

MỤC LỤC

| | | |
|----------------|---|-----------|
| BÀI 1 : | NGÔN NGỮ LẬP TRÌNH & PHƯƠNG PHÁP LẬP TRÌNH | 7 |
| 1.1 | Mục tiêu | 7 |
| 1.2 | Lý thuyết..... | 7 |
| 1.2.1 | Ngôn ngữ lập trình (Programming Language) | 7 |
| 1.2.1.1 | Thuật giải (Algorithm) | 7 |
| 1.2.1.2 | Chương trình (Program) | 7 |
| 1.2.1.3 | Ngôn ngữ lập trình (Programming language) | 8 |
| 1.2.2 | Các bước lập trình | 8 |
| 1.2.3 | Kỹ thuật lập trình | 8 |
| 1.2.3.1 | I-P-O Cycle (Input-Process-Output Cycle) (Quy trình nhập-xử lý-xuất) | 8 |
| 1.2.3.2 | Sử dụng lưu đồ (Flowchart)..... | 9 |
| BÀI 2 : | LÀM QUEN LẬP TRÌNH C QUA CÁC VÍ DỤ ĐƠN GIẢN | 12 |
| 2.1 | Mục tiêu | 12 |
| 2.2 | Nội dung..... | 12 |
| 2.2.1 | Khởi động và thoát BorlandC..... | 12 |
| 2.2.1.1 | Khởi động | 12 |
| 2.2.1.2 | Thoát..... | 13 |
| 2.2.2 | Các ví dụ đơn giản | 13 |
| 2.2.2.1 | Ví dụ 1..... | 13 |
| 2.2.2.2 | Ví dụ 2..... | 15 |
| 2.2.2.3 | Ví dụ 3..... | 16 |
| 2.2.2.4 | Ví dụ 4..... | 16 |
| BÀI 3 : | CÁC THÀNH PHẦN TRONG NGÔN NGỮ C | 18 |
| 3.1 | Mục tiêu | 18 |
| 3.2 | Nội dung..... | 18 |
| 3.2.1 | Từ khóa | 18 |
| 3.2.2 | Tên..... | 18 |
| 3.2.3 | Kiểu dữ liệu | 18 |
| 3.2.4 | Ghi chú..... | 19 |
| 3.2.5 | Khai báo biến | 19 |
| 3.2.5.1 | Tên biến | 19 |
| 3.2.5.2 | Khai báo biến | 19 |
| 3.2.5.3 | Vừa khai báo vừa khởi gán..... | 20 |
| 3.2.5.4 | Phạm vi của biến..... | 20 |
| BÀI 4 : | NHẬP / XUẤT DỮ LIỆU | 21 |
| 4.1 | Mục tiêu | 21 |
| 4.2 | Nội dung..... | 21 |

- 4.2.1 Hàm printf..... 21
- 4.2.2 Hàm scanf 24
- 4.3 Bài tập..... 25
- BÀI 5 : CẤU TRÚC RỄ NHÁNH CÓ ĐIỀU KIỆN 26**
- 5.1 Mục tiêu 26
- 5.2 Nội dung..... 26
 - 5.2.1 Lệnh và khối lệnh..... 26
 - 5.2.1.1 Lệnh 26
 - 5.2.1.2 Khối lệnh 26
 - 5.2.2 Lệnh if 26
 - 5.2.2.1 Dạng 1 (if thiếu) 26
 - 5.2.2.2 Dạng 2 (if đủ) 30
 - 5.2.2.3 Cấu trúc else if 33
 - 5.2.2.4 Cấu trúc if lồng 37
 - 5.2.3 Lệnh switch..... 41
 - 5.2.3.1 Cấu trúc switch...case (switch thiếu)..... 41
 - 5.2.3.2 Cấu trúc switch...case...default (switch đủ)..... 44
 - 5.2.3.3 Cấu trúc switch lồng..... 46
- 5.3 Bài tập..... 48
 - 5.3.1 Sử dụng lệnh if 48
 - 5.3.2 Sử dụng lệnh switch 49
- 5.4 Bài tập làm thêm..... 49
- BÀI 6 : CẤU TRÚC VÒNG LẶP 51**
- 6.1 Mục tiêu 51
- 6.2 Nội dung..... 51
 - 6.2.1 Lệnh for..... 51
 - 6.2.2 Lệnh break..... 56
 - 6.2.3 Lệnh continue 56
 - 6.2.4 Lệnh while..... 56
 - 6.2.5 Lệnh do...while 58
 - 6.2.6 Vòng lặp lồng nhau 60
 - 6.2.7 So sánh sự khác nhau của các vòng lặp 61
- 6.3 Bài tập..... 62
- BÀI 7 : HÀM..... 65**
- 7.1 Mục tiêu 65
- 7.2 Nội dung..... 65
 - 7.2.1 Các ví dụ về hàm 65
 - 7.2.2 Tham số dạng tham biến và tham trị..... 68

