chương trình.

- b)** Thời lượng để kiểm tra, đánh giá là 6 tiết, mỗi học kì 3 tiết. Có thể dành 2 tiết cho bài kiểm tra cuối học kì, tiết còn lại dành cho các bài kiểm tra định kì trong học kì. Nếu tiến hành hai bài kiểm tra định kì trong mỗi học kì, thì nên có một bài kiểm tra trên giấy, một bài kiểm tra thực hành trên máy.
- c)** Nội dung kiểm tra phải đảm bảo cả lý thuyết và thực hành. Cần lựa chọn nội dung kiểm tra để đảm bảo bao quát hết kiến thức, kỹ năng trọng tâm của chương trình.
- d)** Một số nội dung trong chương I. *Lập trình đơn giản* thuận lợi cho việc áp dụng phương pháp trắc nghiệm khách quan trong kiểm tra, đánh giá. Vì vậy, cần lưu ý tăng cường sử dụng trắc nghiệm khách quan trong kiểm tra, đánh giá nội dung này.
- e)** Việc kiểm tra, đánh giá có tác động đến quá trình dạy học. Để định hướng học tập đúng cho HS, bên cạnh việc kiểm tra những kiến thức, kỹ năng gắn liền với ngôn ngữ lập trình cụ thể, cần dành một tỉ lệ thích đáng cho câu hỏi, bài tập về kiến thức, kỹ năng lập trình nói chung. Những câu hỏi, bài tập này sẽ giúp HS có ý thức chú trọng đến kiến thức, kỹ năng lập trình nói chung, tránh làm cho HS chỉ chú trọng đến chi tiết cụ thể của ngôn ngữ lập trình Pascal.
- f)** Cần tiến hành đánh giá HS trong giờ thực hành, điểm này lấy làm điểm kiểm tra thường xuyên (hệ số 1). Trong tiết thực hành có thể đánh giá, cho điểm cả lớp, một nhóm hoặc một vài HS. Tuy nhiên, cần lưu ý mục tiêu của giờ thực hành là để HS thực hành, không phải là giờ kiểm tra. Kiểm tra trong giờ thực hành là để HS tập trung, chăm chỉ, nghiêm túc học tập.
- g)** Việc kiểm tra, đánh giá môn Tin học cấp THCS được thực hiện theo Quy chế đánh giá, xếp loại HS Trung học cơ sở và HS Trung học phổ thông (Ban hành kèm theo Quyết định số: 40/2006/QĐ-BGDĐT ngày 05 tháng 10 năm 2006 của Bộ trưởng Bộ Giáo dục và Đào tạo).

PHẦN HAI. NHỮNG VẤN ĐỀ CỤ THỂ

CHƯƠNG I. LẬP TRÌNH ĐƠN GIẢN

I. GIỚI THIỆU

1. Mục tiêu

Mục tiêu của chương này là cung cấp cho HS một số kiến thức, kỹ năng cơ bản, phổ thông về lập trình thông qua ngôn ngữ lập trình bậc cao Pascal.

a) Kiến thức

- Biết được khái niệm bài toán, thuật toán, mô tả thuật toán bằng cách liệt kê;
- Biết được một chương trình là mô tả của một thuật toán trên một ngôn ngữ cụ thể;
- Hiểu thuật toán của một số bài toán đơn giản (tìm số lớn nhất, số nhỏ nhất; kiểm tra ba số cho trước có phải là độ dài ba cạnh của một tam giác không);
- Biết cấu trúc của một chương trình, một số thành phần cơ sở của ngôn ngữ lập trình;
- Biết một số kiểu dữ liệu chuẩn, đơn giản, cách khai báo biến;
- Biết các khái niệm: phép toán, biểu thức số học, hàm số học chuẩn, biểu thức quan hệ;
- Hiểu được lệnh gán;
- Biết các câu lệnh vào/ra đơn giản để nhập thông tin từ bàn phím và đưa thông tin ra màn hình;
- Hiểu được câu lệnh điều kiện, câu lệnh ghép, vòng lặp với số lần biết trước, câu lệnh lặp kiểm tra điều kiện trước;
- Biết được các tình huống sử dụng từng loại lệnh lặp;
- Biết được khái niệm mảng một chiều kiểu dữ liệu số, cách khai báo mảng, truy cập các phần tử của mảng.

b) Kỹ năng

- Mô tả được thuật toán đơn giản bằng liệt kê các bước;

- Viết được chương trình đơn giản, khai báo đúng biến, câu lệnh vào/ra để nhập thông tin từ bàn phím hoặc đưa thông tin ra màn hình;
- Viết đúng các lệnh rẽ nhánh khuyết, rẽ nhánh đầy đủ;
- Biết sử dụng đúng và có hiệu quả câu lệnh điều kiện;
- Viết đúng lệnh lặp với số lần biết trước;
- Thực hiện được khai báo mảng kiểu dữ liệu số, truy cập phần tử mảng, sử dụng các phần tử của mảng trong biểu thức tính toán.

c) Thái độ

- Nghiêm túc trong học tập, ham thích lập trình trên máy tính để giải các bài tập.

2. Nội dung chủ yếu

- Sự cần thiết phải lập trình giải các bài toán trên máy tính;
- Bài toán, thuật toán, các bước để giải bài toán trên máy tính;
- Một số kiểu dữ liệu chuẩn, đơn giản;
- Cách khai báo biến;
- Phép toán, biểu thức số học, hàm số học chuẩn, biểu thức quan hệ, câu lệnh gán;
- Các thủ tục vào/ra đơn giản;
- Cấu trúc điều khiển rẽ nhánh, lặp (*if-then*, *for-do* và *while-do*);
- Câu lệnh ghép *begin-end*;
- Kiểu mảng một chiều kiểu dữ liệu số; Truy cập, xử lý phần tử mảng một chiều;
- Một số thuật toán đơn giản: thuật toán tìm số lớn nhất, nhỏ nhất,

II. HƯỚNG DẪN CHI TIẾT

Bài 1. MÁY TÍNH VÀ CHƯƠNG TRÌNH MÁY TÍNH

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Biết con người chỉ dẫn cho máy tính thực hiện công việc thông qua lệnh;
- Biết chương trình là cách để con người chỉ dẫn cho máy tính thực hiện nhiều công việc liên tiếp một cách tự động;

- Biết rằng viết chương trình là viết các lệnh để chỉ dẫn máy tính thực hiện các công việc hay giải một bài toán cụ thể;
- Biết ngôn ngữ được dùng để viết chương trình máy tính gọi là ngôn ngữ lập trình;
- Biết vai trò của chương trình dịch.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Khái niệm về lệnh, nút lệnh HS đã được biết đến ở Quyển 1 và Quyển 2. Dựa trên hiểu biết có sẵn của HS về lệnh, GV cần nhắc để HS nhớ lại và hình dung về lệnh một cách đơn giản, phổ thông.

HS đã thực hiện các thao tác khởi động/thoát khỏi phần mềm, sao chép, di chuyển và thực hiện các bước để tắt máy tính (theo đúng trình tự). Đặc biệt, HS vẫn thường xuyên sử dụng các nút lệnh trên dải lệnh để làm việc với chương trình soạn thảo văn bản, chương trình bảng tính. Khi thực hiện các thao tác này chính là HS ra lệnh cho máy tính thực hiện một công việc nào đó. Ví dụ, khi thực hiện thao tác nháy vào nút lệnh **Cut** trên dải lệnh **Home** của Word là đã ra lệnh cho máy tính thực hiện công việc xoá một phần văn bản và lưu vào bộ nhớ của máy tính.

Thực ra khái niệm về lệnh trong máy tính khá phức tạp, tuy nhiên ở đây không nên giới thiệu sâu về lệnh, mà chỉ nên cho HS thấy lệnh máy tính là một chỉ dẫn của con người để máy tính thực hiện một công việc cụ thể nào đó.

b) Sau phần khởi động, HS cần biết được con người điều khiển máy tính thông qua lệnh.

Cần cho HS nhận thấy sự khác biệt giữa việc ra lệnh cho máy tính với ra lệnh cho con người. Qua ví dụ điều khiển rô-bốt nhặt rác, GV cần cho HS nhận thấy một công việc rất đơn giản với con người, nhưng khi muốn máy tính thực hiện thì cần phải chia thành nhiều thao tác nhỏ, đơn giản, cụ thể mà rô-bốt có thể thực hiện được.

Có hai cách để có thể điều khiển rô-bốt thực hiện công việc trên: Cách thứ nhất là ra từng lệnh và rô-bốt thực hiện từng thao tác; Cách thứ hai là chỉ dẫn để rô-bốt tự động thực hiện lần lượt các thao tác trên. Việc viết các lệnh để điều khiển, chỉ dẫn rô-bốt (hay máy tính) thực hiện tự động một loạt các thao tác liên tiếp chính là viết *chương trình máy tính* (hay còn gọi tắt là chương trình).

c) Kết thúc mục 1, HS cần biết chương trình là cách để con người chỉ dẫn cho máy tính thực hiện nhiều công việc liên tiếp một cách tự động và viết chương

trình là viết các lệnh để chỉ dẫn máy tính thực hiện các công việc hay giải một bài toán cụ thể.

Có thể dẫn dắt HS tiếp cận khái niệm ngôn ngữ lập trình như sau: Chương trình mà con người viết ra phải đảm bảo máy tính có thể "hiểu" được. HS đã biết máy tính chỉ có thể hiểu được ngôn ngữ máy (dãy các bit, tức là dãy số 0 và 1). Vì vậy, về nguyên tắc để máy tính "hiểu" được phải viết chương trình bằng ngôn ngữ máy.

Vấn đề là ngôn ngữ máy lại rất khó hiểu, khó nhớ đối với con người nên khi sử dụng ngôn ngữ này để viết chương trình người lập trình rất vất vả.

Do đó xuất hiện nhu cầu cần có một ngôn ngữ trung gian giữa con người và máy tính làm sao để con người dễ dàng sử dụng khi viết chương trình và máy tính cũng có thể hiểu được. Ngôn ngữ lập trình bậc cao được ra đời để đáp ứng nhu cầu này. Có thể liệt kê ra một số ngôn ngữ lập trình bậc cao như Pascal, C, Java...

d) Qua mục 2, như đã nêu ở trên, máy tính chỉ hiểu được ngôn ngữ máy, nên chương trình viết bằng ngôn ngữ lập trình phải được chuyển sang thành chương trình ở ngôn ngữ máy. Điều này cũng giống như việc phiên dịch khi trao đổi với người nước ngoài vậy. Chương trình đóng vai trò dịch từ ngôn ngữ lập trình bậc cao sang ngôn ngữ máy gọi là "chương trình dịch".

Như vậy, để có được một chương trình mà máy tính có thể thực hiện được cần qua hai bước:

(1) *Viết* chương trình theo ngôn ngữ lập trình;

(2) *Dịch* chương trình thành ngôn ngữ máy để máy tính hiểu được.

Cần lưu ý rằng, các bước nêu trên chỉ là hai trong số rất nhiều bước (giai đoạn) để tạo ra một chương trình cụ thể có thể "chạy" trên máy tính. Để có một chương trình hoạt động hiệu quả trên máy tính và phục vụ đúng mục tiêu, người ta còn phải thực hiện nhiều công việc khác, bắt đầu từ việc khảo sát nhu cầu của người dùng (xác định mục tiêu, yêu cầu, khảo sát các quy trình nghiệp vụ,...), phân tích, thiết kế, lập trình, kiểm thử, triển khai cài đặt, đào tạo, hỗ trợ,... Hai bước nói trên chỉ là một phần của công việc lập trình, sau khi thuật toán đã được xây dựng. Tuy nhiên, đối với HS mới bắt đầu làm quen với lập trình và ngôn ngữ lập trình, cách giới thiệu như trong SGK là đủ và nhằm để HS phân biệt được hai công việc viết chương trình và dịch chương trình. Trong các bài thực hành với Pascal, HS sẽ phân biệt rõ hơn hai bước này.

Có thể nói nhiệm vụ chính của ngôn ngữ lập trình là dịch chương trình đã được soạn thảo sang ngôn ngữ máy. Tuy nhiên, một ngôn ngữ lập trình thường cung cấp một số công cụ đi kèm với chương trình dịch để hỗ trợ người lập trình như: phần mềm soạn thảo văn bản; phát hiện và thông báo lỗi; công cụ theo dõi, gỡ rối chương trình; các thư viện chương trình chuẩn;... Các dịch vụ, công cụ này tạo nên môi trường lập trình. GV không cần giải thích kỹ về môi trường lập trình với HS.

e) Ở mục này, HS cần ghi nhớ được ngôn ngữ lập trình là công cụ để viết chương trình máy tính và chương trình dịch đóng vai trò dịch chương trình viết bằng ngôn ngữ lập trình sang ngôn ngữ máy.

Nội dung của bài 1 được xem như là một cầu nối cho HS từ người dùng sang người xây dựng chương trình máy tính. Qua bài này HS cần biết phía sau những thao tác như nháy nút lệnh **Cut**, **Copy**,... mà các em đã quen sử dụng là những chương trình máy tính tương ứng. Khi nháy chuột vào một nút lệnh là các em đã yêu cầu máy tính thực hiện một chương trình tương ứng đã được viết sẵn.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. Lệnh "Tìm kiếm và thay thế" trong phần mềm soạn thảo có thể được hiểu là dãy các lệnh sau:

- 1) Tìm kiếm từ cần tìm.
- 2) Nếu không tìm thấy thì thông báo hoàn thành công việc.
- 3) Nếu tìm thấy thì thực hiện việc thay thế cụm từ này.
- 4) Tìm tiếp bằng cách thực hiện lại lệnh 1) cho phần văn bản tiếp theo.

Thứ tự các lệnh trên không thể thay đổi.

Bài 2. Nếu thay đổi thứ tự của lệnh 1 "Tiến 2 bước" và lệnh 2 "Quay trái, tiến 1 bước" trong chương trình điều khiển rô-bốt thì sau hai lệnh trên rô-bốt sẽ "Quay trái và tiến 3 bước" và nó sẽ đi tới vị trí không có rác, dẫn đến rô-bốt sẽ không thực hiện được công việc nhặt rác. Nói chung, các lệnh trong chương trình cần được đưa ra theo một thứ tự xác định sao cho ta đạt kết quả mong muốn.

Vị trí mới của rô-bốt sau khi thực hiện xong chương trình "Hãy nhặt rác" là vị trí có thùng rác (ở góc đối diện).

Ta có nhiều cách khác nhau để đưa ra hai lệnh để rô-bốt trở lại vị trí ban đầu của mình, một trong các cách đó là ba lệnh "Quay trái, tiến 3 bước", "Quay trái, tiến 3 bước" và "Quay phải, tiến 2 bước".

Bài 3. Lí do: Điều khiển máy tính tự động thực hiện các công việc đa dạng và phức tạp mà một lệnh đơn giản không đủ để chỉ dẫn.

Bài 4. Trong ngôn ngữ máy, mọi lệnh đều được biểu diễn bằng các con số 0 và 1. Ngôn ngữ máy khó đọc và khó sử dụng.

Các ngôn ngữ lập trình được phát triển để khắc phục các nhược điểm của ngôn ngữ máy. Ngôn ngữ lập trình sử dụng các cụm từ tự nhiên nên dễ nhớ, dễ sử dụng.

Bài 5. Chương trình dịch giúp chuyển đổi chương trình được viết bằng ngôn ngữ lập trình thành chương trình bằng ngôn ngữ máy thực hiện được trên máy tính. Như vậy, chương trình dịch chuyển đổi tệp gồm các dòng lệnh được soạn thảo thành tệp có thể chạy trên máy tính.

Bài 2. LÀM QUEN VỚI CHƯƠNG TRÌNH VÀ NGÔN NGỮ LẬP TRÌNH

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Biết ngôn ngữ lập trình gồm các thành phần cơ bản là bảng chữ cái và các quy tắc để viết chương trình, câu lệnh;
- Biết ngôn ngữ lập trình có tập hợp các từ khoá dành riêng cho mục đích sử dụng nhất định;
- Biết *tên* trong ngôn ngữ lập trình là do người lập trình đặt ra, tên phải tuân thủ các quy tắc của ngôn ngữ lập trình. Tên không được trùng với các từ khoá;
- Biết cấu trúc chương trình bao gồm phần khai báo và phần thân.

2. Những điểm cần lưu ý và gợi ý dạy học

Mục tiêu của bài 2 là giới thiệu cho HS về một số thành phần cơ bản của ngôn ngữ lập trình nói chung, làm quen với cấu trúc chương trình đơn giản, làm quen với FP để chuẩn bị cho bài thực hành 1.

a) Để giới thiệu về thành phần của ngôn ngữ lập trình SGK sử dụng cách tiếp cận xuất phát từ một chương trình Pascal cụ thể, sau đó khái quát hoá và tổng kết. Cần lưu ý không giải thích ngay tất cả những gì có trong chương trình ví dụ, tránh sa đà giới thiệu chi tiết cú pháp và ngữ nghĩa của các câu lệnh Pascal mà chỉ nên tập trung khai thác những điểm cần thiết phục vụ cho mục tiêu của bài học. Cần đến thành phần, câu lệnh nào thì tập trung vào đó, phân tích, giải thích vừa đủ, biết điểm dừng để luôn hướng đến mục tiêu của mục, bài.

HS đã biết để viết chương trình cần sử dụng một ngôn ngữ lập trình cụ thể ở bài 1. Do vậy, GV có thể đặt câu hỏi tại sao lại phải viết chương trình theo một ngôn ngữ lập trình cụ thể nào đó để HS thảo luận, trả lời nhằm ôn lại bài cũ.

Cần cho HS quan sát hình 1.6 (SGK) để thấy trực quan một chương trình cụ thể. Cách làm này nhằm gây hứng thú cho HS ngay khi vào bài học. Từ việc quan sát ví dụ này GV khái quát lên thành những kiến thức chung về ngôn ngữ lập trình.

Ngôn ngữ lập trình Pascal đã được nhắc đến ở bài 1, do vậy GV có thể giới thiệu cho các em biết ví dụ trong hình 1.6 là một chương trình viết bằng ngôn ngữ lập trình Pascal.

b) Để dạy mục 1, SGK dựa trên những gì HS quan sát được như chữ cái, kí hiệu để khái quát thành phần thứ nhất: bảng chữ cái, các kí hiệu. Dựa trên ví dụ về câu lệnh *writeln('Chao cac ban')* để khái quát thành quy tắc viết.

Có thể xuất phát từ ngôn ngữ tiếng Việt, nếu HS được học ngoại ngữ thì dùng ngôn ngữ tự nhiên là chính ngoại ngữ các em đang học để lấy ví dụ sẽ thuận tiện hơn. HS biết một ngôn ngữ bao gồm các chữ cái, các từ và quy tắc ngữ pháp. Muốn người khác hiểu được và hiểu đúng thì cần dùng các chữ cái, những từ cho phép và phải được ghép theo đúng quy tắc ngữ pháp.

Ngôn ngữ lập trình cũng vậy (quan sát hình 1.6), có bảng chữ cái và các quy tắc viết. Khi viết chương trình phải sử dụng các chữ cái, các từ và tuân thủ quy tắc viết mà ngôn ngữ lập trình đặt ra. Có như vậy chương trình mới có thể được dịch sang ngôn ngữ máy mà máy tính có thể hiểu và thực hiện được. Cụ thể, để ra lệnh cho máy tính hiển thị dòng chữ "Chao cac ban" thì trong chương trình phải viết đúng là: *writeln('Chao cac ban');*

GV có thể sử dụng cách so sánh với ngôn ngữ tự nhiên để HS dễ dàng hiểu được nội dung này. Chẳng hạn, trong tiếng Việt, không phải cứ ghép các chữ cái bất kì là được một từ có nghĩa, hoặc cứ ghép các từ (có nghĩa) là có một câu có nghĩa. Như vậy, có thể xem quy tắc viết các câu lệnh trong một ngôn ngữ lập trình (cú pháp và ngữ nghĩa) là quy tắc "chính tả" và "ngữ pháp" của ngôn ngữ lập trình đó.

Trên thực tế, một chương trình có thể được viết không phải chỉ bằng một ngôn ngữ lập trình cụ thể mà có thể là hai hoặc nhiều ngôn ngữ lập trình cùng được sử dụng trong một chương trình. Ví dụ, như trong một chương trình được soạn thảo và dịch với FP có thể có một số lệnh được viết bằng ngôn ngữ Assembly (hợp ngữ). Tuy nhiên, SGK không đề cập đến vấn đề này mà ngầm định những chương trình đề cập đến chỉ sử dụng một ngôn ngữ lập trình.

c) Mục 2, GV sử dụng ví dụ ở hình 1.7, SGK (*CT_Dau_tien*) để minh họa cho HS về các thành phần của ngôn ngữ lập trình.

Các từ như *program*, *uses*, *begin*, *end* được gọi là từ khoá là các từ mà ngôn ngữ lập trình đã quy định dùng với ý nghĩa, chức năng cố định. Từ khoá là khái niệm mới với HS, vì vậy để HS hiểu về quy định từ khoá trong ngôn ngữ lập trình, có thể lấy ví dụ về cụm từ *Lớp trưởng*. Lớp trưởng là một cụm từ dành riêng để gọi một HS trong lớp đảm nhiệm chức vụ lớp trưởng của lớp, không thể có một HS nào khác trong lớp cũng được gọi là lớp trưởng (trong cùng thời điểm).

Tên là do người lập trình tự đặt ra và sử dụng những kí tự mà ngôn ngữ lập trình cho phép, tất nhiên là tên không được trùng với từ khoá.

Câu lệnh *writeln('Chao cac ban')* là một câu lệnh chỉ dẫn máy tính hiển thị dòng chữ "*Chao cac ban*" trên màn hình.

GV không cần giải thích sâu về chương trình này, cũng không nên giải thích quá kĩ về từ khoá, tên, câu lệnh ở đây. HS sẽ còn tiếp cận dần với những kiến thức này ở các bài học sau.

GV có thể giới thiệu thêm về việc thay cụm từ *Chao cac ban* thành cụm từ khác để HS có thể thực hiện ở bài thực hành sắp tới, tạo hứng thú cho HS trong tiết thực hành.

d) Mục 3, dựa vào hình 1.7 (CT_Dau_tien), chỉ cần cho HS nhận biết được chương trình gồm hai phần:

Phân khai báo: Khai báo tên và một số khai báo khác (các em sẽ học sau).

Phần thân: Bắt đầu bằng từ khoá *Begin* và kết thúc bằng từ khoá *End* và dấu chấm (*End.*). Giữa từ khoá *Begin* và *End* là các câu lệnh.

Lưu ý: Phần thân là phần quan trọng, bắt buộc phải có ở mọi chương trình, còn phần khai báo có thể có hoặc không.

Đến đây có thể cho HS phát hiện về từ khoá và chức năng của từ khoá qua chương trình trong hình 1.7 SGK: *Program* là từ khoá dùng để khai báo tên chương trình; Từ khoá *Begin* dùng để khai báo bắt đầu chương trình, từ khoá *End* dùng để khai báo kết thúc chương trình.

Cách tiếp cận từ cụ thể đến khái quát có ưu điểm là phù hợp với lứa tuổi HS THCS. Tuy nhiên, hạn chế của cách tiếp cận này là HS dễ bị dẫn đến nhận thức rằng đang học ngôn ngữ Pascal. Như đã biết, mục tiêu là dạy học lập trình, ngôn ngữ Pascal là một minh hoạ cụ thể. Do vậy, GV cần lưu ý trước khi chuyển sang mục 4, cần nhấn mạnh để HS ghi nhớ, với tất cả ngôn ngữ lập trình đều có tập hợp các kí hiệu (bảng chữ cái) và quy tắc riêng để viết chương trình.

Có thể giới thiệu nội dung mục 4 trên lớp hoặc yêu cầu các em đọc nội dung này ở nhà chuẩn bị cho bài thực hành 1. Mục tiêu là để HS nhận biết giao diện của phần mềm FP và biết các bước soạn thảo, dịch chương trình phục vụ trực tiếp cho bài thực hành 1. Nếu giới thiệu trên lớp, GV nên chuẩn bị sẵn một số hình ảnh về giao diện của FP trên giấy khổ rộng và tận dụng kênh hình trong SGK. Trong cả hai trường hợp, khi tiến hành bài thực hành 1 GV cần tổng kết lại nội dung này ngay trước khi HS bắt đầu sử dụng máy tính để thực hành.

Cần nhấn mạnh cho HS về việc tạo chương trình chạy được trên máy tính gồm hai bước: soạn thảo chương trình trên máy tính theo một ngôn ngữ lập trình cụ thể và dịch chương trình vừa soạn thảo sang ngôn ngữ máy.

Việc soạn thảo chương trình về cơ bản giống với soạn thảo văn bản mà các em đã học. Việc dịch chương trình cũng rất đơn giản, ví dụ với ngôn ngữ lập trình FP, sau khi soạn thảo xong chỉ cần nhấn tổ hợp phím **Alt+F9** là máy tính

tự động dịch chương trình. Để dịch và chạy chương trình, nhấn tổ hợp phím **Ctrl+F9**.

Trong phần này cần cho HS biết có nhiều ngôn ngữ lập trình, trong chương trình học các em sẽ làm việc với một ngôn ngữ lập trình, đó là Pascal.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. Các thành phần cơ bản ngôn ngữ lập trình gồm *bảng chữ cái* và *các quy tắc* để viết các câu lệnh (cú pháp) có ý nghĩa xác định, cách bố trí các câu lệnh,... sao cho có thể tạo thành một chương trình hoàn chỉnh và chạy được trên máy tính.

Bài 2. Tên trong chương trình là dãy các kí tự hợp lệ được lấy từ bảng chữ cái của ngôn ngữ lập trình.

Từ khoá của một ngôn ngữ lập trình (còn được gọi là *từ dành riêng*) là tên được dùng cho các mục đích nhất định do ngôn ngữ lập trình quy định, không được dùng cho bất kì mục đích nào khác.

Người lập trình có thể đặt tên một cách tuỳ ý nhưng phải tuân thủ các quy tắc của ngôn ngữ lập trình cũng như của chương trình dịch, trong đó:

- (1) Hai tên khác nhau ứng với những đại lượng khác nhau;
- (2) Tên không được trùng với các từ khoá.

Bài 3. Các tên hợp lệ: *a*, *Tamgiac*, *beginprogram*, *b1*, *abc*. Tên không hợp lệ: *8a* (bắt đầu bằng số), *Tam giac* (có dấu cách), *end* (trùng với từ khoá).

Bài 4. Xem SGK, mục 3, bài 2.

Bài 5. Chương trình 1 là chương trình Pascal đầy đủ và hoàn toàn hợp lệ, mặc dù chương trình này không thực hiện điều gì cả. Phần nhất thiết phải có trong chương trình là phần thân được xác định bởi hai từ khoá *begin* và *end* (có dấu chấm).

Chương trình 2 là chương trình Pascal không hợp lệ vì câu lệnh khai báo tên chương trình *program CT_thu* nằm ở phần thân.



Bài thực hành 1

LÀM QUEN VỚI FREE PASCAL

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Thực hiện được thao tác khởi động/thoát khỏi FP, làm quen với màn hình soạn thảo FP;
- Thực hiện được các thao tác mở bảng chọn và chọn lệnh;
- Soạn thảo được một chương trình Pascal đơn giản;
- Biết cách dịch, sửa lỗi trong chương trình, chạy chương trình và xem kết quả;
- Biết sự cần thiết phải tuân thủ quy định của ngôn ngữ lập trình.

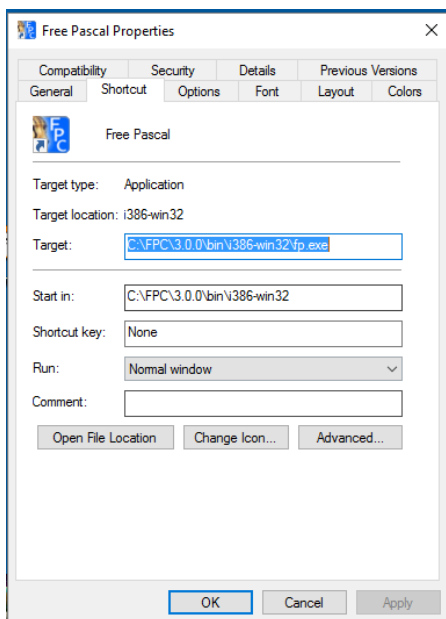
2. Những điểm cần lưu ý và gợi ý dạy học

a) Trong lần tái bản này, chúng tôi sẽ sử dụng FP làm môi trường lập trình chính trong việc dạy và học ngôn ngữ Pascal. FP là phần mềm miễn phí, có thể chạy trên tất cả các phiên bản hệ điều hành Windows 32 hoặc 64 bit nên rất thích hợp cho các nhà trường. Sau khi cài đặt (ví dụ phiên bản Free Pascal 3.0, ton bộ các tệp chính của FP được sao chép trong thư mục C:\FPC\3à.0.0\bin\i386-win32. Trên thực tế các GV có thể lựa chọn môi trường Free Pascal hoặc Turbo Pascal, nhưng chúng tôi kiến nghị nên sử dụng FP.