| | | |
|----------------|--|-----------|
| 7.2.3 | Sử dụng biến toàn cục | 69 |
| 7.2.4 | Dùng dẫn hướng #define | 71 |
| 7.3 | Bài tập..... | 71 |
| BÀI 8 : | MẢNG VÀ CHUỖI | 72 |
| 8.1 | Mục tiêu | 72 |
| 8.2 | Nội dung..... | 72 |
| 8.2.1 | Mảng..... | 72 |
| 8.2.1.1 | Cách khai báo mảng..... | 72 |
| 8.2.1.2 | Tham chiếu đến từng phần tử mảng | 72 |
| 8.2.1.3 | Nhập dữ liệu cho mảng | 73 |
| 8.2.1.4 | Đọc dữ liệu từ mảng | 73 |
| 8.2.1.5 | Sử dụng biến kiểu khác..... | 74 |
| 8.2.1.6 | Kỹ thuật Sentinel..... | 74 |
| 8.2.1.7 | Khởi tạo mảng..... | 75 |
| 8.2.1.8 | Khởi tạo mảng không bao hàm kích thước..... | 76 |
| 8.2.1.9 | Mảng nhiều chiều | 76 |
| 8.2.1.10 | Tham chiếu đến từng phần tử mảng 2 chiều | 76 |
| 8.2.1.11 | Nhập dữ liệu cho mảng 2 chiều | 77 |
| 8.2.1.12 | Đọc dữ liệu từ mảng 2 chiều | 77 |
| 8.2.1.13 | Sử dụng biến kiểu khác trong mảng 2 chiều..... | 78 |
| 8.2.1.14 | Khởi tạo mảng 2 chiều | 78 |
| 8.2.1.15 | Dùng mảng 1 chiều làm tham số cho hàm | 79 |
| 8.2.1.16 | Dùng mảng 2 chiều làm tham số cho hàm | 82 |
| 8.2.2 | Chuỗi..... | 84 |
| 8.2.2.1 | Cách khai báo chuỗi..... | 84 |
| 8.2.2.2 | Hàm nhập (gets), xuất (puts) chuỗi..... | 85 |
| 8.2.2.3 | Khởi tạo chuỗi..... | 86 |
| 8.2.2.4 | Mảng chuỗi..... | 86 |
| 8.3 | Bài tập..... | 87 |
| BÀI 9 : | CON TRỎ | 90 |
| 9.1 | Mục tiêu | 90 |
| 9.2 | Nội dung..... | 90 |
| 9.2.1 | Con trỏ? | 90 |
| 9.2.2 | Khái báo biến con trỏ | 90 |
| 9.2.3 | Truyền địa chỉ sang hàm | 91 |
| 9.2.4 | Con trỏ và mảng..... | 92 |
| 9.2.5 | Con trỏ trỏ đến mảng trong hàm | 92 |
| 9.2.6 | Con trỏ và chuỗi..... | 93 |
| 9.2.7 | Khởi tạo mảng con trỏ trỏ đến chuỗi | 94 |
| 9.2.8 | Xử lý con trỏ trỏ đến chuỗi | 95 |
| 9.2.9 | Con trỏ trỏ đến con trỏ..... | 97 |
| 9.3 | Bài tập..... | 98 |