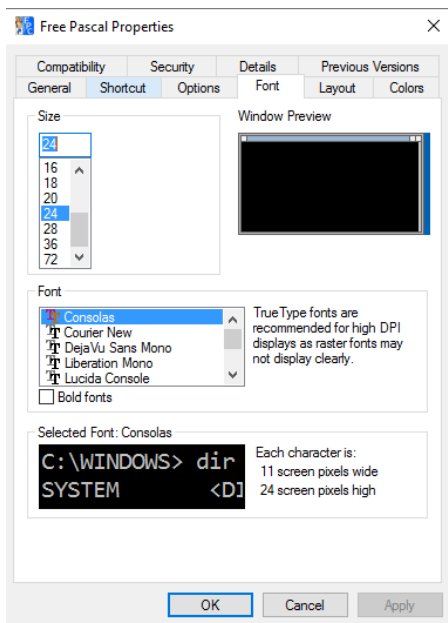
b) Một số điểm chú ý khi cài đặt FP

Muốn thay đổi phong chữ và kích thước của cửa sổ cho giao diện của FP cần thực hiện các thao tác sau:

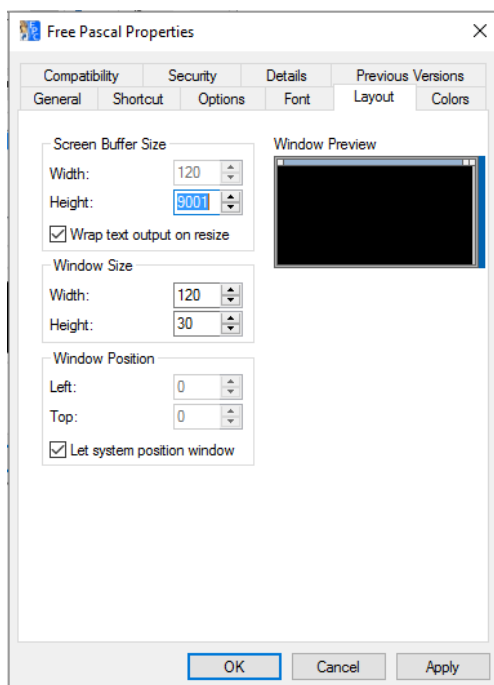
- Nháy nút phải chuột trên biểu tượng của tệp FP.exe, chọn lệnh Properties sẽ làm xuất hiện cửa sổ như hình a.
- Chọn trang Font để cài đặt phong chữ và cỡ chữ cho cửa sổ soạn thảo chương trình FP như hình b.
- Vào trang **Layout** để chọn kích thước của sổ soạn thảo (tính theo số kí tự chiều ngang và chiều dọc) và vị trí của cửa sổ chương trình FP như hình c.



Hình a



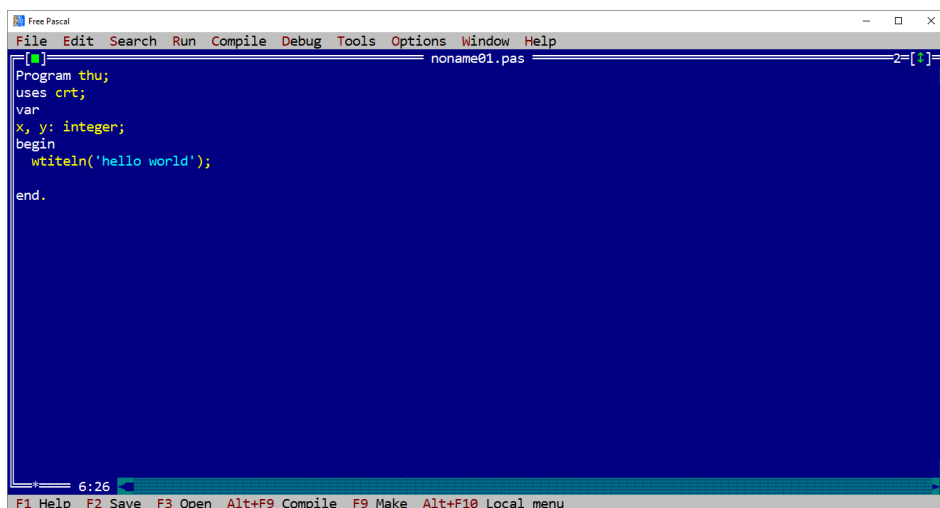
Hình b



Hình c

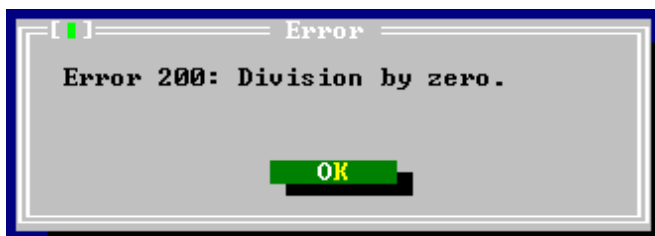
– Nháy **OK** để kết thúc, đóng cửa sổ cài đặt thuộc tính của FP.

Giao diện FP sẽ tương tự như hình sau:



c) Riêng về Turbo Pascal. Mục này dành riêng cho GV sử dụng TP để hướng dẫn học sinh.

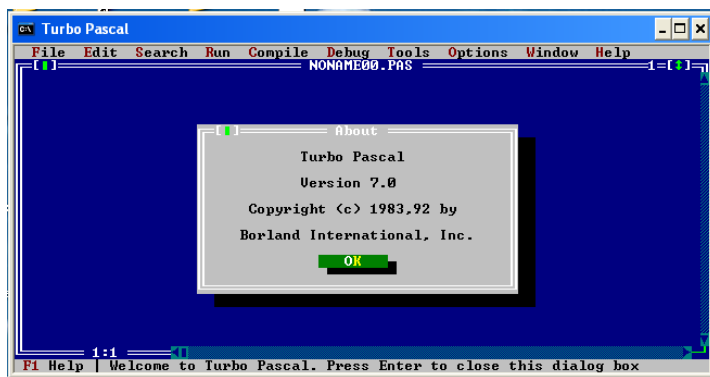
Để chạy được chương trình TP cần có tối thiểu hai tệp: **TURBO.exe** và **TURBO.TPL**. Lưu ý rằng nếu sử dụng Turbo Pascal for DOS, trong chương trình có sử dụng thư viện `crt` (khai báo `uses crt`) thì khi dịch chương trình có thể sẽ gặp thông báo lỗi *Error 200: Division by zero* như hình sau:



Lỗi này không phải do chương trình được viết có lỗi mà do phần mềm TP đang sử dụng không phù hợp với máy tính có tốc độ cao.

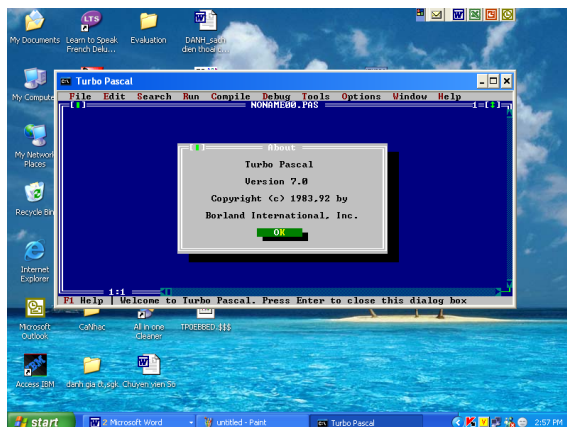
Vì HS đã được học, thực hành về khởi động chương trình ở các năm học trước nên việc khởi động TP là dễ dàng với các em. Mặc dù vậy, GV vẫn nên tạo biểu tượng tắt (shortcut) của chương trình TP trên màn hình nền để thuận tiện cho HS khởi động trong tiết thực hành.

Khi khởi động TP, màn hình hiện lên như hình dưới đây:



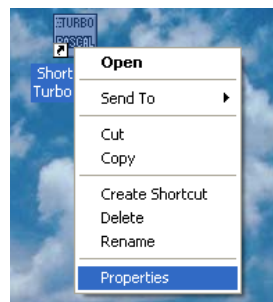
Rất có thể HS bối rối, lúng túng vì có một thông báo giữa màn hình vì điều này không được nhắc đến trong SGK. GV lưu ý nhắc HS nhấn nút **OK** để bắt đầu làm việc với TP.

Một lưu ý nữa, có thể màn hình làm việc của TP chỉ là một cửa sổ nhỏ, không chiếm hết toàn bộ màn hình như minh họa dưới đây:

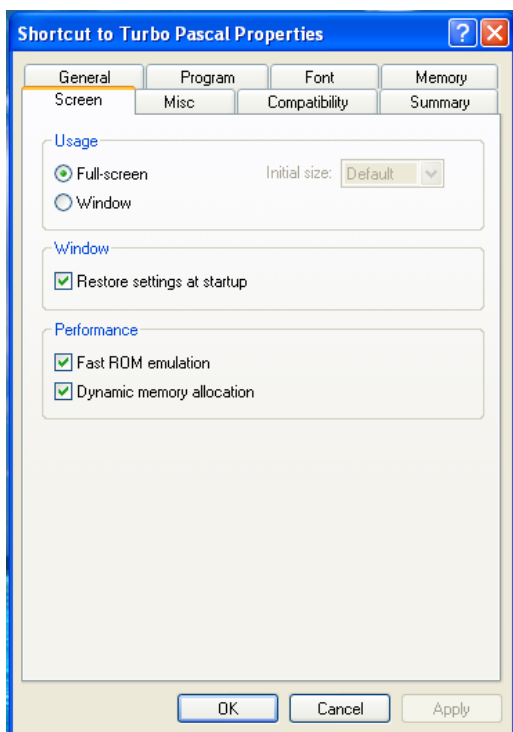


Để HS tiện theo dõi thì nên mở rộng cửa sổ TP chiếm hết toàn bộ màn hình. Cách làm như sau:

- Nháy nút phải chuột lên biểu tượng của TP trên màn hình nền để mở bảng chọn tắt như hình bên.
- Trong bảng chọn tắt, chọn mục **Properties**, cửa sổ *Shortcut to Turbo Pascal Properties* hiện lên.



- Trong cửa sổ *Shortcut to Turbo Pascal Properties*, chọn trang **Screen**, sau đó nhấp chuột chọn **Full-screen** như hình sau:



- Nhấp **OK** để hoàn tất.

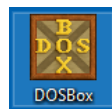
Từ đó, mỗi khi khởi động TP, màn hình làm việc của TP sẽ mở rộng toàn bộ màn hình máy tính.

Cài đặt, chạy Turbo Pascal nhờ DOSBox

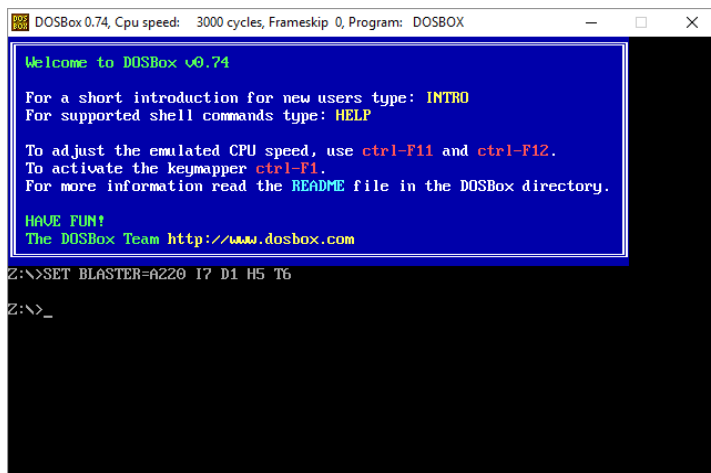
DOSBox là gì và khi nào cần sử dụng DOSBox?

TP chỉ được chạy tốt trên nền các hệ điều hành cũ (8 bit, ví dụ DOS), do đó trên các phiên bản 32 bit sau này của Windows thường có lỗi. Để tránh các lỗi này có thể dùng phần mềm DOSBox, một công cụ có khả năng mô phỏng môi trường DOS trên nền các hệ điều hành Windows 32 bit. Cách thiết lập DOSBox và chạy TP trên nền DOSBox như sau:

Bước 1. Cài đặt phần mềm DOSBox (trên nền Windows hiện thời). Biểu tượng của phần mềm như hình bên.



Bước 2. Khi chạy DOSBox có dạng như hình sau:



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Welcome to DOSBox v0.74

For a short introduction for new users type: INTRO
For supported shell commands type: HELP

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

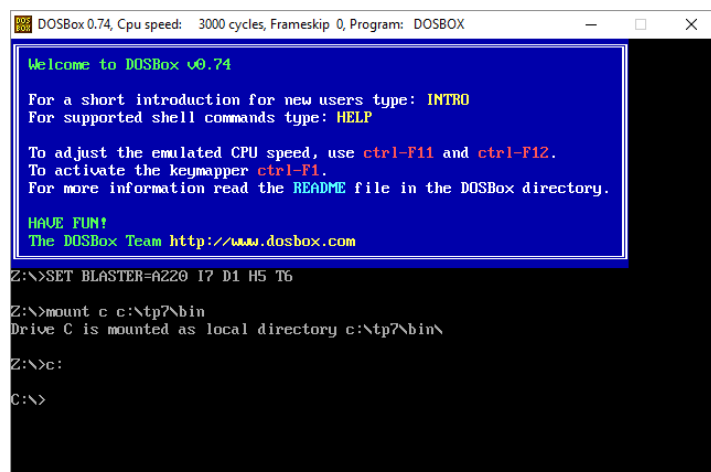
HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>_
```

Chúng ta thấy ngay cửa sổ làm việc của DOSBox có dạng dòng lệnh tương tự DOS. "Ổ đĩa" ngầm định là Z.

Bước 3. Tiếp theo là cần thiết lập một "ổ đĩa ảo" để chạy được TP. Giả sử đã có thư mục TP7 ở ổ đĩa C, gõ lệnh: **mount C C:\TP7\bin**



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Welcome to DOSBox v0.74

For a short introduction for new users type: INTRO
For supported shell commands type: HELP

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com

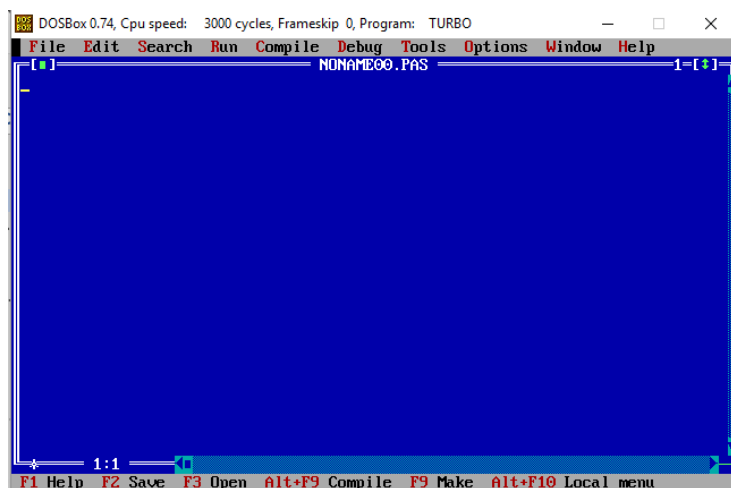
Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount c c:\tp7\bin
Drive C is mounted as local directory c:\tp7\bin\

Z:\>c:
C:\>
```

Sau lệnh trên ổ đĩa C sẽ chính là thư mục chứa các tệp chạy của TP.

Bước 4. Tiếp theo chuyển ổ đĩa làm việc sang C: và chạy TP một cách dễ dàng. Giao diện chạy của TP trên DOSBox như sau:



Bước 5. Muốn thoát khỏi DOSBox trước tiên thoát khỏi TP, sau đó gõ lệnh **Exit**.

d) Trong bài 1, cần cho HS nhận biết biểu tượng của FP trên màn hình nền, khởi động/thoát khỏi FP; biết cách mở bảng chọn; nhận biết được dòng trợ giúp nằm dưới cùng của màn hình để biết tổ hợp phím tắt khi thực hiện các lệnh. Không nên mất nhiều thời gian cho bài 1 bởi vì những kỹ năng này HS sẽ còn phải làm quen, sử dụng ở những bài sau.

e) Với bài 2, cần nhắc HS gõ chính xác chương trình vào máy tính.

Mặc dù việc soạn thảo một chương trình ngắn như ví dụ đưa ra chưa cần sử dụng nhiều đến các công cụ soạn thảo nhưng GV vẫn cần lưu ý cho HS: Soạn thảo trong FP có một số điểm khác với soạn thảo văn bản mà các em đã được học, cần hướng dẫn HS sử dụng phím **Delete**, **Backspace** khi soạn thảo trong FP. Các công cụ soạn thảo như: sao chép, di chuyển,... trong FP cũng khác, cần hướng dẫn HS cách tra cứu các lệnh này trong bảng chọn khi cần thiết. Có thể HS muốn gõ tiếng Việt có dấu ở những câu tiếng Việt (do đã quen với gõ tiếng Việt có dấu khi làm việc với phần mềm bảng tính, phần mềm soạn thảo văn bản ở các lớp trước), cần lưu ý các em chỉ gõ tiếng Việt không dấu, FP không hỗ trợ gõ tiếng Việt có dấu.

Trọng tâm của bài 2 này là HS thực hiện được việc soạn thảo, lưu, dịch và chạy được chương trình.

Khi dịch chương trình rất có thể máy tính sẽ báo lỗi do HS soạn thảo chương trình còn lỗi chính tả, không hoàn toàn chính xác. GV yêu cầu HS tự đối chiếu chương trình vừa gõ với chương trình trong SGK để chỉnh sửa theo

đúng chương trình mẫu. Việc làm này là cần thiết để HS thấy được sự nghiêm ngặt của ngôn ngữ lập trình và rèn luyện thái độ nghiêm túc trong học tập, làm việc với ngôn ngữ lập trình.

Khi nhấn **Ctrl+F9** để dịch và chạy chương trình, có thể HS không xem được kết quả hiển thị trên màn hình. Để dừng màn hình lại cho HS quan sát kết quả cần nhấn **Alt+F5** hoặc thêm lệnh *Readln* ngay trước từ khoá *End* kết thúc chương trình. Khi đó, màn hình sẽ dừng lại để HS quan sát kết quả, quan sát kết quả xong nhấn phím **Enter** để trở về màn hình soạn thảo của FP.

GV có thể hướng dẫn các em thay các cụm từ *Chao cac ban* và *Minh la Free Pascal* bằng các cụm từ khác để tạo hứng thú trong học tập.

f) Bài 3 nhằm mục đích để HS làm quen với việc sử dụng FP sửa lỗi cú pháp trong chương trình. Có thể căn cứ vào thông báo lỗi của FP để sửa chương trình.

Cùng với việc cung cấp chương trình soạn thảo, việc dịch, phát hiện và thông báo lỗi là các yếu tố quan trọng của một môi trường lập trình. Một môi trường lập trình tốt là một môi trường có nhiều công cụ hỗ trợ cho người lập trình trong việc soạn thảo, dịch, phát hiện và sửa lỗi. Hiện nay, có nhiều môi trường lập trình cung cấp các tiện ích hỗ trợ tốt cho người lập trình như Java, Visual C, Visual Basic,...

Nếu còn thời gian, GV có thể yêu cầu HS thay đổi giữa cách viết thường và cách viết hoa của từ khoá để thấy được Pascal không phân biệt chữ hoa và chữ thường. Cho HS thay lệnh *write* bằng *writeln* (hoặc ngược lại) và quan sát để nhận thấy sự khác biệt giữa hai lệnh này. Ví dụ, ban đầu trong chương trình có hai dòng lệnh *writeln('Chao cac ban');* và *write('Minh la Free Pascal');* thì kết quả đưa ra màn hình trên hai dòng. Sau đó sửa lệnh đầu tiên thành *write('Chao cac ban')* và giữ nguyên lệnh thứ hai thì kết quả in ra trên một dòng. So sánh hai kết quả để rút ra sự khác nhau giữa lệnh *write* là *writeln*. Cách làm này là một phương pháp hướng dẫn HS tự khám phá, tìm hiểu câu lệnh của ngôn ngữ lập trình.

g) Việc thay đổi nội dung hiển thị trên màn hình ở bài 4 sẽ giúp HS thấy được câu lệnh *write* (hoặc *writeln*) hiển thị nội dung đặt trong dấu nháy đơn của câu lệnh. Thêm nữa, khi mới học lập trình HS sẽ hứng thú khi thấy máy tính hiển thị tên của mình lên màn hình.

Bài 3. CHƯƠNG TRÌNH MÁY TÍNH VÀ DỮ LIỆU

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Biết khái niệm kiểu dữ liệu;
- Biết một số phép toán cơ bản với dữ liệu số;
- Biết khái niệm điều khiển tương tác giữa người với máy tính.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Phân khởi động sẽ giúp HS biết và nhận ra rằng không phải có thể thực hiện các phép toán bất kì trên mọi dữ liệu. Vấn đề được nảy sinh của bài học chính là khái niệm **Kiểu dữ liệu**.

b) HS đã được làm quen với khái niệm dữ liệu ở các lớp trước, không cần thiết phải giải thích sâu thêm về khái niệm dữ liệu ở đây.

Kiểu dữ liệu là một khái niệm tương đối khó với HS. Vì vậy, không yêu cầu truyền đạt hết kiến thức về kiểu dữ liệu ở bài này. HS còn được tiếp cận dần về kiểu dữ liệu ở các bài sau.

SGK chỉ hạn chế giới thiệu các kiểu dữ liệu đơn giản và thường được sử dụng nhất: dữ liệu kiểu số nguyên, kiểu số thực, kiểu xâu kí tự. Thậm chí, kiểu số nguyên trong ngôn ngữ minh hoạ là Pascal, SGK cũng chỉ bước đầu giới thiệu kiểu *integer*. Trong các bài thực hành tiếp theo, HS sẽ dần dần được giới thiệu thêm một vài kiểu dữ liệu khác. Một khi HS đã làm quen và hiểu các khái niệm cơ bản về một vài kiểu dữ liệu, việc giới thiệu các kiểu dữ liệu khác sẽ rất nhẹ nhàng. HS sẽ dần biết về tầm quan trọng của các kiểu dữ liệu khi học đến bài tiếp theo, sử dụng biến trong chương trình.

Với mỗi kiểu dữ liệu thì có các phép toán xử lý tương ứng, ví dụ với dữ liệu là số thì có thể tiến hành các phép toán cộng, trừ, nhân, chia với các số đó. Phép toán như *div*, *mod* lại chỉ có thể thực hiện với kiểu nguyên mà không thực hiện được với kiểu thực.

Về thao tác xử lý dữ liệu kiểu xâu, GV có thể cho HS thấy ví dụ về thực hiện thao tác hiển thị dữ liệu kiểu xâu ra màn hình mà các em đã học ở bài thực hành 1.

```
writeln('Chao Cac Ban');  
  
write('Minh la Free Pascal');
```

Lưu ý rằng dữ liệu kiểu xâu trong Pascal được đặt trong cặp dấu nháy đơn. GV chưa nên giới thiệu về các thao tác xử lý đối với dữ liệu kiểu xâu kí tự gây quá tải với HS.

c) Mục 2 chỉ nêu các phép toán với dữ liệu kiểu nguyên và kiểu thực. Cần lưu ý một số điểm sau:

- Sự khác nhau giữa kí hiệu phép toán trong toán học và trong Pascal. Có thể cho HS tự xem các bảng 1.2 (SGK) để phát hiện ra sự khác nhau này.
- Trong Pascal chỉ cho phép sử dụng cặp dấu ngoặc đơn () để mô tả thứ tự thực hiện các phép toán. Không dùng cặp dấu ngoặc vuông [] hay cặp dấu ngoặc nhọn {} như trong toán học. GV có thể hỏi HS, giả sử khi viết chương trình một bạn nào đó đã quên quy định này của Pascal mà dùng dấu ngoặc vuông hay dấu ngoặc nhọn để viết biểu thức thì có điều gì xảy ra? Mục tiêu của câu hỏi này là để các em nhớ rằng luôn phải tuân thủ những nguyên tắc, quy định mà ngôn ngữ lập trình đặt ra, nếu không chương trình dịch sẽ không hiểu và không thể dịch ra cho máy tính thực hiện được.
- Các phép toán được thực hiện theo thứ tự ưu tiên:
 1. Các phép toán trong ngoặc được thực hiện trước tiên;
 2. Trong dãy các phép toán không có dấu ngoặc, các phép nhân, phép chia, phép chia lấy phần nguyên (div) và phép chia lấy phần dư (mod) được thực hiện trước;
 3. Phép cộng và phép trừ theo thứ tự từ trái sang phải.

Các phép toán lấy phần nguyên (div), lấy phần dư (mod) chỉ giới thiệu cho HS biết, không nên dành nhiều thời gian vào giới thiệu hai phép toán này.

Có một quy tắc quan trọng mà HS thường bỏ qua và làm ảnh hưởng tới kết quả tính toán: Trong một biểu thức chỉ có phép cộng và phép trừ, hoặc chỉ có phép nhân hoặc phép chia, các phép tính được thực hiện theo thứ tự *từ trái sang phải*. Cần đặc biệt lưu ý đến điều này khi chuyển đổi các biểu thức toán học sang dạng biểu thức trong Pascal. GV có thể nêu nhiều ví dụ khác nhau

để nhắc nhở HS lưu ý và sử dụng các cặp dấu ngoặc đơn để nhóm các phép tính, ví dụ:

$10 - 5 + 2 = 7$, nhưng nếu thực hiện phép cộng trước ta được kết quả 3.

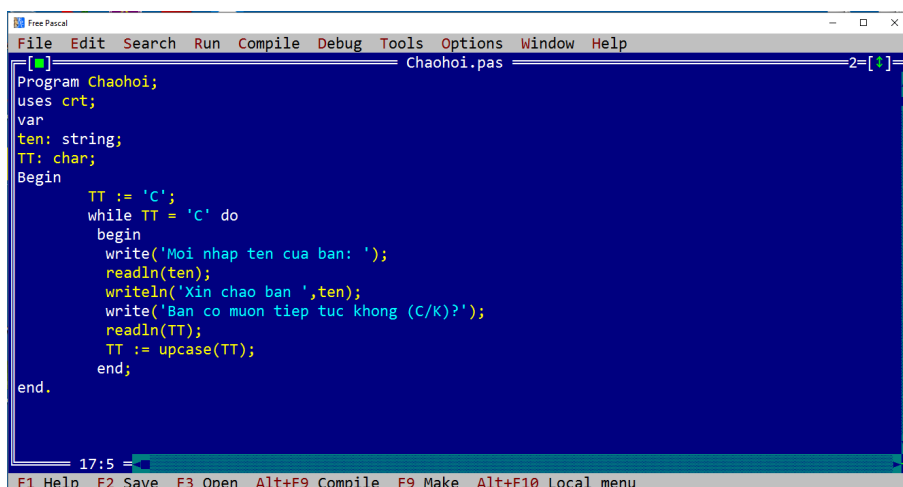
$6 \times 6 / 2 \times 2 = 36$, nhưng nếu thực hiện các phép nhân trước ta được kết quả 9.

Chẳng hạn, HS thường chuyển nhầm biểu thức $\frac{a^2}{(2b + c)^2}$ sang dạng biểu thức Pascal như sau: $a*a / (2*b+c) * (2*b+c)$.

d) Mục 3 - các phép so sánh, cũng giống với mục 2, cần cho HS nhận thấy sự khác biệt về kí hiệu sử dụng trong toán học và trong Pascal. Điểm cần nhấn mạnh ở mục này là kết quả của một phép so sánh chỉ có thể là đúng hoặc sai. HS sẽ hiểu rõ hơn về ý nghĩa của phép so sánh khi học đến câu lệnh điều kiện, cấu trúc điều khiển ở bài sau.

Cần lưu ý rằng các kí hiệu phép toán, phép so sánh ở trên là của Pascal. Có sự khác nhau về các kí hiệu này ở các ngôn ngữ lập trình khác nhau. Khi làm việc với ngôn ngữ lập trình nào thì phải tuân thủ các quy định về kí hiệu phép toán của ngôn ngữ lập trình đó. Tuy nhiên, các ngôn ngữ lập trình đều cho phép biểu diễn các phép tính số học, phép so sánh.

e) Mục 4 (Giao tiếp người-máy tính) tốt nhất nên được GV minh hoạ trên máy tính. Có thể viết sẵn và cho chạy một chương trình như sau (GV có thể sử dụng chương trình khác):



```
Free Pascal
File Edit Search Run Compile Debug Tools Options Window Help
ChaoHoi.pas
Program ChaoHoi;
uses crt;
var
  ten: string;
  TT: char;
Begin
  TT := 'C';
  while TT = 'C' do
  begin
    write('Moi nhap ten cua ban: ');
    readln(ten);
    writeln('Xin chao ban ',ten);
    write('Ban co muon tiep tuc khong (C/K)?');
    readln(TT);
    TT := upcase(TT);
  end;
end.
17:5 =
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu
```

Chương trình này sẽ cho phép nhập tên của người dùng và tiến hành in ra màn hình dòng chữ có tên mà người dùng vừa nhập. Chương trình sẽ lặp đến khi người dùng nhấn phím khác với phím **C**. Có thể mời lần lượt một số HS nhập tên của chính các em để thấy được sự thay đổi tương ứng với dữ liệu nhập vào. Từ đó các em thấy được khái niệm tương tác người-máy tính.

GV cần lưu ý cho HS thấy sự tương tác giữa người và máy có được là do người lập trình tạo ra. Có thể mở chương trình và giải thích sơ bộ cho các em về một số câu lệnh đơn giản để nhập tên, in dòng chào với tên tương ứng. Lưu ý, lúc này không phải là thời điểm thích hợp để giải thích tất cả các câu lệnh trong chương trình. Những tương tác người-máy tính mà các em đã thực hiện khi soạn thảo văn bản, sử dụng hệ điều hành,... là do người lập trình tạo ra đó là kiến thức quan trọng mà các em cần rút ra ở đây. Điều này thể hiện sự khác biệt giữa học tin học đơn thuần chỉ để sử dụng và học tin học với tư cách là một ngành khoa học. HS sẽ dần hiểu rõ hơn về việc này ở những bài học sau.

Để chuẩn bị cho bài thực hành 2, GV có thể cho HS làm câu a, bài 1 của bài thực hành 2 ngay trên lớp.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. Có thể nêu các ví dụ sau đây:

- a) Dữ liệu kiểu số và dữ liệu kiểu xâu kí tự. Phép nhân được định nghĩa trên dữ liệu số, nhưng không có nghĩa trên dữ liệu kiểu xâu.
- b) Dữ liệu kiểu số nguyên và dữ liệu kiểu số thực. Phép chia lấy phần nguyên và phép chia lấy phần dư có nghĩa trên dữ liệu kiểu số nguyên, nhưng không có nghĩa trên dữ liệu kiểu số thực.

Bài 2. Biểu diễn số 2017 có thể dùng kiểu dữ liệu số nguyên, số thực hoặc kiểu xâu kí tự. Tuy nhiên, để chương trình dịch FP hiểu 2017 là dữ liệu kiểu xâu, chúng ta phải viết dãy số này trong cặp dấu nháy đơn (').

```
var a: string; b: integer;  
begin  
  writeln('2017');  
  writeln(2017);
```

```

a:='2017';
b:=2017;
end.

```

Bài 3. Với hai chuỗi ký tự có thể định nghĩa được phép tính ghép (hoặc phép cộng) 2 chuỗi. Phép tính này sẽ nối hai chuỗi vào thành một chuỗi. Ví dụ:

"Lớp" + " " + "8A" = "Lớp 8A".

Bài 4. Lệnh *Writeln*('5+20=', '20+5') in ra màn hình hai chuỗi ký tự '5+20' và '20+5' liền nhau: 5+20=20+5, còn lệnh *Writeln*('5+20=', 20+5) in ra màn hình chuỗi ký tự '5+20' và tổng của 20+5 như sau: 5+20=25.

Bài 5. Các biểu thức trong Pascal:

a) $a/b+c/d$;

b) $a*x*x+b*x+c$;

c) $1/x-a/5*(b+2)$;

d) $(a*a+b)*(1+c)*(1+c)*(1+c)$.

Bài 6. Các biểu thức toán tương ứng:

a) $(a+b)^2 - \frac{x}{y}$;

b) $\frac{b}{a^2+c}$;

c) $\frac{a^2}{(2b+c)^2}$;

d) $1 + \frac{1}{2} + \frac{1}{2.3} + \frac{1}{3.4} + \frac{1}{4.5}$.

Bài 7. Kết quả của các phép so sánh:

a) Đúng;

b) Sai;

c) Đúng;

d) Đúng khi $x > 2.5$; ngược lại, phép so sánh có kết quả sai.

Bài 8. a) $15-8>=3$;

b) $(20-15)*(20-15)<25$;

c) $11*11=121$;

d) $x>10-3*x$.



Bài thực hành 2

VIẾT CHƯƠNG TRÌNH ĐỂ TÍNH TOÁN

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Chuyển được biểu thức toán học sang biểu diễn trong Pascal;
- Biết được kiểu dữ liệu khác nhau thì được xử lý khác nhau;
- Hiểu phép toán div, mod;
- Hiểu thêm về các lệnh in dữ liệu ra màn hình và tạm ngừng chương trình.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Câu a của bài 1 nhằm mục đích để HS tập chuyển biểu thức viết ở dạng toán học sang viết trong Pascal. Chỉ cần tập trung vào bốn phép tính đơn giản là cộng, trừ, nhân, chia. Mặc dù trong Pascal có hàm *sqr*, nhưng trong bài này để biểu diễn bình phương của một số chỉ dùng phép nhân số đó với chính nó. Ví dụ, $3^2 = 3*3$.

Lưu ý, khác với trong toán học, trong Pascal sử dụng kí hiệu * và / tương ứng với phép nhân và phép chia. Để mô tả thứ tự ưu tiên của phép toán trong Pascal chỉ dùng cặp dấu ngoặc đơn (), không sử dụng cặp dấu ngoặc vuông [] và cặp dấu ngoặc nhọn {}.

b) Đây là bài thực hành đầu tiên HS tập viết biểu thức trong Pascal. Do vậy, cần đưa ra các ví dụ đơn giản với các phép tính đơn giản, dễ dàng tính ra kết quả. Tránh đưa ra các ví dụ quá phức tạp, số lượng phép tính nhiều, khó tính toán ra kết quả. Làm như vậy để HS chỉ tập trung vào mục tiêu chính của phần này là chuyển biểu thức toán học sang mô tả trong Pascal mà không mất thời gian vào các tính toán phức tạp.

Nội dung của câu a, bài 1 là để HS làm trên lớp, không cần thiết phải sử dụng đến máy tính. Do vậy, nội dung này có thể được dạy trên lớp ngay sau bài 3.

Trong các câu b và c, HS luyện tập soạn thảo, chỉnh sửa, biên dịch, chạy và xem kết quả của chương trình. Cần lưu ý HS gõ chuẩn xác, dựa vào thông báo lỗi của FP khi biên dịch, đối chiếu với nội dung in trong SGK để chỉnh sửa chương trình nếu có (do HS gõ nhầm).

Các biểu thức ở câu b chính là biểu diễn của các biểu thức toán học ở câu a.

c) Khi quan sát kết quả trên màn hình cần cho HS thấy được với mỗi lệnh *write*, FP hiển thị ra màn hình những xâu kí tự nằm trong cặp dấu nháy đơn và hiển thị kết quả của biểu thức được đặt ngay sau dấu phẩy.

Cần giải thích để HS thấy được hai dãy giống nhau gồm số và kí hiệu phép toán, nếu đặt trong cặp dấu nháy đơn thì Pascal hiểu đó là xâu kí tự và lệnh *write* sẽ hiển thị xâu kí tự ra màn hình. Nhưng nếu không đặt trong cặp dấu nháy đơn thì Pascal coi đó là một biểu thức và sẽ tính toán biểu thức và lệnh *write* sẽ hiển thị kết quả của biểu thức. Đây cũng chính là một ví dụ minh họa cho việc kiểu dữ liệu khác nhau thì cách xử lí dữ liệu khác nhau. Sự kết hợp giữa hiển thị dữ liệu xâu và kết quả biểu thức ở đây tạo thuận lợi cho người dùng theo dõi kết quả tính toán.

Để HS có thể dễ dàng kiểm chứng kết quả tính toán biểu thức, tạo niềm tin, hứng thú trong học tập, GV có thể dành thời gian để HS tự tính toán và đối chiếu với kết quả trên màn hình FP. Nếu cần thiết, GV có thể thay các ví dụ trong SGK bằng các ví dụ khác, đơn giản hơn, để kiểm chứng kết quả hơn đối với HS, tránh mất thời gian tính toán không cần thiết.

d) Qua bài này HS còn nhận ra rằng chương trình Pascal có thể không có phần khai báo. Nói cách khác là phần khai báo không bắt buộc phải có, ngược lại phần thân chương trình thì bắt buộc phải có.

Yêu cầu HS lưu lại bài 1 để còn sử dụng khi tiến hành bài 3.

e) Qua bài 2 và bài 3 về cơ bản HS cần hiểu được lệnh *div*, *mod* và tiếp tục rèn luyện một số thao tác như soạn thảo, dịch, hiệu chỉnh, chạy và quan sát kết quả của chương trình. Cần lưu ý một số điểm sau:

- HS làm quen với phép tính *div*, *mod*, thấy được sự khác nhau giữa phép *div*, *mod* và phép chia.
- Biết lệnh *clrscr* được dùng để làm sạch màn hình hiển thị kết quả. Lệnh này có trong thư viện *crt* nên muốn sử dụng lệnh này thì phải khai báo sử dụng thư viện này ở đầu chương trình. GV gợi ý HS bỏ lệnh *uses crt* để kiểm chứng điều này (nếu như chưa thực hiện thao tác này ở bài thực hành 1).
- Các lệnh *delay*, *readln* được dùng để tạm ngừng chương trình. Các lệnh này thường được dùng ở các vị trí thích hợp trong chương trình để người dùng quan

sát kết quả, theo dõi chương trình. Việc sử dụng các lệnh này là một ví dụ về việc điều khiển giao tiếp người - máy tính.

Khi làm bài 3, HS phải mở chương trình đã được lưu ở bài 1. Mặc dù việc mở tệp có sẵn HS đã được thực hành nhiều ở các lớp dưới nhưng thao tác mở tệp của FP khác đôi chút, có thể HS sẽ lúng túng. Do vậy, GV cần lưu ý hướng dẫn HS các thao tác mở tệp khi bắt đầu bài 3.

Việc điều khiển in số thực ra màn hình của FP chỉ cần giới thiệu qua, đây không phải là kiến thức trọng tâm của bài thực hành.

Bài 4. SỬ DỤNG BIẾN VÀ HẲNG TRONG CHƯƠNG TRÌNH

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Biết khái niệm biến, hằng;
- Hiểu cách khai báo, sử dụng biến, hằng;
- Biết vai trò của biến trong lập trình;
- Hiểu lệnh gán.

2. Những điểm cần lưu ý và gợi ý dạy học

Đây là bài tương đối khó đối với HS. GV cần lưu ý nhấn mạnh một số điểm sau:

a) Biến là đại lượng để lưu trữ dữ liệu, trong chương trình có thể thay đổi giá trị của biến. Muốn sử dụng biến thì phải khai báo, khi khai báo biến phải khai báo kiểu dữ liệu mà biến sẽ lưu trữ. Biến chỉ có thể lưu trữ được dữ liệu có kiểu thuộc kiểu của biến. Người lập trình tự đặt tên cho biến theo quy tắc của ngôn ngữ lập trình đang sử dụng. Có thể gán giá trị cho biến và tính toán với các giá trị của biến.

b) Hằng được khai báo là đại lượng để lưu trữ dữ liệu cố định. Giá trị của hằng không thay đổi trong khi thực hiện chương trình.

c) Biến và **hằng** được sử dụng rất nhiều trong các bài toán lập trình, đóng vai trò như các công cụ hỗ trợ việc triển khai thuật toán để giải quyết bài toán. Do

vậy biết cách sử dụng Biến và Hằng một cách hợp lý là kỹ năng rất quan trọng của người lập trình. Điều này GV có thể nhấn mạnh cho HS biết.

d) Biến nhớ và hằng thực chất là một vùng trong bộ nhớ được chương trình đặt tên và dành riêng cho người lập trình để có thể thực hiện các lệnh, thao tác tính toán của mình.

Chú ý các thao tác với biến và hằng.

Thao tác với biến nhớ:

- Khai báo biến.
- Gán giá trị cho biến.
- Tính toán sử dụng biến nhớ.

Thao tác với hằng:

- Khai báo hằng (đồng thời gán giá trị ban đầu).
- Tính toán sử dụng hằng.

Chú ý: hằng không thể thay đổi giá trị trong suốt quá trình thực hiện chương trình.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. a) Hợp lệ; b) Không hợp lệ; c) Hợp lệ; d) Không hợp lệ.

Bài 2. Mặc dù đều cùng phải khai báo trước khi có thể sử dụng trong chương trình, sự khác nhau giữa biến và hằng là giá trị của hằng không thay đổi trong suốt quá trình thực hiện chương trình, còn giá trị của biến thì có thể thay đổi được tại từng thời điểm thực hiện chương trình.

Bài 3. Không thể gán lại giá trị 3.1416 cho π trong phần thân chương trình vì giá trị của hằng không thay đổi trong suốt quá trình thực hiện chương trình.

Bài 4. a) Hợp lệ; b) Không hợp lệ vì tên biến không hợp lệ;
c) Không hợp lệ vì hằng phải được cho giá trị khi khai báo;
d) Không hợp lệ vì biến không được gán giá trị khi khai báo, cách gán giá trị cũng không đúng cú pháp.

Bài 5. Các lỗi trong chương trình:

- 1) Thừa dấu bằng ở dòng 1 (chỉ cần dấu hai chấm);
- 2) Thừa dấu hai chấm ở dòng 2 (khi khai báo hằng chỉ cần dấu bằng);

3) Thiếu dấu chấm phẩy ở dòng 4;

4) Khai báo kiểu dữ liệu của biến b là số nguyên mà lệnh gán là phép chia hai số nguyên, kết quả luôn luôn là số thực, cho dù có chia hết hay không. Do đó cần phải khai báo biến b là biến có kiểu dữ liệu số thực thì mới đúng.

Bài 6. Cách khai báo hợp lí:

- a) Các biến a và h là kiểu số nguyên; biến S : kiểu số thực.
- b) Cả bốn biến a , b , c và d là các kiểu số nguyên.



Bài thực hành 3

KHAI BÁO VÀ SỬ DỤNG BIẾN

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Thực hiện được khai báo đúng cú pháp, lựa chọn được kiểu dữ liệu phù hợp cho biến;
- Kết hợp được giữa lệnh *write*, *writeln* với *read*, *readln* để thực hiện việc nhập dữ liệu cho biến từ bàn phím;
- Hiểu về các kiểu dữ liệu chuẩn: kiểu số nguyên, kiểu số thực;
- Sử dụng được lệnh gán giá trị cho biến;
- Hiểu cách khai báo và sử dụng hằng;
- Hiểu và thực hiện được việc trao đổi giá trị của hai biến.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Chương trình trong câu a, bài 1 là chương trình giả định số tiền phải trả bao gồm số tiền mua hàng (bằng đơn giá nhân với số lượng) và số tiền cước phí vận chuyển (cố định là 10000). Tổng số tiền phải trả bằng số tiền mua hàng cộng với cước phí.

Với bài này HS tập khai báo biến trong Pascal, cần cho HS tìm hiểu cú pháp khai báo biến, đặt tên đúng theo quy định của Pascal, chọn đúng kiểu dữ liệu của biến.

Rèn luyện soạn thảo, dịch, hiệu chỉnh, chạy và kiểm chứng kết quả cũng là một mục tiêu của bài này.

Cần hướng dẫn để HS tìm hiểu chức năng của lệnh *readln(tên biến)* để nhập giá trị của biến. Sự kết hợp của *write* và *readln* trong việc nhập giá trị biến từ bàn phím. Việc sử dụng biến trong biểu thức:

*thanh tien := soluong * dongia + phi.*

Các chú thích đặt trong cặp dấu ngoặc {} hoặc (* *) được dùng để giải thích câu lệnh, ý đồ của người viết chương trình. Gặp cặp dấu ngoặc này Pascal bỏ qua, không dịch những nội dung bên trong. Việc viết chú thích trong chương trình đôi khi rất cần thiết để giúp người khác có thể nhanh chóng hiểu được chương trình, thậm chí là để chính người đã viết ra chương trình dễ dàng hơn khi xem lại hoặc chỉnh sửa chương trình của mình.

Có thể gợi ý HS nhập số lượng là một số thực, ví dụ 6.5 chẳng hạn và giải thích hiện tượng xảy ra. Nguyên nhân là do kiểu dữ liệu nhập vào là số thực không phù hợp với kiểu của biến *Soluong* đã khai báo trong chương trình là số nguyên. Có thể gợi ý để HS thay đổi kiểu của biến số lượng đã khai báo để có thể nhập số lượng là một số thực.

Khi nhập bộ số liệu (1, 35000), kết quả không còn đúng nữa, nguyên nhân của hiện tượng này là do tràn số. Biến *Soluong* có kiểu là *integer* nên chỉ cho phép chứa các giá trị trong khoảng từ -32768 đến 32767, giá trị 35000 nằm ngoài phạm vi giá trị trên cho nên đã gây ra lỗi, kết quả đưa ra không chính xác. Có thể gợi ý cho HS chỉnh sửa khai báo kiểu dữ liệu để khắc phục hạn chế này.

b) Một trong những nội dung quan trọng của bài này là giúp HS luyện tập việc nhận biết và khai báo kiểu dữ liệu hợp lý cho các biến. Khai báo kiểu dữ liệu hợp lý một mặt sẽ giúp cho việc sử dụng bộ nhớ một cách tối ưu (ví dụ, với đại lượng chỉ nhận giá trị số tự nhiên không vượt quá 255 thì không cần thiết phải khai báo biến kiểu *integer*, mà chỉ cần kiểu *byte*), mặt khác giúp tránh lỗi tràn dữ liệu và dẫn đến kết quả sai. Trong các bài thực hành sau, GV nên lưu ý HS đến điểm này.

c) Với bài 2, trọng tâm của bài này là cho HS luyện tập với lệnh gán và thực hiện việc trao đổi giá trị của hai biến *x*, *y*. Đây là một công việc hay gặp trong lập trình và qua ví dụ này HS có thể hiểu rõ hơn về biến, cách sử dụng biến. Bài này cũng giới thiệu cách viết câu lệnh nhập nhiều dữ liệu từ bàn phím bằng một câu lệnh *readln* hoặc *read*.

Để thực hành bài này, có thể tiến hành như sau:

Cho HS gõ chương trình trong SGK, tiến hành dịch, chỉnh sửa và cho chạy chương trình.

Do không có thông báo cho người dùng về yêu cầu nhập giá trị tương ứng của các biến x , y nên HS có thể gặp khó khăn không biết nhập thế nào. GV cần hướng dẫn HS cú pháp của câu lệnh và cách nhập hai số nguyên (cách nhau bởi dấu cách) rồi nhấn phím **Enter** và quan sát kết quả.

Nên gợi ý cho HS cải tiến chương trình trên để hướng dẫn người dùng nhập giá trị cho x , y từ bàn phím. In ra màn hình giá trị của x , y vừa được người dùng nhập vào và in ra màn hình giá trị x , y sau khi đã trao đổi giá trị. Có thể tham khảo chương trình *Tinhtien.pas* để thực hiện việc này.

Về việc trao đổi giá trị giữa biến x và biến y , có thể lấy ví dụ minh họa như việc muốn trao đổi nước giữa hai cốc. Giả sử có hai cốc nước, một cốc nước chứa nước màu đỏ, một cốc nước chứa nước màu xanh. Làm thế nào để trao đổi nước giữa hai cốc nước này? Đương nhiên là phải dùng cốc thứ ba làm trung gian để thực hiện điều này, cụ thể: Giả sử cốc X chứa nước màu đỏ, cốc Y chứa nước màu xanh và cốc Z là cốc trung gian, không chứa gì cả. Cách trao đổi nước chứa trong cốc X và cốc Y như sau:

Đổ nước màu đỏ trong cốc X sang cốc Z;

Đổ nước màu xanh trong cốc Y sang cốc X;

Đổ nước màu đỏ trong cốc Z sang cốc Y.

Sau khi thực hiện như trên nước trong hai cốc đã được trao sang nhau.

Việc trao đổi giá trị của biến cũng tương tự, trong chương trình đã phải sử dụng biến z làm biến trung gian để lưu giữ giá trị ban đầu của biến x . Cụ thể:

```
z:=x; {Lưu giá trị x cho biến z}
x:=y; {Giá trị của biến x được thay bằng giá trị của biến y}
y:=z; {Giá trị của biến y được thay bằng giá trị của biến z,
      giá trị của biến z lúc này chính bằng giá trị của biến x
      ban đầu}
```

* Có một khác biệt cần được lưu ý ở đây. Khi đổ nước ở cốc Y sang nước ở cốc X thì cốc X có nước còn cốc Y hết nước. Khác với khi gán $X:=Y$ thì giá trị biến X bằng giá trị biến Y, nhưng giá trị biến Y không mất. Nếu HS thắc mắc thì GV có thể giải thích điều này, nếu HS không thắc mắc GV không nên giải thích để tránh làm phức tạp vấn đề.

Chương trình sau khi chỉnh sửa có thể như sau:

```

Program hoan_doi;
var x,y,z:integer;
begin
    write('Nhap gia tri bien x = '); readln(x);
    write('Nhap gia tri bien y = '); readln(y);
    Writeln('Truoc trao doi, gia tri cua bien x: ', x);
    Writeln('Truoc trao doi, gia tri cua bien y: ', y);
    {Bat dau thuc hien trao doi}
    z:=x;
    x:=y;
    y:=z;
    {Ket thuc trao doi}
    Writeln('Sau trao doi, gia tri cua bien x = ', x);
    Write('Sau trao doi, gia tri cua bien y = ', y);
    readln
end.

```

Lưu ý, để HS dễ tiếp thu, trong chương trình trên đã lựa chọn cách viết nhiều lệnh đơn giản, mặc dù có thể ghép một số lệnh thành một lệnh để chương trình ngắn gọn hơn. Ví dụ, có thể thay hai lệnh:

```

Writeln('Truoc trao doi, gia tri cua bien x = ', x);
Writeln('Truoc trao doi, gia tri cua bien y = ', y);

```

bằng một lệnh như sau:

```

Writeln('Truoc trao doi, gia tri cua bien x=', x, ' y=', y);

```

Bài 5. TỪ BÀI TOÁN ĐẾN CHƯƠNG TRÌNH

Thời lượng: 4 tiết

1. Mục đích, yêu cầu

- Biết khái niệm bài toán, thuật toán;
- Biết các bước giải bài toán trên máy tính;
- Xác định được Input, Output của một bài toán đơn giản;
- Biết chương trình là thể hiện của thuật toán trên một ngôn ngữ cụ thể;
- Biết mô tả thuật toán bằng phương pháp liệt kê các bước;
- Hiểu thuật toán tính tổng của N số tự nhiên đầu tiên, tìm số lớn nhất của một dãy số.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Bài toán và thuật toán để giải quyết bài toán là những nội dung rất quan trọng, nếu không nói là quan trọng nhất trong lập trình. Nếu HS nắm vững được những kiến thức này thì sẽ dễ dàng hơn rất nhiều trong việc tiếp thu kiến thức trong các bài sau. Vì lẽ đó các tác giả đề xuất dành thời lượng cho bài 5 là 4 tiết lí thuyết và 2 tiết bài tập. GV cần tận dụng hết thời gian dành cho bài này để truyền đạt cho HS một cách kĩ lưỡng.

b) Khái niệm bài toán và giải bài toán đã trở thành quen thuộc với HS trong các môn học khác như Toán, Vật lí,... Bài toán trong tin học không chỉ là những bài toán trong lĩnh vực toán học mà còn có thể là một nhiệm vụ, một công việc cần giải quyết trong cuộc sống thực tiễn (có khi không liên quan gì đến *toán học*) như: tính điểm trung bình một môn học, một học kì, nấu món ăn hay điều khiển rô-bốt nhặt rác chẳng hạn.

Để cho HS mở rộng nhận thức về khái niệm bài toán mà các em được biết ở môn học khác, có thể sử dụng lại ví dụ về bài toán điều khiển rô-bốt nhặt rác HS đã học ở bài 1. Ví dụ này cho thấy bài toán có thể còn là một công việc, một nhiệm vụ gắn liền với cuộc sống hằng ngày. Do đã được tiếp cận với ví dụ này ở bài học đầu tiên nên việc tìm ra điều kiện cho trước và kết quả thu được của bài toán này sẽ dễ dàng hơn với HS.

Để dẫn dắt đến khái niệm xác định bài toán, GV có thể dựa vào giả thiết và kết luận của một bài toán trong môn Toán để dẫn dắt HS xác định Input, Output của bài toán trong Tin học. Trong môn Toán, thường trước khi bắt đầu giải một bài toán các em đã quen với việc tìm giả thiết và kết luận của bài toán. Trong tin học, phần giả thiết là các điều kiện cho trước (Input), phần kết luận là kết quả cần thu được (Output).

Có thể sử dụng một bài toán đơn giản, quen thuộc với HS để HS dễ dàng tìm ra được điều kiện cho trước (giả thiết) và kết quả cần thu được (kết luận) của bài toán này (không nhất thiết cứ phải lấy ví dụ trong SGK).

c) Để máy tính có thể "giải" được bài toán thì con người cũng phải chỉ dẫn cho máy tính. Tuy nhiên, sự chỉ dẫn của con người để máy tính thực hiện phải rất cụ thể, chi tiết và đặc biệt là máy tính phải "hiểu" được những chỉ dẫn này. Có thể đặt câu hỏi với HS: Máy tính có "giải" được bài toán không? Có thể sử dụng ví dụ về rô-bốt nhặt rác để HS thảo luận tìm ra câu trả lời. Việc viết chương trình điều khiển máy tính rẽ phải, tiến, rẽ trái,

nhặt rác,... là do con người nghĩ ra, máy tính chỉ thực hiện những thao tác theo chỉ dẫn của con người.

Như vậy, con người tìm ra cách thức, chỉ ra các thao tác và trình tự thực hiện các thao tác để giải quyết công việc, máy tính chỉ biết thực hiện các thao tác theo chỉ dẫn. Máy tính không tự giải được bài toán.

Tập hợp các bước để điều khiển rô-bốt nhặt rác chính là một thuật toán.

d) Có thể mô tả thuật toán bằng cách liệt kê các bước như giới thiệu trong SGK hoặc bằng sơ đồ khối. Tuy nhiên, GV không cần giới thiệu thêm về cách mô tả bằng sơ đồ khối.

HS đã biết máy tính chỉ có thể hiểu được ngôn ngữ máy. Ngôn ngữ lập trình là ngôn ngữ con người sử dụng để viết chương trình. Vì vậy, có thể gợi ý để HS suy luận dẫn đến phải mô tả thuật toán bằng ngôn ngữ lập trình, cụ thể: cách mô tả thuật toán với ngôn ngữ tự nhiên thì chỉ có con người mới hiểu, máy tính không hiểu được. Để máy tính có thể hiểu và có thể thực hiện được thuật toán thì cần mô tả thuật toán bằng ngôn ngữ lập trình. Việc mô tả thuật toán bằng một ngôn ngữ lập trình để máy tính có thể hiểu, thực hiện được chính là viết chương trình.

Đề đi từ bài toán đến chương trình, SGK nêu ba bước: Xác định bài toán, mô tả thuật toán, viết chương trình. Đây là phương án chia bước đơn giản, HS dễ hiểu, dễ tiếp thu. Mặc dù có một số cách chia bước khác, tuy vậy để tránh làm phức tạp, GV không cần giới thiệu thêm hay phân tích sâu về các bước ở đây.

e) Nội dung mục 4 là quan trọng và tương đối khó với HS. Cần dành thời gian thích đáng cho mục này. Nên sử dụng tiết bài tập để bổ sung thêm thời lượng dạy học cho nội dung này. Mục tiêu của các ví dụ ở mục 4 là rèn luyện kỹ năng xác định Input, Output và mô tả thuật toán bằng cách liệt kê các bước cùng với việc giới thiệu thuật toán. Các thuật toán giới thiệu ở mục này là những thuật toán cơ bản HS cần tiếp thu. Trong đó, thuật toán tìm số lớn nhất của dãy số là thuật toán được yêu cầu cụ thể trong chuẩn kiến thức kỹ năng. Hơn nữa, các ví dụ trong mục này còn đề cập đến những thuật toán liên quan đến các cấu trúc rẽ nhánh, cấu trúc lặp và một số bài toán ở các bài sau.

Lưu ý rằng tất cả các thuật toán nêu trong mục này và trong phần câu hỏi và bài tập đều được sử dụng làm ví dụ minh họa hoặc để giải các câu hỏi và bài

tập trong các bài sau. Do vậy việc rèn luyện kĩ với các ví dụ này và giải một số bài tập sẽ tạo điều kiện thuận lợi để HS tiếp thu các nội dung tiếp theo.

Để giúp HS hiểu rõ về thuật toán, GV có thể mô phỏng thuật toán với một bộ dữ liệu cụ thể. Ví dụ:

Mô phỏng thuật toán tính tổng N số tự nhiên đầu tiên, với $N = 10$ (trong SGK, $N = 100$).

i	1	2	3	4	5	6	7	8	9	10	11
$i \leq N$	Đúng	Đúng	Đúng	Đúng	Đúng	Đúng	Đúng	Đúng	Đúng	Đúng	Sai
SUM	1	3	6	10	15	21	28	36	45	55	Kết thúc

Mô phỏng thuật toán tìm số lớn nhất trong dãy số cho trước:

Dãy số	5	3	4	7	6	3	15	9	11	
i	1	2	3	4	5	6	7	8	9	10
$i > n$	Sai	Sai	Sai	Sai	Sai	Sai	Sai	Sai	Sai	Đúng
$a_i > \text{MAX}$		Sai	Sai	Đúng	Sai	Sai	Đúng	Sai	Sai	Kết thúc
MAX	5	5	5	7	7	7	15	15	15	

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. Đáp án:

a) INPUT: Danh sách họ tên của HS trong lớp.

OUTPUT: Số HS có họ Trần.

b) INPUT: Dãy n số.

OUTPUT: Tổng của các phần tử lớn hơn 0.

c) INPUT: Dãy n số.

OUTPUT: Số các số có giá trị nhỏ nhất.

Bài 2. Sau ba bước, x có giá trị ban đầu của y và y có giá trị ban đầu của x , tức là giá trị của hai biến x và y được trao đổi cho nhau.

Bài 3. Mô tả thuật toán:

INPUT: Ba số dương $a > 0$, $b > 0$ và $c > 0$.