| | |
|---|------------|
| BÀI 10 : CÁC KIỂU DỮ LIỆU TỰ TẠO | 99 |
| 10.1 Mục tiêu | 99 |
| 10.2 Nội dung..... | 99 |
| 10.2.1 Structure | 99 |
| 10.2.1.1 Khai báo kiểu structure | 99 |
| 10.2.1.2 Cách khai báo biến có kiểu structure | 99 |
| 10.2.1.3 Tham chiếu các phần tử trong structure..... | 99 |
| 10.2.1.4 Khởi tạo structure | 101 |
| 10.2.1.5 Structure lồng nhau..... | 102 |
| 10.2.1.6 Truyền structure sang hàm | 103 |
| 10.2.2 Enum | 105 |
| 10.2.2.1 Định nghĩa kiểu enum | 105 |
| 10.2.2.2 Cách khai báo biến có kiểu enum | 106 |
| 10.2.2.3 Sử dụng enum trong chương trình | 106 |
| 10.3 Bài tập..... | 108 |
| BÀI 11 : TẬP TIN | 109 |
| 11.1 Mục tiêu | 109 |
| 11.2 Nội dung..... | 109 |
| 11.2.1 Ví dụ ghi, đọc số nguyên..... | 109 |
| 11.2.2 Ghi, đọc mảng | 110 |
| 11.2.3 Ghi, đọc structure | 111 |
| 11.2.4 Các mode khác để mở tập tin | 112 |
| 11.2.5 Một số hàm thao tác trên file khác..... | 112 |
| 11.3 Bài tập..... | 113 |
| BÀI 12 : ĐỆ QUY | 114 |
| 12.1 Mục tiêu | 114 |
| 12.2 Nội dung..... | 114 |
| 12.3 Bài tập..... | 117 |
| BÀI 13 : TRÌNH SOẠN THẢO CỦA BORLAND C | 118 |
| 13.1 Mở tập tin soạn thảo mới..... | 118 |
| 13.2 Lưu tập tin..... | 118 |
| 13.2.1 Nếu là tập tin soạn thảo mới chưa lưu | 118 |
| 13.2.2 Nếu là tập tin đã lưu ít nhất 1 lần hoặc được mở bằng lệnh Open: | 118 |
| 13.3 Mở tập tin | 119 |
| 13.4 Các phím, tổ hợp phím thường dùng..... | 119 |
| 13.4.1 Các phím di chuyển con trỏ | 119 |

| | |
|---|------------|
| 13.4.2 Các phím thao tác trên khối..... | 120 |
| 13.4.3 Các thao tác xóa | 120 |
| 13.4.4 Các thao tác copy, di chuyển..... | 120 |
| 13.4.5 Các thao tác khác..... | 120 |
| 13.5 Ghi một khối ra đĩa | 121 |
| 13.6 Chèn nội dung file từ đĩa vào vị trí con trỏ | 121 |
| 13.7 Tìm kiếm văn bản trong nội dung soạn thảo | 121 |
| 13.8 Tìm và thay thế văn bản trong nội dung soạn thảo..... | 121 |
| 13.9 Sửa lỗi cú pháp..... | 122 |
| 13.10 Chạy từng bước | 122 |
| 13.11 Sử dụng Help (Giúp đỡ)..... | 122 |
| BÀI 14 : CÁC HỆ ĐẾM | 124 |
| 14.1 Khái niệm | 124 |
| 14.2 Quy tắc..... | 124 |
| 14.3 Chuyển đổi giữa các hệ | 125 |
| 14.3.1 Chuyển đổi giữa hệ 2 và hệ 10 | 125 |
| 14.3.2 Chuyển đổi giữa hệ 8 và hệ 10 | 126 |
| 14.3.3 Chuyển đổi giữa hệ 16 và hệ 10 | 126 |
| 14.3.4 Chuyển đổi giữa hệ 2 và hệ 16 | 127 |
| BÀI 15 : BIỂU THỨC VÀ PHÉP TOÁN | 128 |
| 15.1 Biểu thức..... | 128 |
| 15.2 Phép toán..... | 128 |
| 15.2.1 Phép toán số học..... | 128 |
| 15.2.2 Phép quan hệ | 128 |
| 15.2.3 Phép toán luận lý..... | 129 |
| 15.2.4 Phép toán trên bit (bitwise)..... | 129 |
| 15.2.5 Các phép toán khác..... | 130 |
| 15.2.6 Độ ưu tiên của các phép toán..... | 130 |
| 15.3 Bài tập..... | 130 |
| BÀI 16 : MỘT SỐ HÀM CHUẨN THƯỜNG DÙNG..... | 132 |
| 16.1 Các hàm chuyển đổi dữ liệu | 132 |
| 16.1.1 atof..... | 132 |
| 16.1.2 atoi..... | 132 |
| 16.1.3 itoa | 132 |
| 16.1.4 tolower..... | 132 |