OUTPUT: Thông báo " a, b và c có thể là độ dài ba cạnh của một tam giác" hoặc thông báo " a, b và c không thể là độ dài ba cạnh của một tam giác".

Bước 1. Nếu $a + b \leq c$, chuyển tới bước 5.

Bước 2. Nếu $b + c \leq a$, chuyển tới bước 5.

Bước 3. Nếu $a + c \leq b$, chuyển tới bước 5.

Bước 4. Thông báo " a, b và c có thể là độ dài ba cạnh của một tam giác" và kết thúc thuật toán.

Bước 5. Thông báo " a, b và c không thể là độ dài ba cạnh của một tam giác" và kết thúc thuật toán.

Bài 4. Có thể giải bài toán này bằng cách sử dụng một biến phụ hoặc không dùng biến phụ.

Thuật toán 1. Sử dụng biến phụ z .

INPUT: Hai biến x và y .

OUTPUT: Hai biến x và y có giá trị không giảm.

Bước 1. Nếu $x \leq y$, chuyển tới bước 5.

Bước 2. $z \leftarrow x$.

Bước 3. $x \leftarrow y$.

Bước 4. $y \leftarrow z$.

Bước 5. Kết thúc thuật toán.

Thuật toán 2. Không sử dụng biến phụ (xem bài tập 2 ở trên).

INPUT: Hai biến x và y .

OUTPUT: Hai biến x và y có giá trị tăng dần.

Bước 1. Nếu $x \leq y$, chuyển tới bước 5.

Bước 2. $x \leftarrow x + y$.

Bước 3. $y \leftarrow x - y$.

Bước 4. $x \leftarrow x - y$.

Bước 5. Kết thúc thuật toán.

Bài 5. Đây là thuật toán tính tổng của các số tự nhiên từ 1 đến 100. Để HS tập trung vào hiểu thuật toán hơn là dành thời gian cho việc tính toán đơn

thuần, GV có thể thay cộng từ 1 đến 100 bằng cộng từ 1 đến 5 hoặc từ 1 đến 10.

Bài 6. Mô tả thuật toán:

INPUT: n và dãy n số a_1, a_2, \dots, a_n .

OUTPUT: Tổng $S = a_1 + a_2 + \dots + a_n$.

Bước 1. $S \leftarrow 0; i \leftarrow 1$.

Bước 2. $S \leftarrow S + a_i; i \leftarrow i + 1$.

Bước 3. Nếu $i \leq n$, quay lại bước 2.

Bước 4. Thông báo giá trị S và kết thúc thuật toán.

Bài 6. CÂU LỆNH ĐIỀU KIỆN

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Biết sự cần thiết của cấu trúc rẽ nhánh trong lập trình;
- Biết cấu trúc rẽ nhánh được sử dụng để chỉ dẫn cho máy tính thực hiện các thao tác phụ thuộc vào điều kiện;
- Biết cấu trúc rẽ nhánh có hai dạng: Dạng thiếu và dạng đủ;
- Biết mọi ngôn ngữ lập trình đều có câu lệnh để thể hiện cấu trúc rẽ nhánh;
- Hiểu cú pháp, hoạt động của các câu lệnh điều kiện dạng thiếu và dạng đủ trong Pascal;
- Bước đầu viết được câu lệnh điều kiện trong Pascal.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Trước khi bắt đầu bài học mới, GV có thể kiểm tra bài cũ và yêu cầu HS nhận xét về thứ tự thực hiện câu lệnh trong các chương trình đã học. GV cần nhấn mạnh cho HS biết rằng: các lệnh trong chương trình được thực hiện theo thứ tự từ trên xuống dưới. Thực hiện các lệnh tuần tự từ đầu đến cuối là thứ tự thực hiện ngầm định (và là cấu trúc điều khiển) của mọi ngôn ngữ lập trình.

b) Bắt đầu bài học mới, GV có thể xuất phát từ những hoạt động phụ thuộc vào điều kiện trong đời sống để dẫn dắt HS đến sự cần thiết cần có cấu trúc rẽ nhánh trong ngôn ngữ lập trình. Nhiều bài toán mà máy tính giúp con người giải quyết là những vấn đề của đời sống thực tiễn. Trong thực tiễn, một công việc có thể được thực hiện nếu như một điều kiện nào đó được thoả mãn. Vì vậy, mọi ngôn ngữ lập trình đều phải có cấu trúc điều khiển cho phép giải quyết các tình huống này. Nghĩa là, trong chương trình một số câu lệnh có thể được thực hiện hoặc không được thực hiện phụ thuộc vào một điều kiện cụ thể nào đó.

c) Cũng từ những ví dụ về hoạt động phụ thuộc vào điều kiện, GV có thể khái quát lên hai cấu trúc rẽ nhánh dạng thiếu và dạng đủ bằng ngôn ngữ tự nhiên như sau:

Nếu... thì...

Nếu... thì... ngược lại thì...

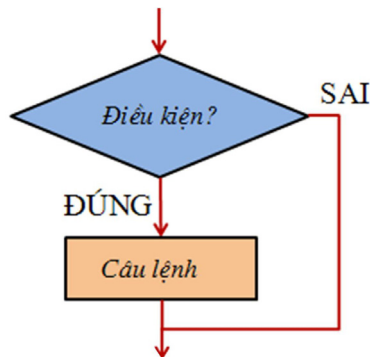
Tương ứng với hai cấu trúc rẽ nhánh dạng thiếu và dạng đủ, trong Pascal có hai câu lệnh điều kiện dạng thiếu và dạng đủ như sau:

Dạng thiếu: **if** <điều kiện> **then** <câu lệnh>;

Dạng đủ: **if** <điều kiện> **then** <câu lệnh 1> **else** <câu lệnh 2>;

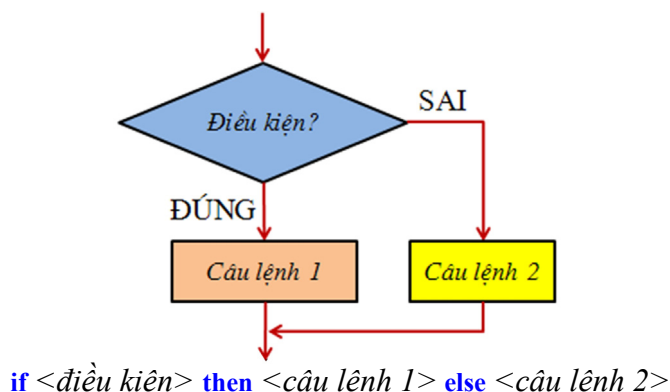
Sau đó giới thiệu cho HS về sơ đồ hoạt động của câu lệnh điều kiện dạng thiếu và dạng đủ.

Ở dạng thiếu: Nếu *điều kiện* thoả mãn thì *câu lệnh* được thực hiện, ngược lại thì bỏ qua *câu lệnh*.



if <điều kiện> **then** <câu lệnh>;

Ở dạng đủ: Nếu *điều kiện* thoả mãn thì *câu lệnh 1* được thực hiện, ngược lại thì thực hiện *câu lệnh 2*.



Trong đó, *câu lệnh*, *câu lệnh 1*, *câu lệnh 2* là câu lệnh của Pascal.

Điều kiện thường là phép so sánh (lưu ý, do HS THCS chưa học về biểu thức quan hệ nên ở đây dùng cụm từ *phép so sánh* để nói về *điều kiện* trong câu lệnh điều kiện). Phép so sánh cho kết quả là đúng tương đương điều kiện được thoả mãn, ngược lại phép so sánh cho kết quả sai tương đương với điều kiện không thoả mãn.

d) Trong SGK dành mục 2 để nói về điều kiện và phép so sánh. Trọng tâm của mục này là cần cho HS biết trong lập trình điều kiện thường được thể hiện bằng phép so sánh (biểu thức so sánh) và khái niệm điều kiện *được thoả mãn* (hay *không được thoả mãn*) trong đời sống tương đương với khái niệm phép so sánh cho kết quả là *đúng* (hay *sai*) trong ngôn ngữ lập trình.

Các phép so sánh, kí hiệu các phép so sánh trong Pascal đã được giới thiệu ở bài học trước. Tuy nhiên, HS chưa được luyện tập, do vậy GV cần lưu ý hướng dẫn HS luyện tập về phép so sánh cùng với việc tìm hiểu, tập viết câu lệnh điều kiện trong bài học này.

e) Đến đây GV có thể sử dụng các ví dụ ở mục 4 (SGK) để HS luyện tập nhằm hiểu rõ về hoạt động của câu lệnh điều kiện dạng thiếu, dạng đủ, biết ý nghĩa *câu lệnh*, *câu lệnh 1*, *câu lệnh 2* và *điều kiện*. Cần dành thời gian thích đáng cho HS luyện tập qua các ví dụ minh hoạ (khoảng 1 tiết học). Các ví dụ ở đây cần đơn giản, dễ hiểu để HS có thể dễ dàng nắm bắt được hoạt động của câu lệnh điều kiện, ý nghĩa của biểu thức điều kiện, câu lệnh. GV có thể chủ

động chọn các ví dụ khác, nhưng tránh ra những ví dụ quá phức tạp về điều kiện, nhiều phép so sánh.

f) Cuối bài học GV cần khái quát hoá để HS biết cấu trúc rẽ nhánh, hoạt động của cấu trúc rẽ nhánh là giống nhau ở mọi ngôn ngữ lập trình nhưng mỗi ngôn ngữ lập trình lại có những câu lệnh riêng để thể hiện cấu trúc rẽ nhánh.

Lưu ý: Hoàn toàn có thể theo các trình tự giới thiệu trong SGK để tiến hành dạy học bài học này. Tuy nhiên, tiến trình dạy học theo cách giới thiệu ở trên là một phương án khả thi. Phương án này đi từ câu lệnh cụ thể của Pascal, sau đó khái quát thành những kiến thức, nguyên tắc chung cho mọi ngôn ngữ lập trình. Căn cứ vào điều kiện thực tế, GV chủ động lựa chọn cách tiến hành phù hợp.

Việc dịch câu lệnh *if... then...* và *if... then... else...* ra tiếng Việt tương ứng là *nếu... thì...* và *nếu... thì... ngược lại thì...* có thể là cần thiết cho HS để nhớ ý nghĩa của câu lệnh, nhất là với HS chưa được học tiếng Anh.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. Có thể nêu vài ví dụ về các hoạt động hằng ngày phụ thuộc vào điều kiện. Dưới đây là một số ví dụ:

- a) Nếu đạt điểm tổng kết cả năm cao hơn 8.0, em sẽ đạt danh hiệu "*HS giỏi*".
- b) Nếu bị ốm, em (cần phải) đến phòng khám để bác sĩ khám bệnh.
- c) Nếu không được tưới đủ nước đúng thời kì phát triển, lúa sẽ không cho thu hoạch cao.

Bài 2. Đáp án: a) Đúng; b) Sai; c) Đúng; d) Sai, nếu $x \geq 1$ hoặc $x \leq -1$.

Bài 3. Giả sử *Điểm_1* là số điểm của người thứ nhất và *Điểm_2* là số điểm của người thứ hai, ngoài ra người thứ nhất nghĩ trong đầu một số tự nhiên $n < 10$.

Điều kiện ở trò chơi là người thứ hai đoán đúng số n . Khi đó *Điểm_2* được cộng thêm 1; ngược lại, *Điểm_2* được giữ nguyên. Tương tự, nếu người thứ hai nghĩ số tự nhiên m và điều kiện thứ hai là người thứ nhất đoán đúng số m đó. Khi đó *Điểm_1* được cộng thêm 1; ngược lại, *Điểm_1* được giữ nguyên.

Điều kiện ở trò chơi là sau 10 lần, nếu $Điểm_1 > Điểm_2$ thì người thứ nhất được tuyên bố thắng cuộc; ngược lại, người thứ hai thắng. Trường hợp $Điểm_1 = Điểm_2$ thì không có người thắng và người thua.

Bài 4. Điều kiện để điều khiển chiếc khay trong trò chơi là người chơi nhấn phím mũi tên \rightarrow hoặc \leftarrow . Nếu người chơi nhấn phím \rightarrow , biểu tượng chiếc khay sẽ di chuyển sang phải một đơn vị khoảng cách; nếu phím \leftarrow được nhấn, biểu tượng chiếc khay sẽ di chuyển sang trái. Nếu một phím khác ngoài hai phím mũi tên trên được nhấn, chiếc khay vẫn giữ nguyên vị trí.

Bài 5. a) Sai (thừa ":" chấm ở lệnh $x:=1$, thiếu ":" ở lệnh $a=b$);

b) Sai (thừa dấu chấm phẩy thứ nhất);

c) Đúng, nếu phép gán $m:=n$ không phụ thuộc điều kiện $x>5$; ngược lại, sai và cần đưa hai câu lệnh $a:=b$; $m:=n$; vào giữa cặp từ khoá *begin* và *end*;

d) Sai (thừa dấu chấm phẩy thứ nhất trước *else*).

Bài 6. a) Vì 45 chia hết cho 3, điều kiện được thoả mãn nên giá trị của X được tăng lên 1, tức là bằng 6;

b) Điều kiện không được thoả mãn nên câu lệnh không được thực hiện, tức là X giữ nguyên giá trị 5.

Bài 7. Các bước thực hiện thuật toán:

1) Khai báo biến nhớ n .

2) Nhập số n từ bàn phím.

3) Kiểm tra nếu n chia hết cho 2 thì thông báo n là số chẵn; ngược lại thông báo n là số lẻ.

Chương trình:

Var n: integer;

Begin

Write('Hay nhap so n: '); Readln(n);

if (n mod 2) = 0 then writeln(n, ' la so chan')

else witeln(n, ' la so le');

End.



Bài thực hành 4

SỬ DỤNG CÂU LỆNH ĐIỀU KIỆN

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Viết được câu lệnh điều kiện *if...then* trong chương trình;
- Rèn luyện kỹ năng ban đầu về đọc các chương trình đơn giản và hiểu được ý nghĩa của thuật toán sử dụng trong chương trình.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Bài này là bài đầu tiên HS thực hành sử dụng lệnh *if... then*. Do vậy, các ví dụ cần đơn giản, dễ hiểu để HS dễ dàng nhận ra ý nghĩa, hoạt động của câu lệnh điều kiện, biểu thức điều kiện đơn giản (phép so sánh), câu lệnh trong cấu trúc rẽ nhánh.

HS đã được làm quen với thuật toán này ở bài tập 4, bài 5. Do vậy, nói chung HS sẽ không gặp khó khăn về thuật toán này khi tìm hiểu chương trình *Sap_xep* ở bài 1 của bài thực hành này. Hoàn toàn có thể sử dụng chương trình *Sap_xep* để đạt mục tiêu thực hành sử dụng câu lệnh *if... then...else*.

Tuy nhiên, dưới đây xin giới thiệu một phương án về cơ bản vẫn dựa trên các yêu cầu của bài 1 nhưng có chỉnh sửa đôi chút để GV tham khảo khi tiến hành dạy học.

b) Giữ nguyên yêu cầu đề bài của bài 1. Đối với câu a, yêu cầu HS mô tả các bước để giải quyết bài toán. Các bước cơ bản để giải bài toán này là:

Bước 1. Nhập hai số nguyên a, b từ bàn phím;

Bước 2. Nếu $a \leq b$ thì hiển thị ra màn hình giá trị biến a trước rồi đến giá trị biến b ;

Bước 3. Nếu $b < a$ thì hiển thị ra màn hình giá trị biến b trước rồi đến giá trị biến a ;

Bước 4. Kết thúc.

Trên cơ sở phân mô tả thuật toán, GV hướng dẫn để HS viết được chương trình tương ứng. Chương trình có thể như sau:

```
program Sap_xep;  
uses crt;
```

```

var A, B, T: integer;
begin
    clrscr;
    {Buoc 1: Nhap hai so nguyen a, b tu ban phim}
    write('Nhap so A:'); readln(A);
    write('Nhap so B:'); readln(B);
    {Buoc 2: Neu a < b thi hien thi ra man hinh gia tri bien a
    truoc roi den gia tri bien b}
    if A<=B then write(A, ' ', B);
    {Neu b < a thi hien thi ra man hinh gia tri bien b truoc roi
    den gia tri bien a}
    if B<A then write(B, ' ', A);
    readln
End.

```

c) Thuật toán thực hiện trong chương trình này có thể sẽ gần với cách nghĩ của HS hơn. Do vậy, hi vọng HS sẽ thấy gần gũi và dễ dàng hơn trong việc tìm hiểu chương trình này cũng như hiểu được hoạt động, cách sử dụng lệnh rẽ nhánh trong chương trình này. Hơn nữa, về mặt sư phạm nên giới thiệu câu lệnh điều kiện dạng thiếu trước vì câu lệnh này đơn giản hơn câu lệnh điều kiện dạng đủ. Sau khi giới thiệu câu lệnh dạng thiếu, việc giới thiệu câu lệnh dạng đủ sẽ thuận lợi hơn.

Yêu cầu HS chạy chương trình và thử với các bộ số liệu có trong câu c, bài 1 (SGK).

d) Có một số điểm chính GV cần lưu ý khi cho HS tiến hành bài 2 như sau:

- Với câu lệnh điều kiện dạng đủ *if...then...else*, lưu ý không đặt dấu chấm phẩy sau câu lệnh trước từ khoá *else*. Trong Pascal, dấu chấm phẩy dùng để phân cách các câu lệnh (không phải là kết thúc câu lệnh).
- Đoạn chương trình:

```

If Long>Trang then writeln('Ban Long cao hon');
If Long<Trang then writeln('Ban Trang cao hon')
    else writeln('Hai ban cao bang nhau');

```

Thoạt nhìn, có thể theo cách suy luận của nhiều HS thì sự kết hợp của hai câu lệnh điều kiện là đảm bảo đầy đủ các trường hợp, đặc biệt câu lệnh thứ nhất đã xét đến trường hợp *Long>Trang* nên câu lệnh thứ hai sẽ không còn xét đến trường hợp này nữa mà chỉ xét đến hai trường hợp còn lại là *Long<Trang* và *Long = Trang*. Cách suy luận như vậy không phải là không có lí khi xét

trong trường hợp con người xử lý tình huống này. Tuy nhiên, máy tính không xử lý như vậy.

Khi máy tính thực hiện đến câu lệnh thứ hai là

```
If Long<Trang then writeln('Ban Trang cao hon')  
      else writeln('Hai ban cao bang nhau');
```

thì máy tính lại xét tất cả các trường hợp có thể. Vì vậy, đã dẫn đến lỗi thực hiện chương trình trong trường hợp Long cao hơn Trang.

e) Dưới đây xin phân tích rõ hơn về đoạn chương trình này, GV xem xét, tham khảo để giảng dạy cho HS. GV không cần dành quá nhiều thời gian giải thích quá chi tiết, tỉ mỉ như trình bày dưới đây.

Khi thực hiện đến câu lệnh thứ hai, máy tính thử kiểm tra điều kiện *Long<Trang* và có các trường hợp xảy ra như sau:

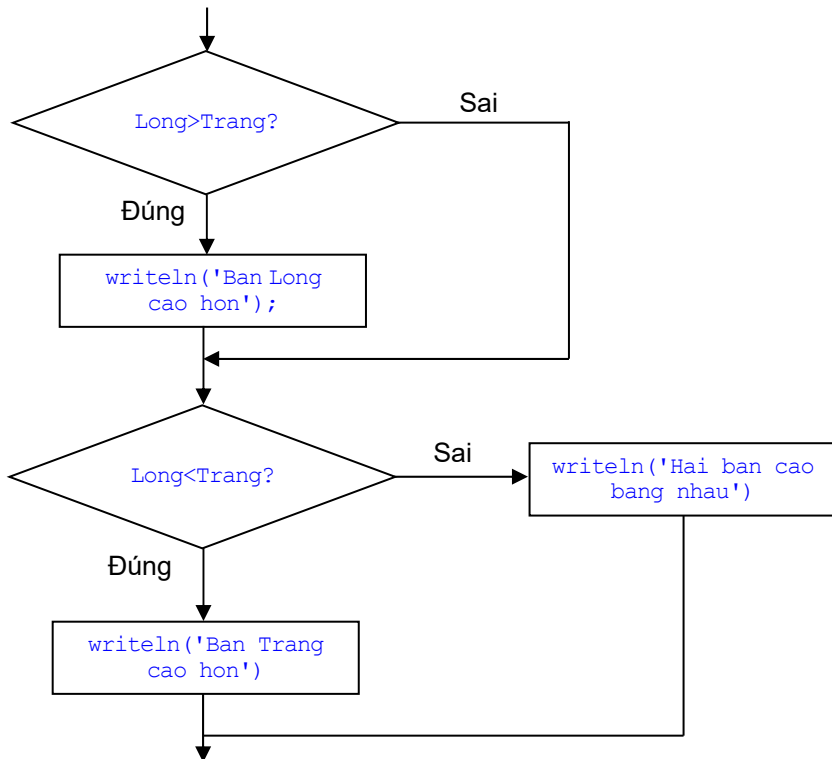
- Nếu *Long < Trang* cho kết quả đúng, tức là Trang cao hơn Long thì máy hiển thị ra màn hình dòng chữ "*Ban Trang cao hon*". Trong trường hợp này thì câu lệnh trước là

```
If Long>Trang then writeln('Ban Long cao hon');
```

đã không hiển thị ra màn hình dòng chữ "*Ban Long cao hon*" bởi vì biểu thức điều kiện *Long > Trang* trong câu lệnh điều kiện thiếu này cho kết quả sai. Kết quả là màn hình chỉ hiển thị một dòng chữ "*Ban Trang cao hon*".

- Nếu *Long < Trang* cho kết quả sai, Pascal thực hiện câu lệnh 2 in ra màn hình dòng chữ "*Hai ban cao bang nhau*". Tuy nhiên, có hai trường hợp dẫn đến biểu thức *Long < Trang* cho kết quả sai:
 - + Long và Trang cao bằng nhau, khi đó biểu thức điều kiện *Long > Trang* cũng cho kết quả sai và câu lệnh điều kiện thứ nhất không hiển thị ra màn hình dòng chữ nào cả. Kết quả là màn hình chỉ có một dòng chữ "*Hai ban cao bang nhau*".
 - + Long cao hơn Trang, khi đó biểu thức điều kiện *Long>Trang* ở câu lệnh điều kiện thứ nhất cho kết quả đúng. Vì vậy, câu lệnh điều kiện thứ nhất đã hiển thị ra màn hình dòng chữ "*Ban Long cao hon*". Như vậy, màn hình sẽ hiển thị ra hai dòng thông báo khác nhau là "*Ban Long cao hon*" và "*Hai ban cao bang nhau*". Đây chính là một lỗi ngữ nghĩa của chương trình cần được chỉnh sửa.

Sơ đồ thực hiện hai lệnh trên được mô tả trong hình dưới đây:



Về cách dạy phần này, nên cho HS gõ chương trình vào máy và chạy thử với các bộ dữ liệu kiểm tra. Các bộ số liệu cần phủ kín các trường hợp: Trang cao hơn Long, hai bạn cao bằng nhau và Long cao hơn Trang. Yêu cầu HS quan sát kết quả để phát hiện vấn đề và tìm hiểu chương trình, phát hiện lỗi.

f) Thực ra, việc thử chương trình với một số bộ dữ liệu mẫu là một trong các công đoạn của phát triển phần mềm, còn gọi là bước kiểm thử. Sau khi lập trình xong, phần mềm cần được thử nghiệm với một số bộ dữ liệu mẫu (hay còn gọi là bộ test) mà người ta dễ dàng biết được kết quả để kiểm chứng với kết quả mà chương trình đưa ra.

Việc thử chương trình với một bộ dữ liệu test chỉ chứng minh được chương trình sai mà không chứng minh được chương trình là đúng. Nghĩa là khi thử với bộ dữ liệu test nếu chương trình cho kết quả sai khác với kết quả đã được biết trước thì kết luận là chương trình sai, nhưng nếu thử với một số bộ dữ liệu test mà chương trình cho kết quả đúng với kết quả đã được biết trước thì chưa thể kết luận chương trình hoàn toàn đúng đắn được mà chỉ có thể nói

chưa phát hiện ra sai sót của chương trình mà thôi. Tuy nhiên, người ta có thể có cách khác để chứng minh tính đúng đắn của chương trình mà không chỉ sử dụng đến các bộ dữ liệu test.

Có hai cách để chỉnh sửa chương trình trên để đảm bảo chỉ đưa ra một thông báo đúng.

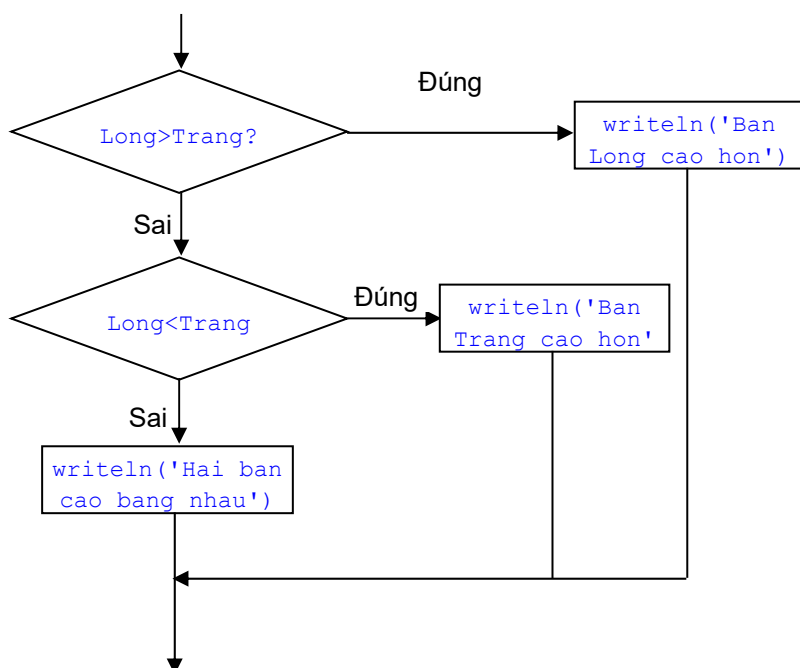
Cách đơn giản nhất là sử dụng ba câu lệnh điều kiện dạng thiếu như sau:

```
If Long>Trang then writeln('Ban Long cao hon');  
If Long=Trang then writeln('Hai ban cao bang nhau');  
If Long<Trang then writeln('Ban Trang cao hon');
```

Cách thứ hai là sử dụng câu lệnh điều kiện lồng nhau như trong SGK, mục đích của cách này là giới thiệu cho HS có thể sử dụng các câu lệnh điều kiện lồng nhau:

```
If Long>Trang then writeln('Ban Long cao hon') else  
If Long<Trang then writeln('Ban Trang cao hon')  
else writeln('Hai ban cao bang nhau');
```

Có thể sử dụng sơ đồ dưới đây để giải thích về hoạt động của hai câu lệnh điều kiện lồng nhau để HS hiểu được lý do câu lệnh này chỉ hiển thị ra màn hình một thông báo.



g) Với bài 3, HS cần biết điều kiện để ba số dương a, b, c là độ dài ba cạnh của một tam giác thì tổng hai cạnh phải lớn hơn cạnh còn lại, nghĩa là phải đồng thời thoả mãn ba điều kiện $a + b > c$, $b + c > a$ và $c + a > b$. GV hướng dẫn HS về cách biểu diễn ba điều kiện này trong Pascal:

$a+b>c$) **and** ($b+c>a$) **and** ($c+a>b$)

Điểm khó của bài này là HS biết chuyển biểu thức điều kiện toán học sang biểu diễn trong Pascal. Cần giải thích HS hiểu dùng phép quan hệ *and* là để đảm bảo cả ba điều kiện $a + b > c$, $b + c > a$ và $c + a > b$ đồng thời thoả mãn. Việc phải sử dụng dấu ngoặc đơn trong phép so sánh trên là để đảm bảo thứ tự ưu tiên thực hiện phép toán và để đảm bảo tham số của phép *and* (và *or*) chỉ có thể là giá trị đúng hoặc sai (không là số).

Cần cho HS đọc, thảo luận kĩ để hiểu chương trình này. Bài toán này là một trong các bài toán yêu cầu trong chuẩn kiến thức, kĩ năng. Sau khi học bài này, HS phải hiểu và phải tự viết chương trình giải bài toán tương tự (ví dụ kiểm tra tính chất (cân, đều, vuông) của tam giác dựa trên số đo của các cạnh).

Để HS luyện tập thêm về câu lệnh điều kiện, phép so sánh, có thể yêu cầu HS viết chương trình cho phép nhập điểm bài kiểm tra của một bạn nào đó, rồi thực hiện:

- Nếu điểm nhỏ hơn 5, in ra dòng chữ "*Ban can co gang hon*";
- Nếu điểm lớn hơn hoặc bằng 5 và nhỏ hơn 6.5, in ra dòng chữ "*Ban dat diem trung binh*";
- Nếu điểm lớn hơn hoặc bằng 6.5 và nhỏ hơn 8, in ra dòng chữ "*Ban dat diem kha*";
- Nếu điểm lớn hơn hoặc bằng 8, in ra dòng chữ "*Hoan ho, ban dat diem gioi*".