| | |
|--|------------|
| 16.1.5 toupper..... | 132 |
| 16.2 Các hàm xử lý chuỗi ký tự..... | 133 |
| 16.2.1 strcat..... | 133 |
| 16.2.2 strcpy..... | 133 |
| 16.2.3 strcmp..... | 133 |
| 16.2.4 strcmpi..... | 133 |
| 16.2.5 strlwr..... | 133 |
| 16.2.6strupr..... | 133 |
| 16.2.7 strlen..... | 134 |
| 16.3 Các hàm toán học..... | 134 |
| 16.3.1 abs..... | 134 |
| 16.3.2 labs..... | 134 |
| 16.3.3 rand..... | 134 |
| 16.3.4 random..... | 134 |
| 16.3.5 pow..... | 134 |
| 16.3.6 sqrt..... | 134 |
| 16.4 Các hàm xử lý file..... | 135 |
| 16.4.1 rewind..... | 135 |
| 16.4.2 ftell..... | 135 |
| 16.4.3 fseek..... | 135 |



Bài 1 :

NGÔN NGỮ LẬP TRÌNH & PHƯƠNG PHÁP LẬP TRÌNH

1.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Ý nghĩa, các bước lập trình.
- Xác định dữ liệu vào, ra.
- Phân tích các bài toán đơn giản.
- Khái niệm so sánh, lặp.
- Thể hiện bài toán bằng lưu đồ.

1.2 Lý thuyết

1.2.1 Ngôn ngữ lập trình (Programming Language)

Phần này chúng ta sẽ tìm hiểu một số khái niệm căn bản về thuật toán, chương trình, ngôn ngữ lập trình. Thuật ngữ "thuật giải" và "thuật toán" dĩ nhiên có sự khác nhau song trong nhiều trường hợp chúng có cùng nghĩa.

1.2.1.1 Thuật giải (Algorithm)

Là một dãy các thao tác xác định trên một đối tượng, sao cho sau khi thực hiện một số hữu hạn các bước thì đạt được mục tiêu. Theo R.A.Kowalski thì bản chất của thuật giải:

Thuật giải = Logic + Điều khiển

* **Logic**: Đây là phần khá quan trọng, nó trả lời câu hỏi "Thuật giải làm gì, giải quyết vấn đề gì?", những yếu tố trong bài toán có quan hệ với nhau như thế nào v.v... Ở đây bao gồm những kiến thức chuyên môn mà bạn phải biết để có thể tiến hành giải bài toán.

Ví dụ 1: Để giải một bài toán tính diện tích hình cầu, mà bạn không còn nhớ công thức tính hình cầu thì bạn không thể viết chương trình cho máy để giải bài toán này được.

* **Điều khiển**: Thành phần này trả lời câu hỏi: giải thuật phải làm như thế nào?. Chính là cách thức tiến hành áp dụng thành phần logic để giải quyết vấn đề.

1.2.1.2 Chương trình (Program)

Là một tập hợp các mô tả, các phát biểu, nằm trong một hệ thống qui ước về ý nghĩa và thứ tự thực hiện, nhằm điều khiển máy tính làm việc. Theo Niklaus Wirth thì:

Chương trình = Thuật toán + Cấu trúc dữ liệu

Các thuật toán và chương trình đều có cấu trúc dựa trên **3 cấu trúc điều khiển cơ bản**:

* **Tuần tự** (Sequential): Các bước thực hiện tuần tự một cách chính xác từ trên xuống, mỗi bước chỉ thực hiện đúng một lần.

* **Chọn lọc** (Selection): Chọn 1 trong 2 hay nhiều thao tác để thực hiện.

* **Lặp lại** (Repetition): Một hay nhiều bước được thực hiện lặp lại một số lần.

Muốn trở thành lập trình viên chuyên nghiệp bạn hãy làm đúng trình tự để có thói quen tốt và thuận lợi sau này trên nhiều mặt của một người làm máy tính. Bạn hãy làm theo các bước sau:

Tìm, xây dựng thuật giải (trên giấy) → viết chương trình trên máy

→ dịch chương trình → chạy và thử chương trình

1.2.1.3 Ngôn ngữ lập trình (Programming language)

Ngôn ngữ lập trình là hệ thống các ký hiệu tuân theo các qui ước về ngữ pháp và ngữ nghĩa, dùng để xây dựng thành các chương trình cho máy tính.