Một ví dụ khác là yêu cầu HS viết chương trình giải phương trình bậc nhất $ax + b = 0$, với $a \neq 0$ và a, b nhập từ bàn phím. Lưu ý, phương trình bậc nhất một ẩn số được giới thiệu trong chương trình môn Toán lớp 8 (học kì II). Do vậy, nếu HS chưa được học phương trình này ở môn Toán, GV nên dành một vài phút để HS làm quen với khái niệm này.

Bài 7. CÂU LỆNH LẶP

Thời lượng: 3 tiết

1. Mục đích, yêu cầu

- Biết nhu cầu cần có cấu trúc lặp trong ngôn ngữ lập trình;
- Biết ngôn ngữ lập trình dùng cấu trúc lặp để chỉ dẫn máy tính thực hiện lặp đi lặp lại công việc nào đó một số lần;
- Hiểu hoạt động của câu lệnh lặp với số lần biết trước *for...do* trong Pascal;
- Viết đúng được lệnh *for...do* trong một số tình huống đơn giản;
- Biết lệnh ghép trong Pascal.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Giống với cấu trúc rẽ nhánh, cần xuất phát từ những hoạt động trong đời sống thực tiễn có tính chất lặp đi lặp lại để HS hiểu về khái niệm lặp. Ví dụ, tiếng gà trống gáy, tiếng chim hót, tiếng chuông đồng hồ báo thức gọi em dậy mỗi buổi sáng; Các ngày trong tuần các em đều lặp đi lặp lại các hoạt động buổi sáng đến trường và buổi chiều trở về nhà; Hoặc trên lớp, giờ trả bài kiểm tra cô giáo lặp đi lặp lại việc gọi tên HS và ghi điểm của HS vào sổ điểm, cô giáo sẽ ngừng lại khi đã vào điểm cho tất cả HS trong lớp.

Giả sử cô giáo đề nghị em viết chương trình Pascal để in lời chào từng bạn của lớp em (hoặc của nhóm em), cụ thể chương trình cho phép từng bạn nhập tên của mình từ bàn phím và in ra lời chào tương ứng, ví dụ khi một bạn nhập tên là *Mai*, thì chương trình sẽ in ra "*Chao ban Mai*", một bạn khác nhập tên là *Trung* thì sẽ in ra "*Chao ban Trung*". Như vậy em sẽ cần viết một chương trình Pascal cho phép lặp đi lặp lại việc nhập tên và hiển thị ra màn hình lời chào. Làm thế nào để chương trình Pascal của em có thể thực hiện việc lặp này?

Giả sử lớp của em có 40 bạn, em hoàn toàn có thể viết 40 lần lệnh để nhập tên và lệnh hiển thị dòng chào. Các lệnh này hoàn toàn giống nhau. Tuy nhiên, một chương trình như vậy thì vừa dài, vừa nhàm chán, dễ sai sót.

Trong Pascal cung cấp một câu lệnh lặp như sau:

for <biến đếm>:= <giá trị đầu> **to** <giá trị cuối> **do** <câu lệnh>;

trong đó:

- *biến đếm* là biến đơn có kiểu nguyên;

- *giá trị đầu* và *giá trị cuối* là các *biểu thức* có cùng kiểu với *biến đếm*;
- *câu lệnh* có thể là câu lệnh đơn giản hay câu lệnh ghép.

Nói chung, câu lệnh lặp sẽ thực hiện *câu lệnh* sau từ khoá *do* nhiều lần, mỗi lần là một vòng lặp. Số vòng lặp là biết trước và bằng

$$\text{giá trị cuối} - \text{giá trị đầu} + 1.$$

Trong trường hợp *giá trị đầu* lớn hơn *giá trị cuối* thì *câu lệnh* không được thực hiện lần nào.

Khi thực hiện, ban đầu biến đếm sẽ nhận giá trị là *giá trị đầu*, sau mỗi vòng lặp, biến đếm được tự động tăng thêm một đơn vị cho đến khi bằng *giá trị cuối*.

Lưu ý:

- Để tránh phức tạp, gây khó hiểu với HS, mô tả hoạt động của lệnh *for...do* ở trên được ngầm định *biến đếm*, *giá trị đầu*, *giá trị cuối* là số nguyên. Về lý thuyết, biến đếm, giá trị đầu, giá trị cuối có thể thuộc một kiểu dữ liệu bất kì có thứ tự (ví dụ kiểu số nguyên, kiểu kí tự hoặc kiểu đoạn con của số và kí tự), tuy nhiên, ở đây không đề cập đến những vấn đề này với mục đích để giản lược nội dung, giúp HS dễ tiếp thu kiến thức mà vẫn đảm bảo những kiến thức, kĩ năng cần thiết theo yêu cầu.
- Trong Pascal cấu trúc *for...do* có hai dạng tiến và lùi:

Dạng tiến:

for <biến đếm>:= <giá trị đầu> **to** <giá trị cuối> **do** <câu lệnh>;

Dạng lùi:

for <biến đếm>:= <giá trị cuối> **downto** <giá trị đầu> **do** <câu lệnh>;

Trong SGK chỉ giới thiệu dạng tiến. Về cơ bản dạng tiến gần gũi với cách suy nghĩ tự nhiên của HS THCS hơn và chỉ cần dạng tiến là đủ, không yêu cầu phải giới thiệu thêm dạng lùi.

b) Khi thực hiện câu lệnh lặp *for...do* các *giá trị đầu* và *giá trị cuối* phải được xác định trước. Chính vì thế mà ta biết trước được số lần thực hiện câu lệnh sau từ khoá *do* (số lần lặp bằng *giá trị cuối* – *giá trị đầu* + 1). Chính vì vậy, câu lệnh *for...do* còn được gọi là câu lệnh lặp với số lần biết trước.

c) GV có thể sử dụng chương trình *Lap* trong SGK (ví dụ 3), phân tích ví dụ này để cho HS hiểu rõ về hoạt động của câu lệnh lặp, hiểu về biến đếm, giá trị đầu, giá trị cuối và câu lệnh.

Có thể hướng dẫn HS lập bảng quá trình thực hiện chương trình trên như dưới đây:

Lần lặp thứ	i	Kết quả viết ra màn hình
1	1	Day la lan lap thu 1
2	2	Day la lan lap thu 2
3	3	Day la lan lap thu 3
4	4	Day la lan lap thu 4
5	5	Day la lan lap thu 5
6	6	Day la lan lap thu 6
7	7	Day la lan lap thu 7
8	8	Day la lan lap thu 8
9	9	Day la lan lap thu 9
10	10	Day la lan lap thu 10

d) Sau khi cùng với HS phân tích và thực hiện các chương trình ở ví dụ 3 và 4, GV có thể cùng HS sử dụng câu lệnh *for...do* để viết đoạn câu lệnh nhập tên và hiển thị ra màn hình lời chào cho các bạn trong lớp. Giả sử lớp có 40 bạn thì chương trình có thể được viết như sau:

```
Program Chao_hoi;  
var i: integer; ten: string;  
begin  
  for i:= 1 to 40 do  
    Begin  
      write('Nhap ten cua ban: '); Readln(ten);  
      writeln('Chao ban ', ten);  
    end;  
end.
```

Lưu ý: Có thể giới thiệu nhanh về câu lệnh **ghép** (lệnh nằm trong hai từ khoá **begin ... end**). Khác với chương trình *Lap*, sau từ khoá *do* chỉ có một câu lệnh cần thực hiện, ở chương trình *Chao_ho*, (sau từ khoá *do* có hai câu lệnh cần thực hiện). Muốn vậy, hai câu lệnh này cần phải được "gói" trong cặp từ khoá *begin...end*. Cách viết này của Pascal được gọi là câu lệnh ghép. Câu lệnh **ghép** có thể chứa nhiều câu lệnh khác của Pascal. Lưu ý HS trong cấu trúc câu lệnh ghép này sau *end* là dấu chấm phẩy (;), không phải là dấu chấm (.).

e) Cần lấy thêm một số ví dụ khác để HS biết và tập làm quen với các tình huống sử dụng câu lệnh *for...do* và lệnh ghép. Có thể yêu cầu HS đọc hiểu ví dụ có trong SGK (như chương trình *Tinh_tong*, *Tinh_Giai_thua*) hoặc đưa ra bài toán đơn giản cần sử dụng đến câu lệnh *for...do* và hướng dẫn HS viết chương trình.

f) GV cần khái quát cho HS cấu trúc lặp với số lần biết trước có ở mọi ngôn ngữ lập trình, mỗi ngôn ngữ lập trình có câu lệnh riêng để mô tả cấu trúc này. Trên đây các em đã được tìm hiểu về câu lệnh lặp với số lần biết trước trong Pascal (*for...do*).

g) Căn cứ vào tình hình tiếp thu của HS, GV cần lựa chọn và giao một số bài tập cho HS luyện tập, không nhất thiết phải làm hết tất cả các bài tập cuối bài này.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. Có thể nêu rất nhiều ví dụ về các hoạt động lặp. Dưới đây là một số ví dụ:

- Hàng ngày em đặt đồng hồ báo thức lúc 6 giờ để dậy sớm tập thể dục.
- Hàng ngày (hoặc hằng tuần) bác lái xe khách lái xe để chuyên chở hành khách xuất phát vào một thời gian và từ một địa điểm nhất định và đi theo một tuyến đường đã xác định trước.
- Mỗi lần được khởi động, máy tính của em sẽ thực hiện cùng các hoạt động tự kiểm tra các thành phần máy tính, sau đó khởi động hệ điều hành theo một trình tự đã được quy định trước.

Bài 2. Chương trình này không thực hiện hoạt động nào mặc dù có hẵn một lệnh lặp.

Bài 3. Thuật toán tính tổng $A = \frac{1}{1.3} + \frac{1}{2.4} + \frac{1}{3.5} + \dots + \frac{1}{n(n+2)}$.

Bước 1. $A \leftarrow 0, i \leftarrow 1$.

Bước 2. $A \leftarrow A + \frac{1}{i(i+2)}$.

Bước 3. $i \leftarrow i + 1$.

Bước 4. Nếu $i \leq n$, quay lại bước 2.

Bước 5. Ghi kết quả A và kết thúc thuật toán.



Bài thực hành 5

SỬ DỤNG LỆNH LẶP FOR...DO

Thời lượng: 3 tiết

1. Mục đích, yêu cầu

- Viết được chương trình có sử dụng lệnh lặp *for...do*;
- Sử dụng được câu lệnh ghép;
- Rèn luyện kỹ năng đọc hiểu chương trình có sử dụng lệnh lặp *for...do*.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Với bài 1 của bài thực hành này, GV nên lưu ý HS tìm hiểu câu lệnh:

```
for i:=1 to 10 do writeln(N, ' x ', i:2, ' = ', N*i:3);
```

Đồng thời cần giúp HS nhận thấy được sự thay đổi của biến đếm i và các tham số của câu lệnh *write* để viết ra bảng nhân. Các tham số :2, :3 chỉ có ý nghĩa trong việc quy định quy cách trình bày bảng nhân trên màn hình, lưu ý HS không cần quan tâm đến quy cách trình bày, chỉ cần quan tâm đến sự thay đổi của biến đếm i , thông tin được lệnh *writeln* viết ra màn hình.

Có thể cho HS thảo luận theo nhóm để hoàn thành một bảng tiến trình thực hiện của câu lệnh trên như sau:

Giả sử với $N = 3$:

Bước	i	$i \leq 10$?	writeln(N, 'x' , i , ' = ', N*i)
1	1	Đúng	3x1 = 3
2	2	Đúng	3x2 = 6
3	3	Đúng	3x3 = 9
4	4	Đúng	3x4 = 12
5	5	Đúng	3x5 = 15
6	6	Đúng	3x6 = 18
7	7	Đúng	3x7 = 21
8	8	Đúng	3x8 = 24
9	9	Đúng	3x9 = 27
10	10	Đúng	3x10 = 30
11	11	Sai	Không thực hiện lệnh writeln. Kết thúc vòng lặp

b) Bài 2 cung cấp cho HS một câu lệnh mới là thủ tục đưa con trỏ tới một vị trí mong muốn trên màn hình (màn hình soạn thảo văn bản) *GotoXY*. Giới thiệu cùng với thủ tục *GotoXY* là các hàm lấy vị trí cột *WhereX*, vị trí hàng *WhereY* hiện thời của con trỏ. Việc giới thiệu thủ tục này nhằm cung cấp cho HS một công cụ để trình bày trên màn hình. Hơn thế nữa, việc giới thiệu hàm, thủ tục ở đây còn nhằm mục đích hướng dẫn HS tìm hiểu về thư viện chương trình, sử dụng, khai thác hàm, thủ tục có sẵn trong Pascal. Tuy nhiên, đây không phải là yêu cầu bắt buộc trong Chuẩn kiến thức, kỹ năng cho nên GV có thể cho thực hành bài này trên lớp hoặc giao cho HS tự tìm hiểu. Không cần đi sâu vào việc sử dụng các thủ tục này để trình bày kết quả trên màn hình.

c) Bài 3 giới thiệu về việc sử dụng hai lệnh lặp *for...do* lồng nhau. GV có thể sử dụng bài 3 này hoặc lấy một ví dụ khác để giới thiệu về lệnh lặp *for...do* lồng nhau. Có một ví dụ vui thường được sử dụng để minh họa cho việc sử dụng lệnh lặp *for...do* lồng nhau, đó là bài toán cổ:

*Vừa gà vừa chó
Bó lại cho tròn
Ba mươi sáu con
Một trăm chân chẵn.*

Bài toán cổ này có thể sẽ làm HS hứng thú hơn. Lưu ý, HS lớp 8 chưa được học giải hệ phương trình bậc nhất hai ẩn số.

Chương trình giải bài toán này có thể như sau:

```
Var ga, cho: byte;  
Begin  
  for ga:=1 to 35 do  
    for cho:=1 to 35 do  
      if (ga*2 + cho*4 = 100) and (ga + cho = 36) then  
        writeln('So ga la: ', ga, '; So cho la: ', cho);  
  Readln  
End.
```

GV có thể giới thiệu chương trình trước rồi yêu cầu HS tìm hiểu, giải thích tại sao chương trình này cho phép giải bài toán đặt ra.

Thuật toán này rất đơn giản, ý tưởng cơ bản là xét tất cả các trường hợp số lượng gà và chó bằng 36 và kiểm tra xem trường hợp nào thoả mãn: $ga + cho = 36$ và $ga*2 + cho*4 = 100$ thì đó là một đáp số của bài toán.

Qua bài toán này cũng có thể nêu cho HS thấy ưu điểm nổi bật của máy tính trong việc tính toán nhờ tốc độ xử lý rất cao. Với cách giải như trên máy tính tìm ra kết quả nhanh chóng, nhưng nếu để con người thực hiện thì sẽ lâu hơn rất nhiều.

Tuy nhiên, nhược điểm của máy tính là chỉ biết làm việc theo sự điều khiển của con người mà không hề có tư duy sáng tạo. Trong quá trình tính toán tìm ra kết quả, con người còn có khả năng phán đoán, dự đoán xu hướng để có thể bỏ qua một số công đoạn tính toán nhằm đi đến kết quả nhanh hơn. Do đó, con người cần lựa chọn, xây dựng thuật toán sao cho có thể nâng cao hiệu quả làm việc của máy tính.

GV có thể yêu cầu HS cải tiến để có chương trình hiệu quả hơn.

```
var ga, cho:byte;  
Begin  
For cho:= 1 to 24 do  
  Begin  
    ga:= 36 - cho;  
    if (2*ga + 4*cho = 100) then  
      writeln('Ga: ', ga, ', Cho: ', cho);  
    end;  
  readln  
End.
```

GV có thể phân tích, hướng dẫn để HS nhận thấy số lượng các phép tính ở chương trình sau ít hơn với chương trình ban đầu. Điều đó cũng có nghĩa là thuật toán ở chương trình sau hiệu quả hơn. Việc xây dựng, lựa chọn thuật toán hiệu quả có vai trò quan trọng trong lập trình, nhất là với những bài toán có khối lượng tính toán lớn. Luôn cần có ý thức xây dựng, lựa chọn thuật toán hiệu quả nhất khi giải bài toán trên máy tính.

Việc phân tích về số lượng phép toán có thể gây quá tải đối với HS. Do vậy, GV căn cứ vào mức độ tiếp thu của HS để có thể tiến hành giới thiệu hoặc không giới thiệu nội dung về số lượng phép toán, so sánh tính hiệu quả giữa các thuật toán.

Bài 8. LẶP VỚI SỐ LẦN CHƯA BIẾT TRƯỚC

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Biết nhu cầu cần có cấu trúc lặp với số lần chưa biết trước trong ngôn ngữ lập trình;
- Biết ngôn ngữ lập trình dùng cấu trúc lặp với số lần chưa biết trước để chỉ dẫn máy tính thực hiện lặp đi lặp lại công việc đến khi một điều kiện nào đó được thoả mãn;
- Hiểu hoạt động của câu lệnh lặp với số lần chưa biết trước *while...do* trong Pascal.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Giống với bài 7, ở phần đầu của bài này GV cần nêu một số ví dụ về hoạt động lặp với số lần chưa biết trước. Ví dụ ở phần đầu trong SGK là một hoạt động trong đời sống. Tương tự như vậy với phần thuật toán nhập số n từ bàn phím. Chẳng hạn số nhập vào vẫn nhỏ hơn 5 thì chẳng đờ người dùng sẽ còn tiếp tục thực hiện thao tác nhập số n vào từ bàn phím. GV có thể lưu ý HS về việc người ta thường sử dụng thủ thuật này trong kĩ thuật lập trình để thực hiện lặp đi lặp lại một đoạn chương trình nào đó như ví dụ sẽ được trình bày ở mục b dưới đây.

Ví dụ 1 trong SGK là một bài toán toán học thuần túy.

Trong ví dụ 1, sau khi giới thiệu thuật toán SGK một cách khái quát, đưa ra sơ đồ hoạt động của cấu trúc lặp với số lần chưa biết trước, SGK giới thiệu câu lệnh *while...do* của Pascal như một ví dụ minh hoạ. HS được làm quen với cách sử dụng lệnh *while...do* qua các ví dụ.

b) Dưới đây gợi ý một cách tiến hành khác để GV tham khảo. Cách tiếp cận này được thực hiện theo phương án đi từ câu lệnh lặp cụ thể *while...do* trong Pascal, sau đó khái quát thành kiến thức chung ở các ngôn ngữ lập trình.

Dưới đây là chương trình cho một nhóm bạn HS (ví dụ gồm 5 người) lần lượt nhập tên của mình để thực hiện "màn chào hỏi" với máy tính:

```
Program Chao_hoi;  
uses crt;  
var i: integer;
```

```

        Ten: string;
Begin
  for i:=1 to 5 do
    Begin
      write('Nhap ten cua ban: '); Readln(Ten);
      writeln('Chao ban ', Ten);
      readln;
    end;
  End.

```

Sau khi phân tích chương trình này, GV có thể đặt tình huống nếu chưa biết trước số bạn trong nhóm thì phải viết chương trình như thế nào?

Nếu HS chưa đưa ra được phương án giải quyết cho vấn đề này. GV nên chủ động đưa ra một chương trình Pascal như sau (có thể lưu chương trình này với tên *Chao_hoi.pas*):

```

Program Chao_hoi;
uses crt;
var Tieptuc: char;
    Ten: string;
Begin
  Tieptuc:='c';
  while tieptuc = 'c' do
    Begin
      write('Nhap ten cua ban: '); Readln(Ten);
      writeln('Chao ban ', Ten);
      write('Tiep tuc? c/k'); readln(Tieptuc);
    end;
  readln
End.

```

Dựa trên chương trình này, GV giới thiệu về cú pháp, sơ đồ hoạt động của câu lệnh *while...do*.

c) Trong Pascal, cú pháp câu lệnh lặp với số lần chưa biết trước có dạng:

```

while <điều kiện> do <câu lệnh>;

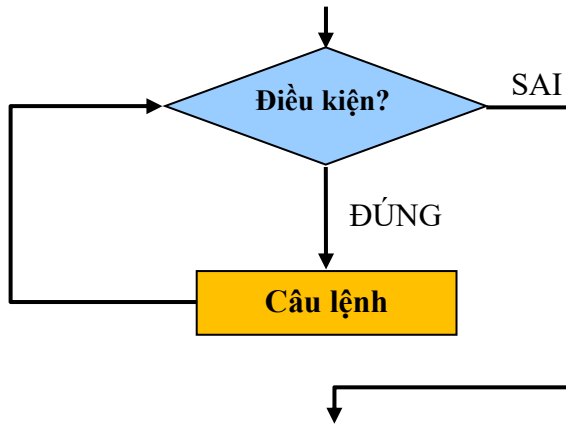
```

trong đó:

- *điều kiện* thường là một phép so sánh;
- *câu lệnh* có thể là câu lệnh đơn giản hay câu lệnh ghép.

Câu lệnh lặp này được thực hiện như sau:

1. Kiểm tra *điều kiện*.
2. Nếu *điều kiện* SAI, *câu lệnh* sẽ bị bỏ qua. Nếu *điều kiện* ĐÚNG, thực hiện *câu lệnh* và quay lại bước 1.



Sơ đồ hoạt động của câu lệnh lặp với số lần chưa biết trước

Việc dịch nghĩa hay diễn giải ý nghĩa của từ tiếng Anh trong câu lệnh *while...do* có thể là cần thiết đối với HS chưa được học tiếng Anh và để HS dễ nhớ (*trong khi ... còn đúng thì thực hiện*).

d) Đến đây, GV có thể khái quát cho HS biết rằng các ngôn ngữ lập trình đều có câu lệnh lặp với số lần chưa biết trước, hoạt động của câu lệnh lặp với số lần chưa biết trước ở các ngôn ngữ lập trình là giống nhau. Điểm khác nhau giữa các ngôn ngữ lập trình là cú pháp câu lệnh để thể hiện cấu trúc này.

Phần cuối bài, GV sử dụng các ví dụ trong SGK hoặc lấy ví dụ khác để HS hiểu được hoạt động, viết đúng cú pháp và biết một số trường hợp sử dụng hiệu quả của câu lệnh *while...do*.

Lưu ý:

– Đối với lệnh lặp *while...do*, trong các câu lệnh của vòng lặp này cần có câu lệnh làm thay đổi biểu thức điều kiện, có nghĩa là phải có câu lệnh để đến lúc nào đó điều kiện không được thoả mãn, khi đó vòng lặp kết thúc.

Ví dụ trong chương trình *Chao_hoi.pas* ở trên, điều kiện không được thoả mãn và vòng lặp kết thúc khi điều kiện *tieptuc = 'c'* cho kết quả sai, tức là *tieptuc <> 'c'*. Câu lệnh *readln(Tieptuc)* để gán giá trị cho biến *Tieptuc* từ bàn phím, giá trị của biểu thức điều kiện thay đổi phụ thuộc vào câu trả lời của người dùng. Khi người dùng nhấn "c" thì vòng lặp tiếp tục, ngược lại, khi người dùng nhấn bất kì chữ cái (hoặc chữ số) nào khác thì vòng lặp kết thúc.

– Trong trường hợp điều kiện luôn được thoả mãn, nghĩa là người dùng luôn luôn nhấn phím *c*, khi đó chương trình này sẽ tiếp tục lặp đi lặp lại. GV có thể sử dụng chương trình ở mục 2 SGK để giải thích về việc lặp vô hạn có thể xảy ra do lỗi của người lập trình:


```

var a:integer;
begin
    a:=5;
    while a<6 do writeln('A');
end.

```

– Các câu lệnh của vòng lặp có thể không được thực hiện một lần nào cả là một đặc điểm khá quan trọng của câu lệnh *while...do*. Trong Pascal, để mô tả cấu trúc lặp với số lần chưa biết trước, còn có một câu lệnh khác là *Repeat...until*. Với câu lệnh *Repeat...until* có đặc điểm là các câu lệnh trong vòng lặp luôn được thực hiện ít nhất một lần. SGK chỉ giới thiệu câu lệnh *while...do*. Người ta đã chứng minh được rằng chỉ cần câu lệnh *while...do* là đủ, có nghĩa là mọi tình huống lặp sử dụng câu lệnh *repeat...until* đều có thể sử dụng *while...do* để thay thế. GV không giới thiệu câu lệnh *Repeat...until*, không so sánh câu lệnh *Repeat...until* với *while...do* để tránh quá tải với HS.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. Có thể nêu rất nhiều ví dụ về các hoạt động lặp với số lần chưa biết trước. Dưới đây là một số ví dụ:

- a) Tìm một từ nhất định bị gõ sai chính tả trong văn bản và sửa lại cho đúng. Số từ cần phải sửa chưa được biết trước.
- b) Khi chuẩn bị bát phở để phục vụ cho khách, cô bán hàng thường thực hiện các công việc sau đây: Cho một lượng bánh phở vào nồi nước phở để trần bánh phở, cho bánh phở đã trần vào bát, làm chín một ít thịt và cho vào bát bánh phở đã được làm nóng, cho thêm gia vị, chan nước phở đang được đun sôi vào bát phở,... Các thao tác đó được thực hiện lặp lại mỗi khi có khách ăn phở. Trong suốt ca bán hàng số lần thực hiện các thao tác lặp đó là không thể biết trước.
- c) Trong xưởng may, mỗi cô công nhân được giao may một chi tiết của chiếc áo, hay chiếc quần với các đường may đã được thiết kế trước. May xong một sản phẩm, cô công nhân sẽ may sản phẩm tiếp theo cho đến khi hết giờ làm việc.

Bài 2. Sự khác biệt giữa câu lệnh lặp với số lần biết trước và câu lệnh với số lần chưa biết trước là:

- a) Như tên gọi của nó, câu lệnh lặp với số lần biết trước chỉ thị cho máy tính thực hiện một lệnh hoặc một nhóm lệnh với số lần đã được xác định từ trước, còn với câu lệnh lặp với số lần chưa biết trước thì số lần chưa được xác định trước.

- b) Trong câu lệnh lặp với số lần biết trước, điều kiện là giá trị của biến đếm có giá trị nguyên đã đạt được giá trị lớn nhất hay chưa, còn trong câu lệnh lặp với số lần chưa biết trước, điều kiện tổng quát hơn, có thể là kiểm tra một giá trị của một số thực, cũng có thể là một điều kiện khác, ví dụ như một số có chia hết cho 3 hay không,...
- c) Trong câu lệnh lặp với số lần biết trước, *câu lệnh* được thực hiện *ít nhất một lần*, sau đó kiểm tra điều kiện. Trong câu lệnh lặp với số lần chưa xác định trước, trước hết điều kiện được kiểm tra. Nếu điều kiện được thoả mãn, *câu lệnh* mới được thực hiện. Do đó có thể có trường hợp *câu lệnh* không được thực hiện một lần nào.

Bài 3. a) Thuật toán 1: Chín vòng lặp được thực hiện. Khi kết thúc thuật toán $S = 5.0$. Đoạn chương trình Pascal tương ứng:

```
S:=10; x:=0.5;
while S>5.2 do S:=S-x;
writeln(S);
```

b) Thuật toán 2: Không vòng lặp nào được thực hiện vì ngay từ đầu điều kiện đã không thoả mãn nên các bước 2 và 3 bị bỏ qua. $S = 10$ khi kết thúc thuật toán. Đoạn chương trình Pascal tương ứng:

```
S:=10; n:=0;
while S<10 do
begin n:=n+3; S:=S-n end;
writeln(S);
```

Nhận xét: Trong các thuật toán và chương trình trên, điều kiện được kiểm tra trước khi các lệnh trong vòng lặp được thực hiện. Do đó nếu điều kiện không được thoả mãn ngay từ đầu, các lệnh trong vòng lặp sẽ bị bỏ qua.

Bài 4. a) Chương trình thực hiện 5 vòng lặp.

b) Vòng lặp trong chương trình được thực hiện vô tận vì sau câu lệnh $n := n + 1$ câu lệnh lặp kết thúc và S vẫn mang giá trị 0 nên điều kiện $S \leq 10$ luôn được thoả mãn.

Nhận xét: Trong câu lệnh lặp, điều kiện cần phải được thay đổi để sớm hay muộn chuyển sang trạng thái không thoả mãn. Khi đó vòng lặp mới được kết thúc sau hữu hạn bước. Để làm được điều này, *câu lệnh* trong câu lệnh lặp *while...do* thường là *câu lệnh ghép*.



Bài thực hành 6

SỬ DỤNG LỆNH LẶP WHILE...DO

Thời lượng: 2 tiết

1. Mục đích, yêu cầu

- Hiểu câu lệnh lặp *while...do* trong chương trình;
- Biết lựa chọn câu lệnh lặp *while...do* hoặc *for...do* phù hợp với tình huống cụ thể;
- Rèn luyện kỹ năng về khai báo, sử dụng biến;
- Rèn luyện khả năng đọc chương trình;
- Biết vai trò của việc kết hợp các cấu trúc điều khiển.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Trước hết cần lưu ý rằng chuẩn kiến thức, kỹ năng không yêu cầu HS phải viết được chương trình có sử dụng câu lệnh lặp với số lần chưa biết trước. Do vậy, trong bài thực hành này không yêu cầu HS phải tự viết chương trình có câu lệnh *while...do* để giải bài toán. GV có thể cho HS đọc, hiểu, thử chạy chương trình trên máy và đặc biệt cần hiểu được hoạt động của lệnh *while...do* trong chương trình.

b) Với bài 1, trước hết cần xác định Input và Output của bài toán:

Input: Dãy số thực x_1, x_2, \dots, x_n ;

Output: Giá trị trung bình $(x_1 + x_2 + \dots + x_n)/n$.

Thuật toán

Bước 1. Nhập n là số lượng các số thực sẽ được nhập từ bàn phím:

1.1. $Dem \leftarrow 0$;

1.2. $Sum \leftarrow 0$.

Bước 2. Trong khi $Dem < N$ thì

2.1. $Dem \leftarrow Dem + 1$;

2.2. Nhập giá trị số thực x từ bàn phím;

2.3. $Sum \leftarrow Sum + x$;

Bước 3. $TB \leftarrow Sum/N$.

Bước 4. Đưa giá trị TB ra màn hình, rồi kết thúc.

Có nhiều cách để mô tả thuật toán này, tuy nhiên cách mô tả trên đây được sử dụng với mục đích HS thuận lợi hơn khi đọc, hiểu, đối chiếu giữa thuật toán với chương trình *Tinh_Trung_bình* ở câu b.

Căn cứ vào mô tả thuật toán, HS tìm hiểu để xác định các biến và kiểu dữ liệu tương ứng cần khai báo trong chương trình.

Câu c yêu cầu HS dịch, chỉnh sửa, chạy và kiểm thử chương trình. Những kỹ năng này HS đã được rèn luyện ở những bài thực hành trước cho nên HS có thể hoàn toàn thực hiện được. GV có thể đưa ra hoặc hướng dẫn HS tạo ra những bộ dữ liệu test.

GV cần yêu cầu HS đọc, thảo luận đối chiếu giữa thuật toán và các câu lệnh mô tả thuật toán trong chương trình. Cần làm cho HS hiểu rõ về hoạt động của lệnh lặp *while...do* trong chương trình, có thể cho HS làm việc nhóm để mô phỏng chương trình (việc này nên được làm trên lớp học, ở tiết bài tập trước khi thực hành trên máy).

Ví dụ dưới đây là một mô phỏng hoạt động chính của chương trình với $n = 3$:

1. Trước khi bắt đầu vòng lặp *while...do*: $dem = 0$, $TB = 0$, $n = 3$;

2. Bắt đầu lệnh lặp *while...do*

Giá trị DEM trước khi thực hiện vòng lặp	DEM < n	Giá trị DEM sau khi thực hiện vòng lặp	X (nhập từ bàn phím)	TB
0	Đúng	1	10	10
1	Đúng	2	15	25
2	Đúng	3	20	45
3	Sai			

3. Kết thúc lệnh lặp *while...do*: $TB = 45/3 = 15$.

Câu d yêu cầu HS chuyển từ sử dụng câu lệnh *while...do* sang sử dụng câu lệnh *for...do*. Qua việc làm này HS được rèn luyện thêm về sử dụng lệnh *for...do*.

Tuy nhiên, về cơ bản tình huống sử dụng *while...do* và *for...do* là khác nhau. *While...do* thích hợp hơn với trường hợp lặp với số lần chưa biết trước, *for...do* thích hợp hơn với trường hợp lặp với số lần biết trước.

Như vậy, ở câu d, bên cạnh mục đích cho HS có ý thức trong việc lựa chọn cấu trúc lặp phù hợp với tình huống, còn có mục đích tiếp tục rèn luyện viết chương trình với câu lệnh *for...do* đảm bảo đạt yêu cầu đề ra trong chuẩn kiến thức, kỹ năng.

c) Với bài 2, cách tiến hành giống như với bài 1. Trước hết cần xác định Input và Output của bài toán:

Input: Số tự nhiên N ;

Output: Trả lời N là số nguyên tố hoặc N không là số nguyên tố.

Thuật toán

HS lớp 8 đã biết tính chất của số nguyên tố là số tự nhiên chỉ chia hết cho 1 và chính nó.

Để kiểm tra N có phải số nguyên tố hay không ta kiểm tra xem N có chia hết cho các số từ 2 đến $N - 1$ hay không. Nếu N không chia hết cho số nào trong khoảng từ 2 đến $N - 1$ thì N là số nguyên tố, ngược lại nếu N chia hết cho bất kì một số nào trong khoảng từ 2 đến $N - 1$ thì N không phải là số nguyên tố.

Sử dụng phép chia lấy phần dư *mod* để kiểm tra tính chia hết.

Bước 1. Nhập số tự nhiên N từ bàn phím.

Bước 2. Nếu $N \leq 1$ thông báo N không phải là số nguyên tố, rồi chuyển đến bước 4.

Bước 3. Nếu $N > 1$:

3.1. $i \leftarrow 2$;

3.2. Trong khi $N \bmod i \neq 0$ còn đúng thì $i \leftarrow i + 1$;

3.3. Nếu $i = N$ thì thông báo N là số nguyên tố, rồi chuyển đến bước 4.

Ngược lại, thông báo N không phải là số nguyên tố;

Bước 4. Kết thúc.

Sau đó GV cho HS đọc chương trình trong SGK, đối chiếu việc sử dụng câu lệnh để mô tả thuật toán trên đây.

Lưu ý: Trong chương trình sử dụng cả câu lệnh điều kiện, câu lệnh lặp *while...do*. Mục đích của bài này là để HS thấy được sự cần thiết phải kết hợp

các cấu trúc điều khiển để giải quyết bài toán. Hơn nữa, trong ví dụ này còn sử dụng phép chia lấy phần dư *mod*. Điều này thể hiện sự cần thiết và tính hiệu quả khi lựa chọn công cụ phù hợp trong lập trình.

d) Đối với đa số HS lớp 8, thuật toán kiểm tra tính nguyên tố của một số tự nhiên là không khó. Tuy nhiên, nếu thấy HS của mình có thể gặp khó khăn khi tìm hiểu thuật toán này, GV có thể lấy một số giá trị n cụ thể để "chạy" mô phỏng hoạt động của chương trình hoặc thay thế bằng ví dụ khác. Ví dụ mà GV đưa ra có thể chỉ cần thể hiện sự kết hợp giữa câu lệnh điều kiện và câu lệnh lặp với số lần chưa biết trước, không nhất thiết phải có tình huống sử dụng phép chia lấy phần dư *mod*.

Sự kết hợp các cấu trúc điều khiển (tuần tự, rẽ nhánh và lặp) trong ngôn ngữ lập trình tạo nên sự linh hoạt và góp phần tạo nên sức mạnh của ngôn ngữ lập trình. Chính sự kết hợp giữa các cấu trúc điều khiển cho phép ngôn ngữ lập trình mô tả được những thuật toán phức tạp, giúp giải quyết được nhiều bài toán xuất phát từ nhu cầu thực tiễn. GV có thể không cần trình bày ý nghĩa của việc kết hợp các cấu trúc điều khiển với HS.

Như trên đã nêu chuẩn kiến thức, kỹ năng không yêu cầu HS phải viết được chương trình có sử dụng câu lệnh lặp với số lần biết trước. Tuy nhiên, nếu HS tiếp thu tốt, GV có thể yêu cầu HS tập viết chương trình đơn giản có sử dụng câu lệnh *while...do*.

Bài 9. LÀM VIỆC VỚI DÃY SỐ

Thời lượng: 3 tiết

1. Mục đích, yêu cầu

- Biết được khái niệm mảng một chiều;
- Biết cách khai báo mảng, nhập, in, truy cập các phần tử của mảng;
- Hiểu thuật toán tìm số lớn nhất, số nhỏ nhất của một dãy số.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Tương tự với câu lệnh điều kiện, câu lệnh lặp, vào đầu bài này cần giới thiệu một số ví dụ nhằm đưa đến nhu cầu cần có biến mảng trong ngôn ngữ lập trình.

Ví dụ ở đầu bài trong SGK dẫn đến nhu cầu biến mảng, sau khi đã phân tích sự bất tiện nếu chỉ sử dụng cách khai báo biến đã biết (khai báo biến đơn). Pascal cung cấp một công cụ hiệu quả để hỗ trợ người lập trình đó là biến mảng.

Mục 1 chỉ nêu nhu cầu của biến mảng trong ngôn ngữ lập trình và mục tiêu của các ví dụ ở mục 1 là dẫn đến nhu cầu cần có biến mảng.

Trong mục 2, cần cho HS biết cách khai báo, truy cập, nhập (gán) giá trị, viết giá trị của biến mảng ra màn hình.

b) Khai báo biến mảng

Có hai cách khai báo biến mảng

Cách 1: Khai báo trực tiếp biến mảng một chiều:

var <tên biến mảng>: **array** [<chỉ số đầu>..<<chỉ số cuối>] **of** <kiểu dữ liệu>;

Cách 2: Khai báo gián tiếp biến mảng qua kiểu mảng một chiều:

type <tên kiểu mảng> = **array** [<chỉ số đầu>..<<chỉ số cuối>] **of** <kiểu dữ liệu>;

var <tên biến mảng>: <tên kiểu mảng>;

trong đó:

– <chỉ số đầu>..<<chỉ số cuối> là một dãy số nguyên liên tục $n_1...n_2$ với n_1, n_2 là các hằng (hoặc biểu thức cho kết quả là số nguyên) ($n_1 \leq n_2$).

– <kiểu dữ liệu> là kiểu dữ liệu của các phần tử trong mảng.

Tuy nhiên, theo yêu cầu giảm tải, GV chỉ cần giới thiệu với HS cách 1. Kiểu chỉ số cũng chỉ cần giới thiệu thật đơn giản là dãy số nguyên dương (không nhất thiết phải giới thiệu những trường hợp còn lại). Kiểu phần tử chỉ hạn chế là số nguyên, số thực.

GV cần sử dụng một số ví dụ để luyện tập về khai báo mảng một chiều và giải thích số lượng phần tử, kiểu phần tử của từng biến mảng tương ứng với mỗi ví dụ.

c) Truy cập các phần tử của mảng

Sau khi đã cho HS luyện tập với khai báo biến mảng, có thể sử dụng các khai báo vừa thực hiện để giới thiệu về các truy cập vào biến mảng. Ví dụ, khai báo

```
var Diem: array[1..50] of real;
```

đã tạo ra một biến mảng có 50 phần tử được đánh số từ 1 đến 50. Các phần tử này được "đặt tên" như thế nào? Để "gọi đích danh" từng phần tử cụ thể Pascal sử dụng cách: *Tên biến mảng[chỉ số phần tử]*. Ví dụ, *Diem[1]* là phần tử thứ nhất; *Diem[5]* là phần tử thứ năm. Có thể thực hiện các thao tác như gán giá trị, so sánh, viết giá trị ra màn hình... với *Diem[1]*, *Diem[2]*, ..., *Diem[50]* như với biến đã học (biến đơn).

d) Nhập giá trị cho biến mảng

Để nhập giá trị cho biến mảng thì cần nhập giá trị cho từng phần tử của mảng. Giống như với việc gán giá trị cho biến đơn, có hai cách để gán giá trị cho phần tử của mảng:

Gán trực tiếp bằng lệnh gán: ví dụ: *Diem[1] := 8*, *Diem[2] := 9.5*.

Gán giá trị bằng cách nhập từ bàn phím, sử dụng lệnh *read*, *readln*.

Có thể viết một đoạn chương trình với 50 lệnh *readln* để thực hiện việc nhập giá trị cho 50 phần tử của mảng từ bàn phím:

```
readln(Diem[1]); readln(Diem[2]); ...; readln(Diem[50]);
```

Tuy nhiên, việc kết hợp lệnh lặp *for...do* với câu lệnh *readln* là một cách lập trình hiệu quả, thường được sử dụng để nhập dữ liệu cho mảng.

```
For i:=1 to 50 do readln(Diem[i]);
```

Tương tự như vậy, để viết giá trị của các phần tử của mảng ra màn hình người ta kết hợp giữa *for...do* với lệnh *writeln* hoặc *write*.

```
for i:=1 to 50 do writeln(Diem[i]);
```

Giả sử chỉ muốn viết ra màn hình những điểm số lớn hơn hoặc bằng 9 chẳng hạn, câu lệnh có thể được viết như sau:

```
For i:=1 to 50 do
```



```
if Diem[i] >= 9 then writeln(Diem[i]);
```

Việc đưa ra yêu cầu này có hai mục đích: thứ nhất là để HS làm quen trước với so sánh phần tử của biến mảng sẽ được sử dụng trong phần sau. Làm như vậy HS sẽ không bị ngỡ ngàng khi gặp phép so sánh này trong chương trình. Thứ hai, HS thấy được sự kết hợp giữa các câu lệnh mà cụ thể là câu lệnh *for...do* và câu lệnh *if-then* trong chương trình.

Trong các ví dụ trên, khi duyệt các phần tử của biến mảng hoàn toàn có thể sử dụng cấu trúc *while...do*, tuy nhiên ở đây cấu trúc *for...do* phù hợp hơn vì biết trước số lần lặp. Mặt khác, sử dụng cấu trúc *for...do* ở đây nói chung là dễ hiểu hơn, gần với cách nghĩ tự nhiên của HS hơn. GV cũng có thể nhắc lại về tầm quan trọng của việc lựa chọn cấu trúc điều khiển phù hợp khi lập trình.

e) Mục 3 là ví dụ về một chương trình cụ thể sử dụng biến mảng và thuật toán tìm giá trị lớn nhất, nhỏ nhất của dãy số nguyên. Trước khi giới thiệu chương trình cụ thể GV nên hướng dẫn HS tìm hiểu lại thuật toán này (đã học trong bài 5) và yêu cầu HS thảo luận, chỉnh sửa thuật toán trên để tìm ra số nhỏ nhất của dãy số. GV có thể yêu cầu HS thực hiện lại việc mô phỏng thuật toán trên một dãy số cụ thể để các em nhớ lại thuật toán. Ví dụ về mô phỏng thuật toán này đã có ở bài 5.

Giải thích về thuật toán trong chương trình tìm giá trị lớn nhất và nhỏ nhất của dãy số nguyên:

- + Đầu tiên gán giá trị số thứ nhất của dãy số cho *Max*, *Min* (ban đầu tạm thời coi số thứ nhất đồng thời là số lớn nhất và nhỏ nhất tạm thời).
- + So sánh số lớn nhất/nhỏ nhất tạm thời này với số thứ hai, nếu số thứ hai lớn hơn số lớn nhất tạm thời *Max* thì gán giá trị của số thứ hai cho *Max*; nếu số thứ hai nhỏ hơn số nhỏ nhất tạm thời *Min* thì gán giá trị của số thứ hai cho *Min*. Như vậy đến thời điểm này *Max* và *Min* là số lớn nhất và số nhỏ nhất của số thứ nhất và số thứ hai.
- + Cứ tiếp tục như vậy, đem so sánh *Max/Min* với tất cả các số còn lại, gặp số nào lớn hơn *Max* thì gán giá trị của số đó cho *Max*; gặp số nào nhỏ hơn *Min* thì gán giá trị của số đó cho *Min*. Sau khi so sánh đến số cuối cùng của dãy số thì *Max* và *Min* tương ứng chính là giá trị lớn nhất và nhỏ nhất của dãy số.

Ví dụ, mô phỏng thuật toán tìm giá trị nhỏ nhất của dãy số có thể như bảng dưới đây:

Dãy số	5	4	4	7	6	3	15	6	8
i	1	2	3	4	5	6	7	8	9
$a_i > \text{MAX}$		Sai	Sai	Đúng	Sai	Sai	Đúng	Sai	Sai
MAX	5	5	5	7	7	7	15	15	15
$a_i < \text{MIN}$		Đúng	Sai	Sai	Sai	Đúng	Sai	Sai	Sai
MIN	5	4	4	4	4	3	3	3	3

f) Để HS hiểu được máy tính làm việc như thế nào, GV có thể yêu cầu các em thực hiện như sau: Viết 10 số nguyên, mỗi số vào một mảnh giấy. Gấp 10 mảnh giấy này lại và bỏ vào một hộp A. Đặt một hộp B rỗng bên cạnh. Yêu cầu HS chuyển lần lượt đến hết từng mảnh giấy ở hộp A sang hộp B. Sau khi chuyển xong HS cho biết số lớn nhất trong các số được ghi trên các mảnh giấy. Hai bạn HS được phép xem số trên mảnh giấy khi chuyển mảnh giấy đó từ hộp A sang hộp B nhưng không ghi chép ra giấy. Mục đích của việc không cho HS ghi chép là để HS mô phỏng hoạt động của máy tính: tại thời điểm hiện tại bạn thứ nhất chỉ cần nhớ số lớn nhất, bạn thứ hai chỉ cần nhớ số nhỏ nhất, so sánh với số vừa lấy ra từ hộp A và nhớ lấy số lớn hơn hoặc nhỏ hơn theo phân công. Cứ tiếp tục như vậy đến khi hết các số trong hộp, số được mỗi bạn nhớ cuối cùng tương ứng sẽ là số lớn nhất và nhỏ nhất.

Để thực hiện công việc này cần hướng dẫn HS thực hiện mô phỏng theo giải thuật tìm dãy số lớn nhất của dãy số nguyên. Nhặt mảnh giấy đầu tiên ở hộp A, mở ra nhớ giá trị của mảnh giấy này (coi là số lớn nhất và nhỏ nhất tạm thời), gấp lại và bỏ vào hộp B. Nhặt mảnh giấy thứ hai, mở ra và mỗi bạn so sánh với số mình đang nhớ. Bạn thứ nhất nhớ giá trị lớn nhất mới nếu thấy số ghi trong mảnh giấy được lấy ra lớn hơn; bạn thứ hai nhớ giá trị nhỏ nhất mới nếu thấy số đó nhỏ hơn số mà bạn ghi nhớ. Lặp lại công việc này đến khi hết các mảnh giấy của hộp A.

GV có thể thêm, bớt các mảnh giấy để HS làm lại. Sau khi HS làm đề nghị các em mô tả lại cách các em đã thực hiện để tìm ra số lớn nhất/nhỏ nhất. Cách mà HS làm giống với cách máy tính thực hiện thuật toán ở trên. Máy

tính chỉ có thể tham chiếu đến từng số trong dãy số, máy tính không có khả năng quan sát cả dãy số vì vậy máy tính phải thực hiện theo thuật toán như trên. Máy tính thực hiện tuần tự theo đúng chỉ dẫn của con người.

Có thể cải tiến nội dung dạy học trên đây thành nhiều trò chơi khác nhau. Ví dụ, yêu cầu các em không sử dụng giấy, bút, GV lần lượt viết từng số lên bảng, rồi xoá đi luôn, HS quan sát để tìm ra số lớn nhất (hoặc nhỏ nhất). Hoặc mời một nhóm HS đứng lên phía trên lớp. Mời một em đi qua từng bạn một, khi em này đến bên bạn nào đó thì bạn này phải đưa ra một số nào đó (có thể là nói thầm hoặc viết ra một mảnh giấy). Đi hết lượt HS phải nói được bạn nào đã đưa ra số lớn nhất (hoặc nhỏ nhất). Kết quả này được kiểm chứng công khai bởi các bạn đã đưa ra các số.

Lưu ý: GV cũng có thể bắt đầu bằng chương trình tìm riêng giá trị *Max*.

Thuật toán tìm *Max* của dãy số nguyên nhập từ bàn phím như sau (Bài 5):

Bước 1. Nhập *N* và dãy A_1, \dots, A_n ;

Bước 2. $Max \leftarrow A_1$;

Bước 3. Với *i* từ 2 đến *N* thực hiện: nếu $Max < A_i$ thì $Max \leftarrow A_i$;

Bước 4. Đưa ra màn hình giá trị *Max* rồi kết thúc.

Sau khi giới thiệu xong thuật toán tìm *Max*, GV hướng dẫn HS xác định các biến, kiểu biến và viết khai báo biến; viết câu lệnh thực hiện các bước nhập *N*, nhập các phần tử của mảng, tìm *Max*, in *Max* ra màn hình. Chương trình có thể được xây dựng dần từng phần và cuối cùng có được một chương trình như dưới đây.

```
program P_Max;
Var i, N, Max: integer;
  A: array[1..100] of integer;
Begin
  {Nhập N}
  write('Hay nhập do dai cua day so, N = '); readln(N);
  {Nhập day so}
  writeln('Nhập cac phan tu cua day so:');
  For i:=1 to N do
    Begin
      write('a[' , i, ']='); readln(a[i]);
    End;
  {Tìm Max}
  Max:=a[1];
```

```

for i:=2 to n do if Max<a[i] then Max:=a[i];
{Hien thi Max ra man hinh}
write('So lon nhat la Max = ',Max);
readln
End.

```

Yêu cầu HS chỉnh sửa chương trình trên để tìm giá trị nhỏ nhất của dãy số, tính tổng dãy số. Sau khi đã hiểu rõ thuật toán và chương trình tìm *Max*, *Min* có thể yêu cầu HS kết hợp tìm *Max*, *Min* trong cùng một chương trình như trong SGK.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. “Có thể xem biến mảng là một biến được tạo từ nhiều biến có cùng kiểu, nhưng chỉ có một tên duy nhất”. Phát biểu này SAI. Vì từ ngay trong nội dung mệnh đề trên, đoạn *"nhiều biến có cùng kiểu, nhưng chỉ có một tên duy nhất"* là không chính xác. Trong Pascal mỗi biến đều phải có một tên khác nhau, nên không thể có nhiều biến với cùng một tên duy nhất.

Bài 2. Lợi ích chính của việc sử dụng biến mảng là rút gọn việc viết chương trình, có thể sử dụng câu lệnh lặp để thay nhiều câu lệnh. Ngoài ra chúng ta còn có thể lưu trữ và xử lý nhiều dữ liệu có nội dung liên quan đến nhau một cách hiệu quả.

Bài 3. a) Sai. Phải thay dấu phẩy bằng hai dấu chấm;
b) và c) Sai, vì chỉ số mảng phải là số nguyên;
d) Đúng.

Bài 4. Không. Giá trị nhỏ nhất và lớn nhất của chỉ số mảng phải được xác định trong phần khai báo chương trình.

Bài 5. Chương trình có thể như sau:

```

var N, i: integer;
    A: array[1..100] of real;
begin
write('Nhap so phan tu cua mang, n= '); readln(n);
for i:=1 to n do
begin
write('Nhap gia tri ',i,'cua mang, a[',i,']= ');
readln(a[i]);
end;
readln
end.

```



Bài thực hành 7

XỬ LÝ DÃY SỐ TRONG CHƯƠNG TRÌNH

Thời lượng: 3 tiết

1. Mục đích, yêu cầu

- Thực hành khai báo và sử dụng các biến mảng;
- Ôn luyện cách sử dụng các câu lệnh *if...then, for...do*;
- củng cố kỹ năng đọc, hiểu và chỉnh sửa chương trình;
- Hiểu và viết được chương trình với thuật toán tìm giá trị lớn nhất, nhỏ nhất của một dãy số, tính tổng dãy số.

2. Những điểm cần lưu ý và gợi ý dạy học

a) Để gây hứng thú cho HS, cần dành thời gian để HS gõ, chạy thử chương trình tìm *Max*, *Min*, tính tổng dãy số. Hơn thế nữa, việc thực hiện các bài học là cần thiết do yêu cầu trong chương trình của môn học. HS cần viết được các chương trình này. GV có thể yêu cầu HS sửa chương trình *Maxmin* trong SGK thành chương trình tìm *Max* (tham khảo chương trình *P_Max* trong bài học trước) sau đó tự viết được chương trình *P_Min* và chương trình tính tổng *P_Sum*. Chương trình tìm *Max* đã được giới thiệu ở bài trên.

Chương trình tìm giá trị nhỏ nhất trong dãy số nguyên *P_Min*:

```
Program P_Min;
Var i, n, Min: integer;
    A: array[1..100] of integer;
Begin
    write('Hay nhap do dai cua day so, n = '); readln(n);
    writeln('Nhap cac phan tu cua day so:');
    For i:=1 to n do
        Begin
            write('a[' , i, ']='); readln(a[i]);
        End;
    Min:=a[1];
    for i:=2 to n do if Min>a[i] then Min:=a[i];
    write('So nho nhat la Min = ',Min);
    readln
End.
```

Trong chương trình tính tổng dãy số dưới đây có thêm câu lệnh in ra màn hình dãy số vừa nhập để người dùng có thể thuận tiện kiểm chứng kết quả chương trình. Nhưng đây cũng nhằm mục đích luyện tập với việc in phần tử của mảng ra màn hình.

```

Program P_Sum;
Var    i, n, Sum: integer;
        A: array[1..100] of integer;
Begin
    write('Hay nhap do dai cua day so, n = '); readln(n);
    writeln('Nhap cac phan tu cua day so:');
    For i:=1 to n do
        Begin
            write('a[' , i, ']='); readln(a[i]);
        End;
    Sum:=0;
    for i:=1 to n do Sum:= Sum + a[i];
    write('Day so vua nhap la: ');
    for i:=1 to n do write(a[i], ' ');
    writeln;
    write('Tong day so la = ', Sum);
    readln
End.

```

b) Sau khi HS đã hiểu rõ các chương trình P_Max, P_Min và P_Sum thì không quá khó khăn để thực hiện yêu cầu trong bài thực hành 7. Do vậy, thời gian còn lại dành để HS thực hành với các bài sử dụng kết hợp nhiều câu lệnh, biểu thức điều kiện như trong SGK.

Lưu ý: Theo yêu cầu của chuẩn kiến thức, kỹ năng, kết thúc bài này phải đảm bảo HS hiểu được thuật toán, tự viết được chương trình tìm số lớn nhất, nhỏ nhất của dãy số, kiểm tra điều kiện ba số a, b, c có phải là độ dài của ba cạnh của một tam giác hay không (hoặc các bài toán tương đương). HS cần tự viết được chương trình nhập giá trị phần tử mảng, in ra màn hình các phần tử của mảng, tính tổng các phần tử của mảng. Do vậy, trong trường hợp cần thiết GV cần lựa chọn, xây dựng nội dung tiết bài tập, ôn tập trên lớp hoặc thực hành trên phòng máy để đảm bảo đạt được yêu cầu quy định trong chuẩn kiến thức, kỹ năng.

CHƯƠNG II. PHẦN MỀM HỌC TẬP

I. GIỚI THIỆU

1. Mục tiêu

a) Kiến thức

- HS hiểu và biết cách sử dụng các phần mềm học tập đã trình bày trong SGK.
- Thông qua các phần mềm HS hiểu được ý nghĩa của các phần mềm máy tính ứng dụng trong các lĩnh vực khác nhau của cuộc sống (ví dụ học Toán, Sinh học).
- Thông qua phần mềm HS hiểu biết thêm và có ý thức trong việc sử dụng máy tính đúng mục đích.

b) Kỹ năng

- HS có kỹ năng sử dụng và khai thác thành thạo các phần mềm học tập đã được giới thiệu.

c) Thái độ

- HS cần có thái độ nghiêm túc khi học và làm việc trên máy tính không phân biệt phần mềm học tập hay phần mềm trò chơi.
- HS có ý thức và khả năng liên hệ từ phần mềm đến thực tế để sử dụng phần mềm vào giải quyết các bài toán, vấn đề đã được học trên lớp, từ đó nâng cao ý thức và lòng say mê học tập các môn học trên lớp của mình.

2. Nội dung chủ yếu

Chương này bao gồm 3 bài học sau đây:

Làm quen với giải phẫu cơ thể người bằng phần mềm Anatomy (4 tiết).

Anatomy là phần mềm hỗ trợ mô phỏng giúp HS quan sát và học tập phần kiến thức giải phẫu người của môn Sinh học. Phần kiến thức này được học

trong chương trình Sinh học lớp 8, do đó phần mềm là một bổ sung rất tốt cho HS hiểu hơn phần kiến thức này.

Giải toán và vẽ hình phẳng với GeoGebra (4 tiết).

GeoGebra là phần mềm cho phép vẽ các đối tượng hình học như điểm, đoạn, đường và đường tròn trên mặt phẳng. Chương trình học với GeoGebra là sự phát triển tiếp theo nội dung đã giới thiệu trong cuốn Tin học dành cho THCS- Quyển 2. Điểm đặc biệt nhất của phần mềm này là sự liên kết chặt chẽ giữa các đối tượng hình học trên. Chính sự liên kết toán học giữa các đối tượng hình học tạo ra khái niệm hình học động (hay toán học động) là một trong những khái niệm được dùng nhiều trong các phần mềm mô phỏng giáo dục hiện nay.

Vẽ hình không gian với GeoGebra (4 tiết).

Trong bài học này HS sẽ bước đầu làm quen với các công cụ làm việc với các đối tượng hình học không gian 3D của Geogebra. Mục đích chính của bài học là HS sẽ làm quen và vẽ được một số hình không gian cơ bản có trong phần Hình học của SGK Toán lớp 8. Đó là hình chóp, hình lăng trụ và hình hộp chữ nhật.

3. Những điểm cần lưu ý trong giảng dạy

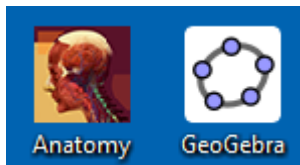
- a)** Do cấu trúc của SGK yêu cầu, các tác giả đã viết riêng một phần dành cho các phần mềm hỗ trợ học tập. Trên thực tế GV cần rất chủ động, không nên dạy phần này dồn vào một thời gian theo đúng thứ tự như trong SGK. Các bài học của phần này có thể dạy xen kẽ trong suốt quá trình học tập của HS.
- b)** Việc học và thực hành theo phần này có thể được tiến hành theo các cách sau:

Cách 1: Dạy bình thường theo đúng trình tự như trong SGK. Không khuyến khích dùng cách này.

Cách 2: Các bài học trong phần này được dạy xen kẽ với các bài học khác. Thứ tự và vị trí chèn bài học do các GV chủ động quyết định. Nên thực hiện theo cách này.

Cách 3: Các bài học trong phần này có thể dạy xen kẽ lí thuyết và thực hành với các bài học của các phần khác. Ví dụ bài học về phần mềm GeoGebra sẽ được dạy ngay từ những bài học đầu tiên của chương trình Tin học, còn phần thực hành của phần mềm này sẽ được tiến hành trải đều trong suốt quá trình học tập. Cách này là tốt nhất rất nên thực hiện.

- c) GV cần cài đặt tất cả các phần mềm trước khi HS thực hành. Mỗi phần mềm, GV cần tạo ra một biểu tượng tắt (shortcut) trên màn hình nền để HS dễ dàng nhận biết.



Biểu tượng của các phần mềm học tập

- d) Tất cả các phần mềm đều được cài đặt bình thường trên hệ điều hành Windows.

II. HƯỚNG DẪN CHI TIẾT

Bài 10. LÀM QUEN VỚI GIẢI PHẪU CƠ THỂ NGƯỜI BẰNG PHẦN MỀM ANATOMY


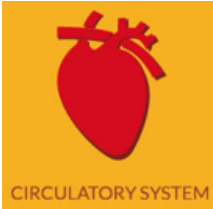






Thời lượng: 4 tiết

1. Mục đích, yêu cầu

- HS hiểu mục đích và ý nghĩa của phần mềm và có thể tự khởi động, tự mở các bài học chức năng và luyện tập liên quan đến giải phẫu cơ thể người của phần mềm;
- Thông qua phần mềm, HS biết và có thể tra cứu hình ảnh, thông tin và nhiều kiến thức khác hỗ trợ cho việc học môn Sinh học 8.

2. Những điểm cần lưu ý và gợi ý dạy học

- a) Phần mềm Anatomy có 8 chức năng chính.

 RESPIRATORY SYSTEM	 CIRCULATORY SYSTEM	 DIGESTIVE SYSTEM	 EXCRETOR SYSTEM
Hệ hô hấp	Hệ tuần hoàn	Hệ tiêu hoá	Hệ bài tiết
 NERVOUS SYSTEM	 SKELETAL SYSTEM	 MUSCULAR SYSTEM	 REPRODUCTIVE SYSTEM
Hệ thần kinh	Hệ xương	Hệ cơ	Hệ sinh sản

Việc phân bổ kế hoạch giảng dạy có thể như sau:

Tiết 1: Giới thiệu chung phần mềm. Hệ **xương**, hệ **cơ**.

Tiết 2. Giới thiệu hệ **tuần hoàn**.

Tiết 3. Giới thiệu hệ **hô hấp**, hệ **tiêu hoá**.

Tiết 4. Giới thiệu hệ **thần kinh**, hệ **bài tiết**.

Chú ý các hệ giải phẫu được in đậm ở trên là có phần mô phỏng bằng video hoạt hình rất hữu ích.

b) Với mỗi hệ giải phẫu, GV có thể cùng học sinh quan sát và thực hành các thao tác sau:

1. Xem tổng thể toàn bộ hệ thống của hệ giải phẫu. Có thể phóng to, thu nhỏ, xoay theo các chiều, hướng khác nhau để quan sát kĩ.
2. Xem từng bộ phận độc lập, riêng biệt. Có thể bóc tách các lớp để xem các bộ phận bên trong.
3. Xem từng bộ phận có kết hợp hiển thị các hệ giải phẫu khác. Ví dụ muốn xem một bộ phận của hệ xương có thể hiện đồng thời hệ cơ để quan sát sự liên quan giữa hai hệ giải phẫu này. Tính năng này rất đặc biệt.
4. Với mỗi bộ phận cấu thành có thể xem thông tin: tên của bộ phận, mô tả chính chức năng chi tiết của bộ phận và xem các thông tin bổ sung.

5. Xem video mô phỏng hoạt động của hệ giải phẫu cơ thể người. Đây là tính năng đặc biệt nhất của phần mềm và chỉ có 5 hệ giải phẫu có tính năng này là hệ **tuần hoàn, hô hấp, thần kinh, tiêu hoá và bài tiết**.

Chú ý: Với mỗi hệ giải phẫu, GV cần giới thiệu cả năm tính năng trên, trong đó đặc biệt nhấn mạnh các tính năng 1, 2, 3 và 5.

d) Trước khi có thể giảng dạy tốt bài học này, GV nên dành thời gian xem lại chương trình môn Sinh học, giải phẫu cơ thể người của SGK Sinh học 8.

e) Với hệ **cơ – hệ xương**, cần chú ý các yếu tố sau:

Hai hệ cơ – hệ xương có quan hệ chặt chẽ với nhau vì trong cơ thể người chỗ nào có xương thì có cơ và ngược lại. Đây cũng là hai hệ của phần mềm không có phần mô phỏng bằng hoạt hình nên có thể được giới thiệu gộp vào trong 1 tiết.

GV cần chuẩn bị trước ở nhà các hoạt động sẽ trình bày trong bài giảng của mình. Vì trong 1 tiết không thể giới thiệu tất cả các bộ phận của hai hệ cơ, xương trong cơ thể người. GV chú ý gợi ý cho HS tìm hiểu các chi tiết và kiến thức gây hứng thú, ví dụ:

- Trong cơ thể người xương nào dài nhất?
- Cơ nào khỏe nhất?
- Xương chậu có các đặc điểm gì?
- Khu vực nào của con người có nhiều xương nhất?

Thông qua việc tìm hiểu như vậy HS sẽ thấy phần mềm rất hấp dẫn và có ích trong việc học tập môn Sinh học.

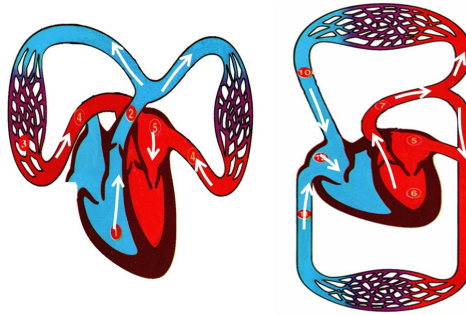
f) Với hệ **tuần hoàn**, cần chú ý các yếu tố sau:

Hệ **tuần hoàn** là một trong những kiến thức chính của SGK Sinh học 8 và cũng là phần nội dung được mô phỏng rất chi tiết trong phần mềm Anatomy. Vì vậy chúng tôi đề nghị phần này sẽ được hướng dẫn trong 1 tiết.

Chức năng quan trọng nhất của hệ tuần hoàn được mô tả trong phần mềm Anatomy là mô phỏng quá trình hoạt động chính của tim đưa máu đi khắp cơ thể. GV cần hướng dẫn để HS quan sát và hiểu được cấu tạo của tim và hoạt động của hai vòng tuần hoàn của con người.

VÒNG TUẦN HOÀN NHỎ

VÒNG TUẦN HOÀN LỚN



Vòng tuần hoàn lớn: Bắt đầu từ ngăn tâm thất trái, máu sau khi đã nạp đầy ô-xi được bơm qua một van (2 lá) lên động mạch chủ. Van này sẽ luôn đóng để bảo đảm máu không thể quay ngược lại được. Từ động mạch chủ máu được đưa lên đầu, xuống bụng và chân tay đi khắp cơ thể. Khi đi như vậy máu sẽ nhả ô-xi để nuôi dưỡng các tế bào và nhận cacbonic và các chất thải khác để quay lại đưa vào tim qua tâm nhĩ phải. Từ đây máu nghèo ô-xi sẽ chảy xuống tâm thất phải kết thúc một vòng tuần hoàn lớn.

Vòng tuần hoàn nhỏ: Bắt đầu từ tâm thất phải, máu nghèo ô-xi sẽ được bơm qua một van (2 lá) vào động mạch chủ của phổi. Từ đây máu sẽ đi qua hệ thống các tế bào phổi, đẩy cacbonic ra ngoài và nạp ô-xi. Sau khi nạp đầy ô-xi từ phổi, máu giàu ô-xi sẽ quay lại tâm nhĩ trái, sau đó được chuyển xuống tâm thất trái, kết thúc một vòng tuần hoàn nhỏ.

Như vậy hệ tuần hoàn của tim người bao gồm một vòng tuần hoàn lớn và một vòng tuần hoàn nhỏ. Quá trình này tiếp diễn liên tục.

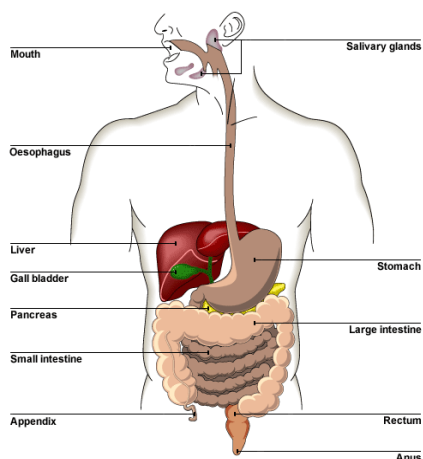
g) Với hệ hô hấp, cần chú ý các yếu tố sau:

Hệ hô hấp có chức năng đặc biệt là làm giàu ô-xi trong máu thông qua trao đổi chất với bên ngoài, ví dụ hít thở không khí. Thông qua hít thở, hệ hô hấp sẽ lấy ô-xi trong không khí và đưa vào máu, sau đó lấy cacbonic trong máu để thải ra ngoài không khí.

Quá trình hô hấp của con người bắt đầu từ khoang mũi. Không khí được đưa vào mũi, tại đây có các lông nhỏ để ngăn cản các vật cứng lớn. Khi được đưa xuống yết hầu, sau đó vào khí quản, tại đây không khí được lọc một lần nữa để đảm bảo sạch. Không khí sạch sẽ được đưa xuống và chạy trong khí quản dài hơn 10cm trước khi đi vào phế quản. Từ phế quản, không khí được toả ra thông qua các tiểu phế quản, sau đó vào các phế nang, tại đây có rất nhiều mao mạch nhỏ. Ô-xi được đưa vào máu và nhận cacbonic để đưa quay trở ra qua con đường cũ.

h) Với hệ tiêu hoá, cần chú ý các yếu tố sau:

GV chú ý đến tên các bộ phận của hệ tiêu hoá (tiếng Việt và tiếng Anh).



Miệng - Mouth
Tuyến nước bọt - Salivary glands
Thực quản - Oesophagus
Gan - Liver
Dạ dày - Stomach
Túi mật - Gall bladder
Tuyến tụy - Pancreas
Lá lách - Spleen
Tá tràng - Duodenum
Ruột già - Large intestine (colon)
Ruột non - Small intestine
Ruột thừa - Appendix
Trực tràng - Rectum
Hậu môn - Anus

Chức năng mô phỏng hệ tiêu hoá của phần mềm có thể tóm tắt như sau:

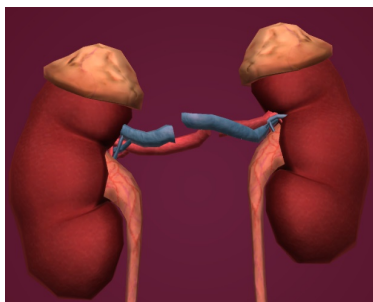
Thức ăn được đưa vào miệng, sau khi được biến đổi qua quá trình cơ học và sinh hoá ngắn (răng nhai, tuyến nước bọt tiết nước bọt) sẽ được đưa xuống thực quản qua yết hầu. Tại yết hầu có một bộ phận đảm bảo thức ăn không bị rơi vào khí quản. Thức ăn sau khi xuống dạ dày sẽ tiếp tục quá trình co bóp và biến đổi sinh hoá (do dạ dày co bóp, tụy, mật tiết dịch) sẽ được đưa xuống ruột non. Tại đây thức ăn đã được biến đổi hoàn toàn và sẽ bắt đầu được hấp thụ thông qua thành ruột. Quá trình hấp thụ này còn kéo dài đến tận ruột già. Ở giai đoạn cuối các chất không thể biến đổi và hấp thụ được nữa sẽ được đưa vào trực tràng và ra ngoài thông qua hậu môn.

i) Với hệ bài tiết, cần chú ý các yếu tố sau:

Hệ thống bài tiết có chức năng thải các chất độc ra bên ngoài cơ thể. Hệ thống này bao gồm thải khí cacbonic thông qua hít thở, thải mồ hôi qua da và thải nước tiểu qua thận. Tuy nhiên phần mềm chỉ mô phỏng chức năng bài tiết thải nước tiểu qua thận.

GV cần tìm hiểu kĩ chức năng mô phỏng hệ bài tiết trong phần mềm để hướng dẫn lại cho học sinh.

1. Quan sát hai quả thận chúng ta thấy mỗi quả đều có hai mạch máu đi vào và đi ra. Động mạch thận (màu đỏ) sẽ dẫn máu đi vào và tĩnh mạch thận (màu xanh) sẽ dẫn máu đi ra khỏi thận sau khi đã được lọc sạch.



2. Hình bên cho chúng ta quan sát bên trong mỗi quả thận. Bên trong quả thận chúng ta thấy một hệ thống các lỗ để nhận nước tiểu được sinh ra ngay bên trong thận để đưa vào niệu quản xuống bàng quang.



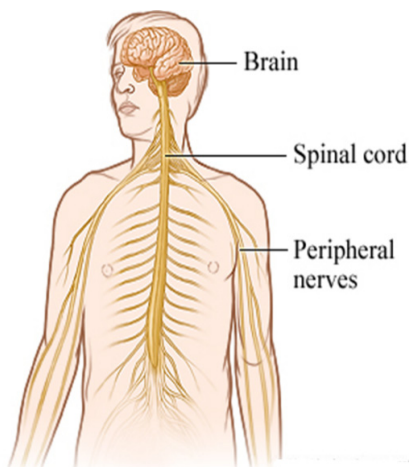
3. Quy trình hoạt động của thận như sau: động mạch (màu đỏ) dẫn máu đi vào thận. Từ đây thông qua các mạch nhỏ, máu sẽ đi đến từng tế bào để tiến hành lọc máu sạch, các chất thải ra sẽ chảy vào niệu quản, phần máu đã lọc sẽ theo tĩnh mạch thận (màu xanh) đi ra khỏi thận.



4. Nước tiểu sẽ theo niệu quản xuống bàng quang và sẵn sàng để thải ra ngoài qua niệu đạo.

k) Với hệ thần kinh, cần chú ý các yếu tố sau:

Hệ thần kinh có thể mô tả trong sơ đồ sau:



Hệ thần kinh của con người được chia làm 2 phần:

1. Hệ thần kinh trung ương bao gồm não (brain) và tuỷ sống (spinal cord).

2. Hệ thần kinh ngoại biên (peripheral nerves) bao gồm các dây và mạch thần kinh toả đi khắp cơ thể.

GV cần tìm hiểu kĩ chức năng mô phỏng hệ thần kinh trong phần mềm để hướng dẫn lại cho học sinh.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. Xem SGK.

Bài 2. Trong cơ thể người:

- Xương dài nhất: Xương đùi.
- Xương dài thứ hai: Xương cẳng chân (xương chày).

Chú ý: Xương cột sống bao gồm 33 đốt không được tính là một cá thể xương.

Bài 3. Trong tim người có hai van lớn. Các van này nằm trong các tâm thất trái và tâm thất phải.

- Van trong tâm thất trái có nhiệm vụ ngăn không cho máu chảy ngược trở lại tâm thất khi tim co bóp để đẩy máu đã giàu ô-xi từ tim vào động mạch chủ.
- Van trong tâm thất phải có nhiệm vụ ngăn không cho máu chảy ngược lại tâm thất phải khi tim co bóp đẩy máu vào phổi để làm giàu ô-xi.

Bài 4. Thức ăn qua đường miệng không bị chui vào khí quản vì đã có một bộ phận là yết hầu sẽ luôn che không cho thức ăn rơi xuống khí quản.

Bài 5. Đây là tên các bộ phận của hệ tiêu hoá con người.

ileum: đoạn cuối của ruột non (hồi tràng).

cecum: đoạn đầu của đại tràng (manh tràng).

ascending colon: đoạn ruột già bên phải (đại tràng lên)

transverse colon: đoạn ruột già ngang (đại tràng ngang).

descending colon: đoạn ruột già bên trái (đại tràng xuống).

sigmoid colon rectum: đoạn ruột già cuối cùng (đại tràng xích ma).

Bài 6 . – Với hệ bài tiết: máu từ động mạch chủ của cơ thể sẽ đi vào thận và sau đó lọc và thải chất độc trong thận, do vậy hình ảnh động mạch này phải là màu đỏ. Khi máu đã lọc sạch và đi ra sẽ được chảy vào tĩnh mạch chủ, nên hình ảnh các mạch máu này là màu xanh.

– Với hệ hô hấp: máu nghèo ô-xi sau khi đi khắp cơ thể sẽ quay trở về tâm thất phải của tim và được bơm vào phổi thông qua động mạch phổi, động mạch này trên hình sẽ có màu xanh để chỉ ra rằng đây là máu nghèo ô-xi. Sau khi được nạp đầy ô-xi từ phổi, máu sẽ được truyền qua các tĩnh mạch nhỏ để đưa sang tâm nhĩ trái của tim, mặc dù là tĩnh mạch nhưng máu này giàu ô-xi nên có màu đỏ trong các sơ đồ mô phỏng.

Bài 7 . – Cơ khoẻ nhất là cơ đùi.

– Cơ dài và lớn nhất là cơ lưng.

Bài 11. GIẢI TOÁN VÀ VẼ HÌNH PHẪNG VỚI GEOGEBRA

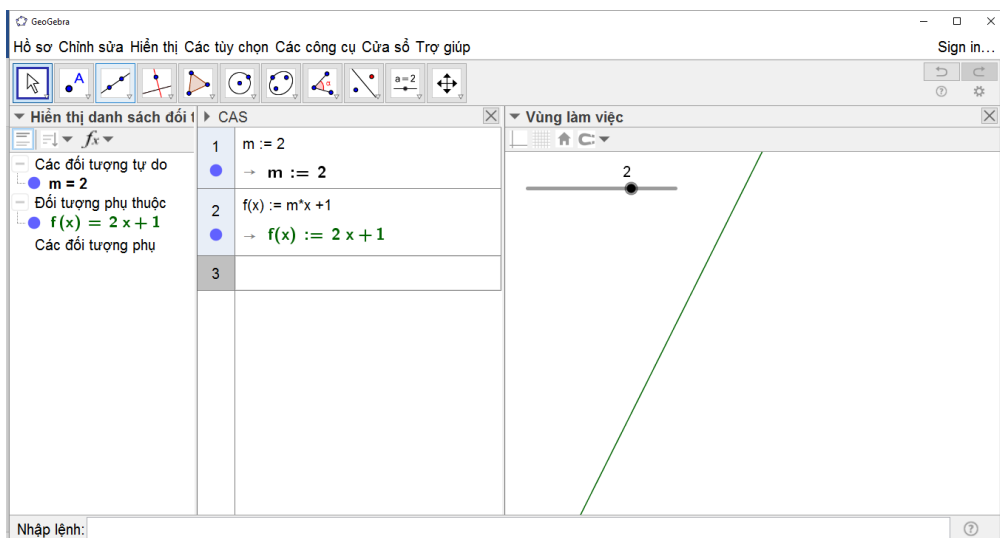
Thời lượng: 4 tiết

1. Mục đích, yêu cầu

- HS biết cách sử dụng phần mềm để thực hiện được các tính toán trên đa thức, phân thức đại số, giải phương trình và bất phương trình bậc nhất một ẩn số;
- HS biết cách sử dụng phần mềm để vẽ các hình phẳng chương trình môn Toán lớp 8;
- HS có ý thức trong việc ứng dụng phần mềm trong việc học tập của mình.

2. Những điểm cần lưu ý và gợi ý dạy học

- a) Bài học dự kiến sẽ được giảng dạy trong 4 tiết, 1 tiết sẽ tập trung vào các công cụ CAS, 3 tiết tập trung vào các công cụ hình học phẳng.
- b) Trong tiết 1 nên thiết lập chế độ thể hiện màn hình như sau:



Còn trong các tiết tiếp theo có thể đóng cửa sổ CAS.

c) Bài này dạy trong 4 tiết, trong đó 2 tiết lí thuyết và 2 tiết thực hành trên máy tính. Dự kiến lịch trình giảng dạy như sau:

- Tiết 1: Tập trung vào thực hiện các tính toán đại số trên cửa sổ CAS: tính toán với đa thức, phân thức đại số, giải phương trình đại số.
- Tiết 2: Quan hệ toán học giữa các đối tượng hình học. Các công cụ hình học cơ bản (điểm, đường, đường song song, vuông góc).
- Tiết 3: Làm quen các công cụ biến đổi hình học: đối xứng qua trục, qua tâm.
- Tiết 4: Sử dụng các công cụ đường tròn.

d) GV cần chú ý và phân biệt về cách thức, sự giống nhau và khác nhau giữa các dòng nhập lệnh trong cửa sổ CAS và dòng nhập lệnh phía dưới màn hình.

e) Phần kiến thức quan trọng nhất cần trình bày cho HS là khái niệm quan hệ giữa các đối tượng hình học. Chính các quan hệ logic chặt chẽ này giữa các đối tượng hình học sẽ tạo ra khái niệm "hình học động" của phần mềm. Đây là điểm khác biệt chính nhất của phần mềm này với các phần mềm vẽ đồ họa khác.

f) Trong cửa sổ danh sách các đối tượng ta sẽ nhìn thấy có ba loại đối tượng trong phần mềm GeoGebra: đối tượng tự do, đối tượng phụ thuộc.

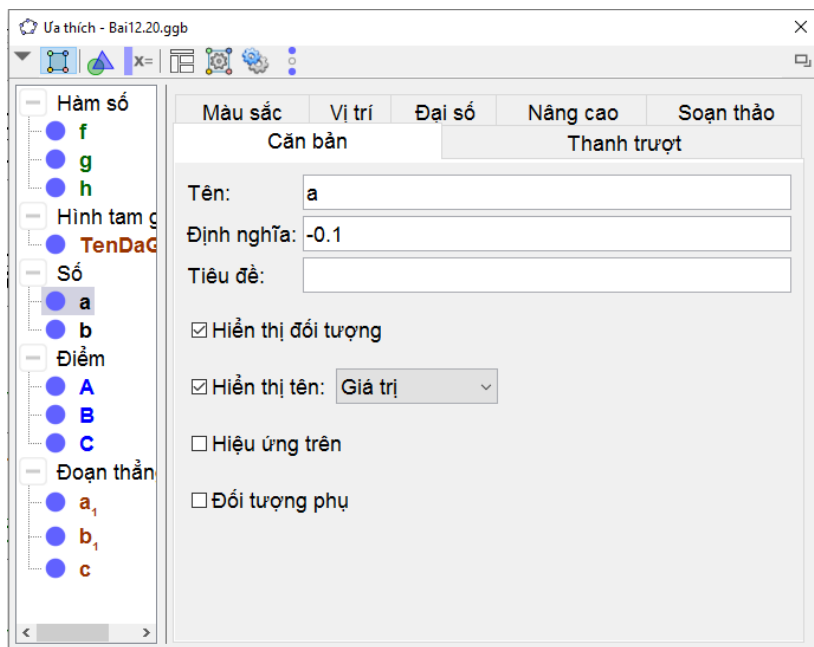
– Đối tượng tự do là các đối tượng không phụ thuộc vào bất cứ đối tượng nào trong hình vẽ. Thông thường đối tượng tự do trong GeoGebra là các điểm và số.

– Phần lớn các đối tượng còn lại của phần mềm đều phụ thuộc vào các đối tượng tự do này (và các đối tượng khác), chúng được gọi là đối tượng phụ thuộc.

- g) Ngoài các cách thay đổi thuộc tính như đã trình bày trong SGK, còn có một cách tổng quát nhất là làm xuất hiện hộp thoại thuộc tính của đối tượng.

Nháy nút phải chuột lên đối tượng và chọn lệnh **Thuộc tính**, hộp thoại **Thuộc tính** của đối tượng xuất hiện.

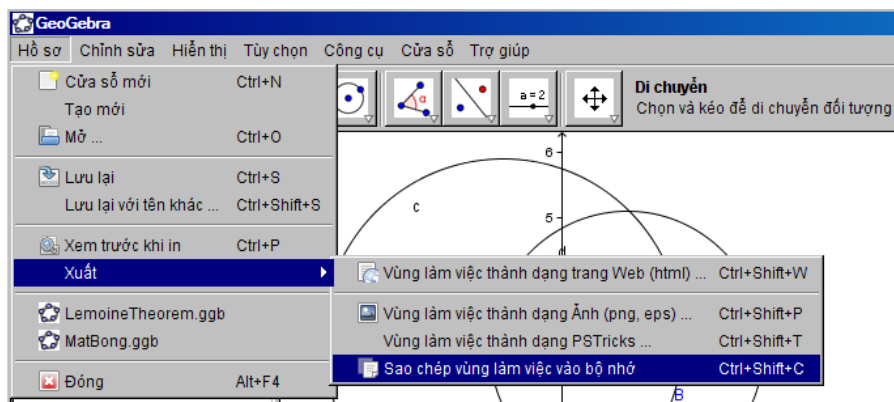
Khung bên trái là danh sách tất cả các đối tượng hiện có trên hình. Bên phải là khung hiển thị thông tin thuộc tính của đối tượng được chọn. Có thể xem và thay đổi các thuộc tính này. Một số thuộc tính quan trọng như màu, kiểu đường và kiểu nền có thể thay đổi trong hộp thoại này.



Hộp thoại thuộc tính của đối tượng

- h)** Phần mềm GeoGebra không cho phép sao chép trực tiếp từng đối tượng hình học riêng biệt sang các phần mềm khác. Nếu muốn sao chép toàn bộ hình vẽ trên màn hình sang các phần mềm khác ta thực hiện lệnh sau:

Hồ sơ → Xuất → Sao chép vùng làm việc vào bộ nhớ.



Sau lệnh trên, ta có thể mở một phần mềm ứng dụng khác (ví dụ Word) dùng lệnh **Paste** để đưa hình vẽ từ bộ nhớ máy tính vào phần mềm đang mở.

3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. a) 220825

b) $x^4 - x^3y + x^2y + xy^3 - xy^2 - y^4$

Bài 2. a) $(y^2 + 1)(x + y)(x^2 + 1)$

b) $x(x + y - 3)(x + y + 3)$

Bài 3. a) $\frac{(x+2)}{2x}$

b) $\frac{(2x^3 - 1)}{x}$

c) $x + y$

Bài 4. a) Hai nghiệm $x = \frac{1}{3}; x = 3$

b) $x = 2$.

Bài 5. a) $x \leq \frac{7}{10}$

b) $x > 2$

c) $x < 0$ hoặc $x > 7$

Bài 6. Công cụ đa giác được thực hiện như sau: lần lượt nháy chuột lên các điểm là đỉnh của đa giác, điểm cuối cùng trùng với điểm đầu tiên.

Bài 12. VẼ HÌNH KHÔNG GIAN VỚI GEOGEBRA

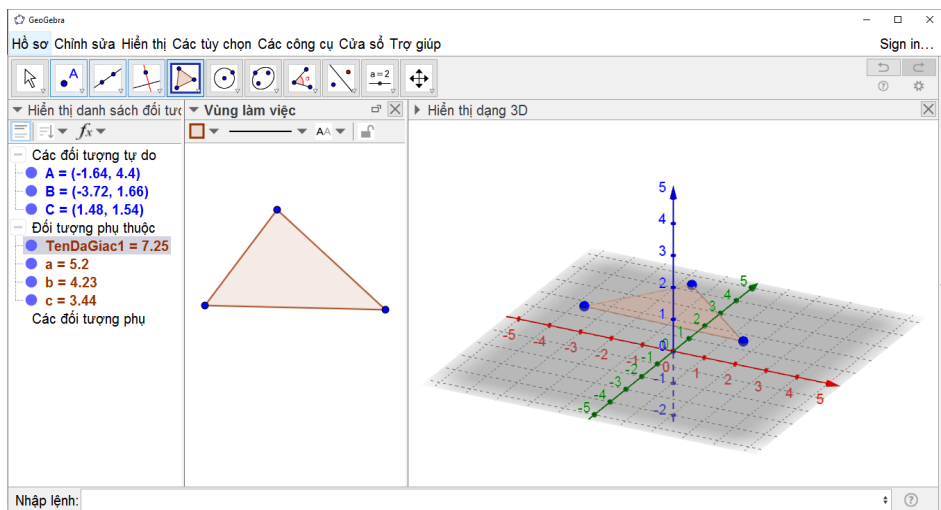
Thời lượng: 4 tiết

1. Mục đích, yêu cầu

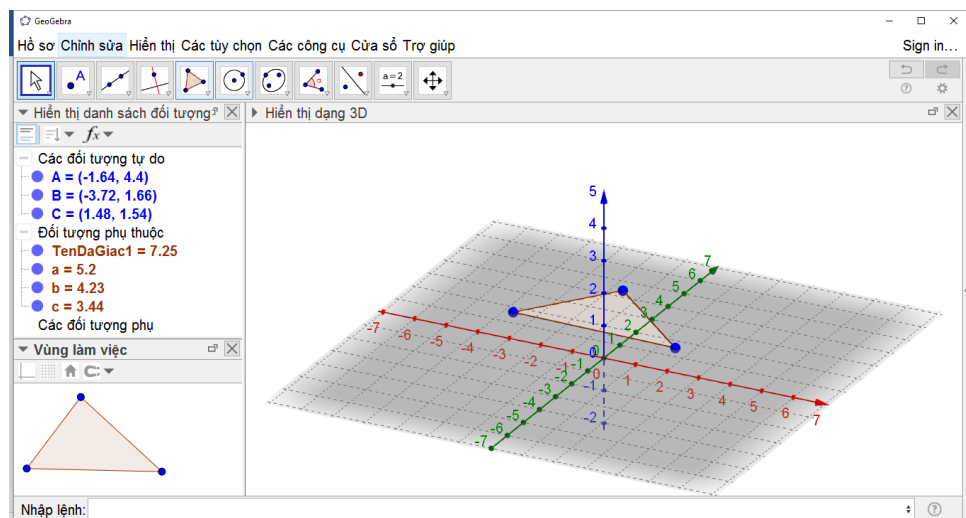
- HS bước đầu thực hiện được vẽ các hình không gian đơn giản như hình chóp, hình lăng trụ và hình hộp trong không gian 3D với GeoGebra.
- Thông qua phần mềm, HS hiểu được cách thể hiện các đối tượng cơ sở trong không gian 3D như điểm, đường thẳng.

2. Những điểm cần lưu ý và gợi ý dạy học

- a) Phần mềm GeoGebra hỗ trợ cho việc vẽ và thể hiện các đối tượng hình học trong không gian 3D, tuy vậy đây là phần kiến thức khó, GV cần tìm hiểu thật kỹ phần mềm trước khi hướng dẫn cho HS.
- b) Trong khi làm việc với cửa sổ 3D, nhất thiết cần giữ lại cửa sổ mặt phẳng làm việc chính của GeoGebra. Có thể thiết lập màn hình làm việc như hình a hoặc b sau:



Hình a



Hình b

c) Có thể phân bổ nội dung dạy học như sau:

Tiết 1: Nhận biết không gian làm việc 3D và liên hệ với các cửa sổ khác.

Thiết lập và di chuyển điểm trong không gian 3D. Xoay không gian 3D.

Tiết 2: Vẽ hình hộp, hình lập phương.

Tiết 3: Vẽ hình lăng trụ đứng.

Tiết 4: Vẽ hình chóp.

Sau đây là một số chú ý quan trọng khi giảng dạy phần dùng công cụ vẽ hình không gian trong GeoGebra.

d) GV luôn thiết lập chế độ màn hình để hiển thị đồng thời ba cửa sổ:

– Cửa sổ danh sách đối tượng.

– Cửa sổ mặt phẳng 2D.

– Cửa sổ không gian 3D.

Lưu ý: các đối tượng trong mặt phẳng 2D và không gian 3D đều thể hiện trong cửa sổ danh sách đối tượng.

e) Trong không gian 3D, các thao tác cơ sở sau là quan trọng:

1. Điều khiển một điểm tự do di chuyển trong không gian. Chú ý rằng chỉ được phép di chuyển điểm tự do theo một trong hai cách: theo hướng thẳng đứng và theo hướng mặt phẳng ngang.

2. Xoay toàn bộ không gian 3D bằng cách nhấn giữ nút phải chuột và kéo thả chuột, hoặc chuyển về chế độ chọn và kéo thả chuột.

f) Vẽ hình hộp chữ nhật, hình lập phương

Trong SGK trình bày hai cách để tạo hình hộp chữ nhật và một cách tạo hình lập phương. GV chú ý giới thiệu cả ba cách để HS nắm được.

g) Vẽ hình lăng trụ

Trong SGK trình bày hai cách vẽ hình lăng trụ. GV giới thiệu cả hai cách này để HS nắm được.

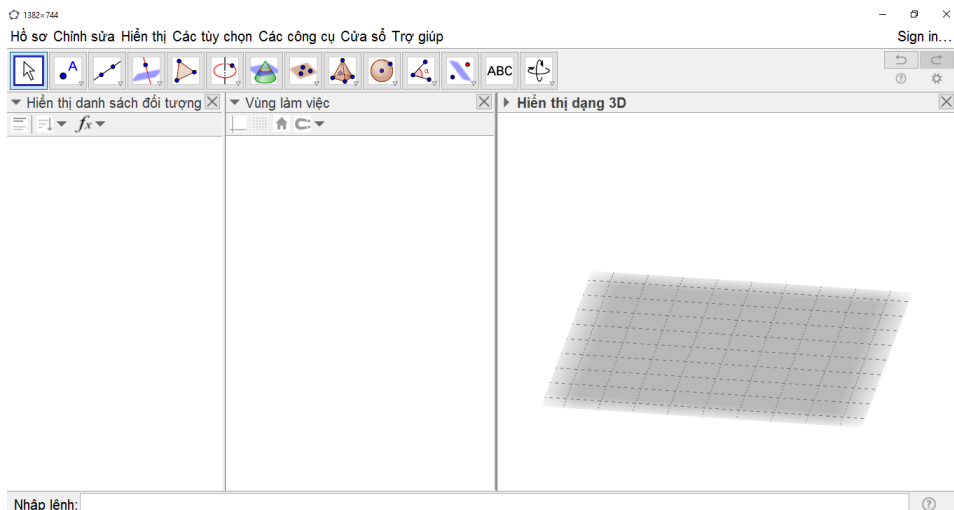
h) Vẽ hình chóp


Trong SGK trình bày hai cách vẽ hình chóp. GV giới thiệu cả hai cách này để HS nắm được.


3. Hướng dẫn trả lời câu hỏi và bài tập

Bài 1. Hướng dẫn: có thể thực hiện theo cách sau.

– Thiết lập các cửa sổ làm việc của GeoGebra như hình dưới đây.



– Nháy chuột lên cửa sổ giữa (mặt phẳng 2D) để kích hoạt cửa sổ này. Dùng công cụ đa giác đều  để vẽ các tứ giác, ngũ giác, lục giác đều.


– Nháy chuột lên cửa sổ không gian 3D để kích hoạt cửa sổ này. Dùng công cụ trái hình chóp  để vẽ các hình như yêu cầu.

Bài 2. Không. Mặt phẳng chuẩn không là một đối tượng toán học trong GeoGebra.

Bài 3. Cách vẽ tương tự bài 1.

Bài 4. Gợi ý cách vẽ:

– Sử dụng cách làm tương tự bài 1 để vẽ một hình lăng trụ đứng có đáy là ngũ giác đều.

– Sử dụng công cụ trải hình lăng trụ  lần lượt thực hiện thao tác sau: nháy lên một mặt bên của lăng trụ, sau đó kéo thả chuột ra phía bên ngoài của lăng trụ để tạo thêm các hình như theo yêu cầu của bài tập.

Bài 5. Không. Chỉ có thể di chuyển một điểm tự do trong không gian theo một trong hai cách:

– Di chuyển theo hướng thẳng đứng.

– Di chuyển trên mặt phẳng nằm ngang.

Bài 6. Mở bảng chọn **Hiển thị** để hiển thị cả ba cửa sổ theo yêu cầu. Sau đó dùng chuột kéo thả các cửa sổ này (chú ý kéo thả chuột lên dòng tên của các cửa sổ) để thu được cách sắp xếp như yêu cầu.

Bài 7. Có thể vẽ theo bước sau:

Bước 1: Vẽ hình chóp tam giác ABCD với tam giác đáy là BCD, điểm A nằm ở phía trên.

Bước 2: Dùng chuột di chuyển điểm D xuống phía dưới mặt phẳng chuẩn như hình vẽ trong đề bài.

Bài 9. Thao tác tương tự như vẽ đường thẳng song song trên mặt phẳng.

Phụ lục

I. MỘT SỐ PHÉP TOÁN THƯỜNG DÙNG

Sau đây là các phụ lục liên quan đến việc sử dụng Pascal.

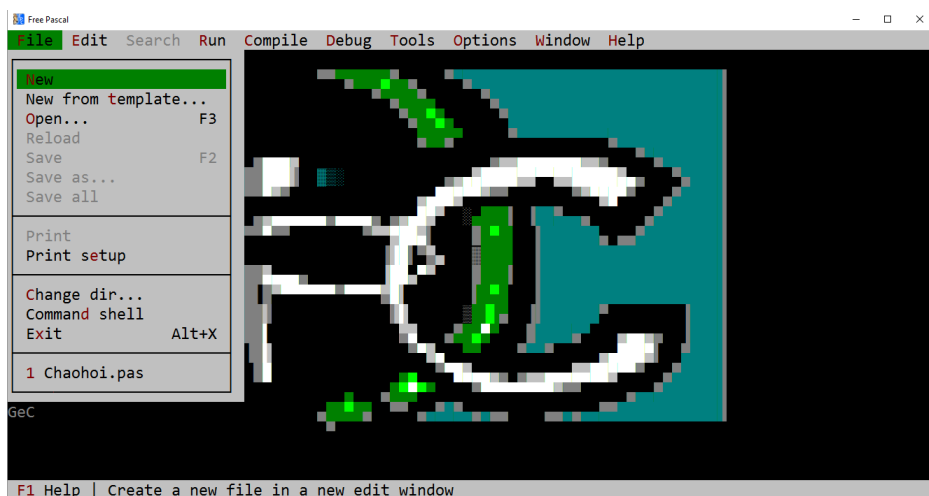
Tên phép toán	Kí hiệu toán học	Kí hiệu trong Pascal	Loại phép toán
Cộng	+	+	Số học
Trừ	−	−	
Nhân	×	*	
Chia	/	/	
Chia nguyên		div	
Lấy phần dư		mod	
Nhỏ hơn	<	<	So sánh
Nhỏ hơn hoặc bằng	≤	<=	
Lớn hơn	>	>	
Lớn hơn hoặc bằng	≥	>=	
Bằng	=	=	
Khác	≠	<>	
Phủ định	¬	not	Logic
Hoặc (tuyển)	∨	or	
Và (hội)	∧	and	

II. MÔI TRƯỜNG FREE PASCAL

Bảng chọn (menu) của FP được kích hoạt bằng cách nhấn phím **F10**. Khi nhấn các phím mũi tên → hay ← ta có thể chuyển tới các mục khác nhau của bảng chọn. Khi nhấn phím **Enter**, bảng chọn sẽ được kích hoạt, giới thiệu các mục công việc. Các bảng chọn thường được dùng là **File** (Tập), **Run** (Thực hiện), **Debug** (Gỡ rối) và **Options** (Tuỳ chọn).

1. Bảng chọn File

Khi kích hoạt bảng chọn này ta có màn hình như hình P.1:



Hình P.1. Bảng chọn File

Một số lệnh chính trong bảng chọn này gồm:

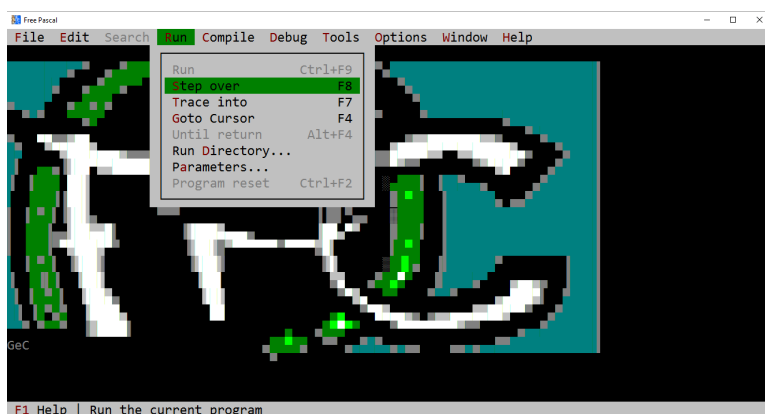
- **New** - Mở cửa sổ mới để soạn thảo chương trình.
- **Open** - Mở tệp đã có trên đĩa. Có thể thực hiện lệnh này bằng cách nhấn phím **F3**. Trên màn hình sẽ xuất hiện cửa sổ để xác định tên tệp cần mở.
- **Save** - Để lưu tệp đang soạn thảo. Nếu đây là tệp chưa đặt tên thì FP sẽ hỏi tên tệp để lưu trữ. Có thể thực hiện lệnh này bằng cách nhấn phím **F2**.
- **Save as** - Để lưu tệp đang soạn thảo với tên mới.
- **Save all** - Để lưu tất cả các tệp đang mở.
- **Change dir** - Thay đổi thư mục chủ.
- **Command shell** - Chuyển sang màn hình dòng lệnh (tương tự như DOS, ở đó có thể thực hiện các lệnh của MS-DOS. Để quay trở về màn hình FP cần gõ lệnh **EXIT**).

- **Exit** - Thoát khỏi FP. Có thể thoát khỏi FP bằng cách nhấn tổ hợp phím **Alt+X**.

Trong một số trường hợp cụ thể, có thể có những mục chưa được phép chọn. Những mục này sẽ hiển thị dưới dạng mờ hơn các mục có thể chọn.

2. Bảng chọn Run

Khi kích hoạt bảng chọn này ta có màn hình tương tự hình P.2. Bảng chọn này dùng để xác định chế độ thực hiện chương trình.



Hình P.2. Bảng chọn Run

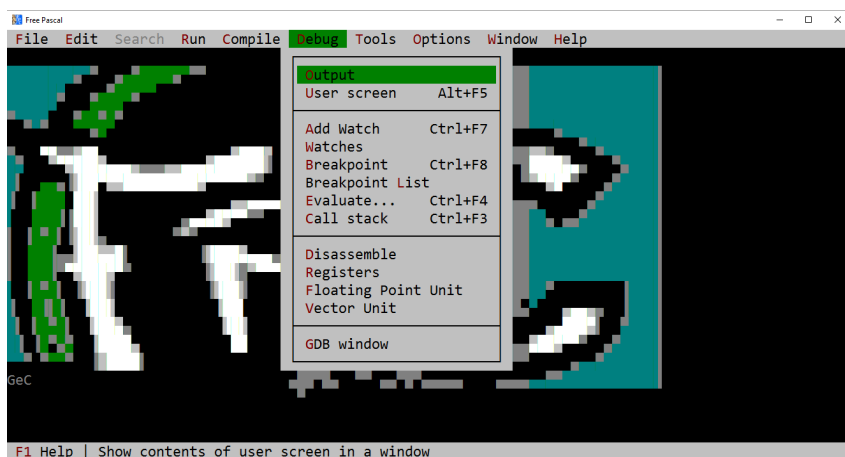
Các lệnh có thể chọn là:

- **Run** - Thực hiện chương trình nếu trước đó chương trình đã được dịch bằng cách nhấn tổ hợp phím **Alt+F9** và chương trình không có lỗi. Nếu chương trình chưa được dịch, FP sẽ dịch và thực hiện chương trình khi chương trình nguồn không có lỗi cú pháp. Nếu chương trình nguồn có lỗi cú pháp, FP sẽ thông báo lỗi và con trỏ màn hình sẽ nhấp nháy ở dòng có lỗi. Có thể nhấn tổ hợp phím **Ctrl+F9** để truy cập nhanh lệnh này.
- **Step over** - Thực hiện từng dòng lệnh trên màn hình soạn thảo. Các lời gọi chương trình con được xem như một lệnh. Cần lưu ý là một dòng lệnh trên màn hình soạn thảo có thể chứa *nhiều câu lệnh* của FP. Có thể chọn nhanh lệnh này bằng cách nhấn phím **F8**.
- **Trace into** - Thực hiện từng dòng lệnh trên màn hình soạn thảo, kể cả các dòng lệnh ở chương trình con. Có thể chọn nhanh lệnh này bằng cách nhấn phím **F7**.
- **Goto Cursor** - Thực hiện chương trình cho đến dòng có con trỏ màn hình. Có thể chọn nhanh lệnh này bằng cách nhấn phím **F4**.

- **Program reset** - Xóa các trạng thái của FP đối với chương trình đang thực hiện để chuẩn bị dịch và thực hiện lại từ đầu. Có thể chọn nhanh lệnh này bằng cách nhấn tổ hợp phím **Ctrl+F2**.

3. Bảng chọn Debug

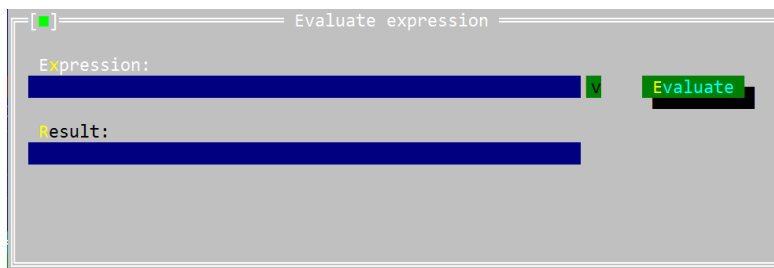
Bảng chọn này dùng để hiệu chỉnh chương trình. Khi kích hoạt bảng chọn này ta có màn hình tương tự hình P.3.



Hình P.3. Bảng chọn Debug

Các lệnh thường dùng là:

- **Evaluate** - Dùng để tính giá trị biểu thức. FP sẽ mở cửa sổ để gõ biểu thức cần tính. Sau khi gõ biểu thức, nhấn phím **Enter**, FP sẽ tính và cho giá trị ở cửa sổ giá trị (h. P.4).



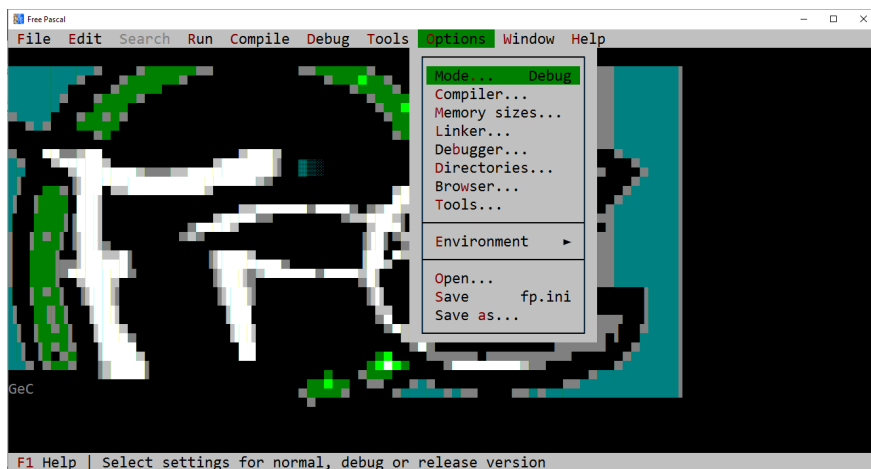
Hình P.4. Cửa sổ tính giá trị biểu thức

- **Add Watch** - Dùng để mở cửa sổ theo dõi giá trị biến trong quá trình thực hiện chương trình. Có thể chọn nhanh lệnh này bằng cách nhấn tổ hợp phím **Ctrl+F7**.
- **Breakpoint** - dùng để tạo ra các vị trí dừng trong chương trình. Khi chạy chương trình, đến dòng lệnh này chương trình sẽ dừng lại để người lập trình

có thể quan sát kết quả hoặc các thông số (ví dụ các thông số watch) của chương trình. Đây là một công cụ hữu hiệu của chức năng debug.

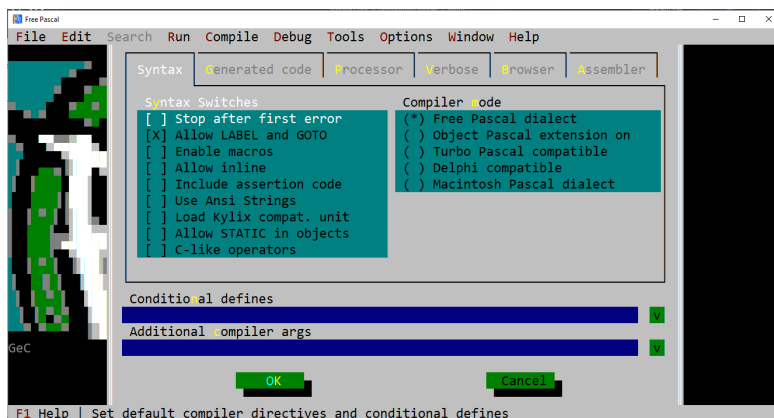
4. Bảng chọn Options

Bảng chọn này được dùng để đặt các tùy chọn cho môi trường lập trình. Mục thường dùng hơn cả là **Compiler** (h.P.5).



Hình P.5. Bảng chọn Options

Khi chọn mục **Compiler**, màn hình tương tự như hình P.6 xuất hiện. Ta có thể xác lập các tùy chọn cho chương trình dịch. Các tùy chọn này cũng có thể xác lập ngay trong chương trình nguồn bằng câu lệnh của FP. Các tùy chọn được chia thành nhiều nhóm. Việc chuyển từ nhóm này sang nhóm khác được thực hiện bằng cách nhấn phím **Tab**. Việc chọn hay không chọn một tùy chọn được thực hiện bằng cách gõ *phím cách*. Những tùy chọn có hiệu lực được đánh dấu **X** (h. P.6).



Hình P.6. Các tùy chọn cho chương trình dịch

5. Chuyển cửa sổ màn hình soạn thảo

FP cho phép mở nhiều cửa sổ soạn thảo đồng thời. Để chuyển tới cửa sổ tiếp theo, nhấn phím **F6**. Để quay về cửa sổ trước đó, nhấn tổ hợp phím **Shift+F6**.

6. Trợ giúp

Phân trợ giúp của FP đi kèm chương trình không có, cần vào trang trợ giúp online của FP tại địa chỉ:

<http://www.freepascal.org/docs-html/current/user/user.html>

III. MỘT SỐ KIỂU DỮ LIỆU CHUẨN

Kiểu	Loại giá trị	Bộ nhớ (byte)	Phạm vi giá trị
byte	Nguyên	1	từ 0 đến 255
integer	Nguyên	2	từ -32768 đến 32767
word	Nguyên	2	từ 0 đến 65535
longint	Nguyên	4	từ -2147483648 đến 2147483647 từ -2^{31} đến $2^{31}-1$
real	Thực	6	0 hoặc có giá trị tuyệt đối trong khoảng từ $1,5 \times 10^{-45}$ đến $3,4 \times 10^{38}$
char	Kí tự	1	Kí tự bất kì
boolean	Lôgic	1	true, false

IV. MỘT SỐ THỦ TỤC VÀ HÀM CHUẨN

1. Đối với các biến kiểu nguyên

Thủ tục/hàm	Chức năng
Inc (x)	Tăng giá trị của biến x một đơn vị.
Dec (x)	Giảm giá trị của biến x một đơn vị.
Inc (x, y)	Đặt cho biến x giá trị mới bằng giá trị cũ cộng với giá trị của biến y.

Dec (x, y)	Đặt cho biến x giá trị mới bằng giá trị cũ trừ đi giá trị của biến y.
Sqr (x)	Cho giá trị bằng bình phương của x.
Pred (x)	Cho giá trị bằng $x - 1$.
Succ (x)	Cho giá trị bằng $x + 1$.
Random (N)	Hàm có biểu thức N kiểu <i>word</i> và cho giá trị là một số nguyên ngẫu nhiên trong phạm vi từ 0 đến $N - 1$. Khi dùng hàm này ta phải gọi thủ tục <i>randomize</i> .

2. Đối với các biến kiểu thực

Hàm	Chức năng
Abs (x)	Cho giá trị bằng trị tuyệt đối của giá trị biến x hoặc số thực x.
ArcTan (x)	Cho giá trị là số đo của cung thuộc khoảng $\left(-\frac{\pi}{2}; \frac{\pi}{2}\right)$ có tang bằng giá trị của biến x hay số thực x.
Exp (x)	Cho giá trị bằng lũy thừa cơ số e của giá trị biến x hoặc số thực x.
Ln (x)	Cho giá trị bằng lôgarit cơ số e của giá trị biến x hoặc số thực x.
Sin (x)	Cho giá trị bằng $\sin x$.
Cos (x)	Cho giá trị bằng $\cos x$.
Pi	Cho giá trị của số π (3,1415...).
Int (x)	Cho giá trị bằng phần nguyên nhưng có kiểu số thực của giá trị biến x hoặc số thực x (phần nguyên của số thực x bằng số nguyên lớn nhất không vượt quá x).
Sqr (x)	Cho giá trị bằng bình phương của giá trị biến x hoặc số thực x.
Sqrt (x)	Cho giá trị bằng căn bậc hai của giá trị không âm của biến x hoặc số thực không âm x.
Randomize	Thủ tục khởi động sinh số ngẫu nhiên.

Hàm	Chức năng
Random	Cho một số thực ngẫu nhiên trong khoảng (0;1). Khi dùng hàm này ta phải gọi thủ tục randomize.
Round (x)	Cho giá trị bằng số nguyên gần số thực x nhất nhưng có kiểu là kiểu số nguyên. Trong trường hợp phần phân của x lớn hơn 0,5 thì hàm cho giá trị làm tròn lên.
Trunc (x)	Cho giá trị bằng phần nguyên của x.

3. Hàm chuẩn trả về giá trị logic

Hàm	Chức năng
Odd (x)	Với biểu thức số nguyên x, cho giá trị true nếu x lẻ và cho giá trị false nếu x chẵn.

4. Đối với biến kiểu kí tự

Thủ tục/hàm	Chức năng
Inc (x)	Cho giá trị của biến x là kí tự đứng ngay sau kí tự ứng với giá trị hiện thời của x trong bộ mã ASCII.
Dec (x)	Cho giá trị của biến x là kí tự đứng ngay trước kí tự ứng với giá trị hiện thời của x trong bộ mã ASCII.
Chr (x)	Cho giá trị là kí tự có mã ASCII thập phân bằng (giá trị của biểu thức) có giá trị nguyên từ 0 đến 255.
Ord (c)	Cho giá trị mã ASCII thập phân của kí tự c.
Pred (c)	Cho kí tự đứng ngay trước kí tự c trong bộ mã ASCII.
Succ (c)	Cho kí tự đứng ngay sau kí tự c trong bộ mã ASCII.
UpCase (c)	Nếu c là chữ cái tiếng Anh, hàm cho giá trị bằng chữ cái hoa tương ứng, ngược lại, hàm cho giá trị bằng giá trị của c.

MỤC LỤC

	Trang
PHẦN MỘT. NHỮNG VẤN ĐỀ CHUNG	3
I. Vài nét chung về môn tin học và sách giáo khoa chỉnh lí	3
II. Giới thiệu chương trình môn tin học	7
III. Giới thiệu sách giáo khoa "Tin học dành cho Trung học cơ sở, Quyển 3"	12
IV. Gợi ý về tổ chức dạy học	19
PHẦN HAI. NHỮNG VẤN ĐỀ CỤ THỂ	24
Chương I. Lập trình đơn giản	24
I. Giới thiệu	24
II. Hướng dẫn chi tiết	25
Bài 1. Máy tính và chương trình máy tính	25
Bài 2. Làm quen với chương trình và ngôn ngữ lập trình	29
Bài thực hành 1. Làm quen với Free Pascal	34
Bài 3. Chương trình máy tính và dữ liệu	42
Bài thực hành 2. Viết chương trình để tính toán	47
Bài 4. Sử dụng biến trong chương trình	49
Bài thực hành 3. Khai báo và sử dụng biến	51
Bài 5. Từ bài toán đến chương trình	54
Bài 6. Câu lệnh điều kiện	59
Bài thực hành 4. Sử dụng lệnh điều kiện if...then	64
Bài 7. Câu lệnh lặp	70
Bài thực hành 5. Sử dụng lệnh lặp for...do	74
Bài 8. Lặp với số lần chưa biết trước	77
Bài thực hành 6. Sử dụng lệnh lặp while...do	82
Bài 9. Làm việc với dãy số	85
Bài thực hành 7. Xử lí dãy số trong chương trình	92
Chương II. Phần mềm học tập	94
I. Giới thiệu	94
II. Hướng dẫn chi tiết	96
Bài 10. Làm quen với giải phẫu cơ thể người bằng phần mềm Anatomy	96
Bài 11. Giải toán và vẽ hình phẳng với GeoGebra	103
Bài 12. Vẽ hình không gian với GeoGebra	107
Phụ lục	111

Chịu trách nhiệm xuất bản :
Chủ tịch Hội đồng Thành viên NGUYỄN ĐỨC THÁI
Tổng Giám đốc HOÀNG LÊ BÁCH
Phó Tổng Giám đốc kiêm Tổng biên tập TS. PHAN XUÂN THÀNH

Biên tập lần đầu:
NGUYỄN THỊ NGUYỄN THUYẾT- NGUYỄN THỊ THANH XUÂN
Biên tập tái bản:
NGUYỄN THỊ NGUYỄN THUYẾT- DƯƠNG VŨ KHÁNH THUẬN

Trình bày bìa :
LƯU CHÍ ĐỒNG

Chế bản :
CÔNG TY CP DỊCH VỤ XUẤT BẢN GIÁO DỤC HÀ NỘI

Bản quyền thuộc Nhà xuất bản Giáo dục Việt Nam -
Bộ Giáo dục và Đào tạo.

**TIN HỌC DÀNH CHO TRUNG HỌC CƠ SỞ, QUYỂN 3
– SÁCH GIÁO VIÊN**

Mã số : 2B826T7

Số đăng kí KHXB : 747-2017/CXBIPH/5-308/GD

In cuốn (QĐ in số :), khổ 17 × 24 cm.

In tại Công ty cổ phần in

In xong và nộp lưu chiểu tháng năm 2017.

Mã ISBN: 978-604-0-10146-4