Một chương trình được viết bằng một ngôn ngữ lập trình cụ thể (ví dụ Pascal, C...) gọi là chương trình nguồn, chương trình dịch làm nhiệm vụ dịch chương trình nguồn thành chương trình thực thi được trên máy tính.

1.2.2 Các bước lập trình

Bước 1: Phân tích vấn đề và xác định các đặc điểm. (xác định I-P-O)

Bước 2: Lập ra giải pháp. (đưa ra thuật giải)

Bước 3: Cài đặt. (viết chương trình)

Bước 4: Chạy thử chương trình. (dịch chương trình)

Bước 5: Kiểm chứng và hoàn thiện chương trình. (thử nghiệm bằng nhiều số liệu và đánh giá)

1.2.3 Kỹ thuật lập trình

1.2.3.1 I-P-O Cycle (Input-Process-Output Cycle) (Quy trình nhập-xử lý-xuất)

Quy trình xử lý cơ bản của máy tính gồm I-P-O.



Ví dụ 2: Xác định Input, Process, Output của việc làm 1 ly nước chanh nóng

Input : ly, đường, chanh, nước nóng, muỗng.

Process : - cho hỗn hợp đường, chanh, nước nóng vào ly.
- dùng muỗng khuấy đều.

Output : ly chanh nóng đã sẵn sàng để dùng.

Ví dụ 3: Xác định Input, Process, Output của chương trình tính tiền lương công nhân tháng 10/2002 biết rằng lương = lương căn bản * ngày công

Input : lương căn bản, ngày công

Process : nhân lương căn bản với ngày công

Output : lương

Ví dụ 4: Xác định Input, Process, Output của chương trình giải phương trình bậc nhất $ax + b = 0$

Input : hệ số a, b

Process : chia - b cho a

Output : nghiệm x

Ví dụ 5: Xác định Input, Process, Output của chương trình tìm số lớn nhất của 2 số a và b.

Input : a, b

Process : Nếu $a > b$ thì *Output* = a lớn nhất
Ngược lại *Output* = b lớn nhất

Bài tập

Xác định Input, Process, Output của các chương trình sau:

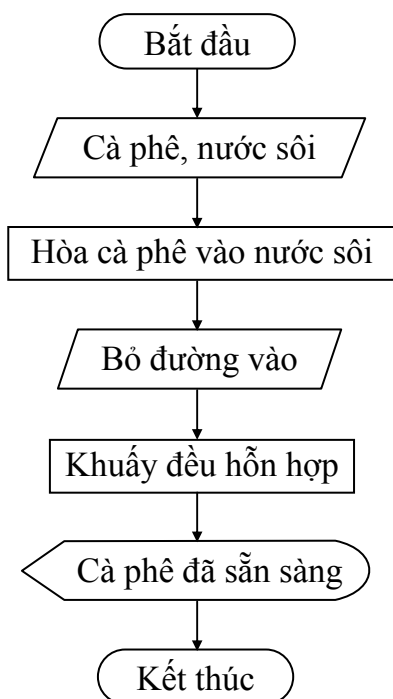
1. Đổi từ tiền VND sang tiền USD.
2. Tính điểm trung bình của học sinh gồm các môn Toán, Lý, Hóa.
3. Giải phương trình bậc 2: $ax^2 + bx + c = 0$
4. Đổi từ độ sang radian và đổi từ radian sang độ
(công thức $\alpha/\pi = a/180$, với α : radian, a: độ)
5. Kiểm tra 2 số a, b giống nhau hay khác nhau.

1.2.3.2 Sử dụng lưu đồ (Flowchart)

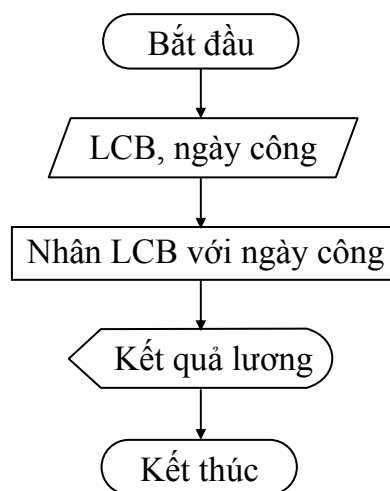
Để dễ hơn về quy trình xử lý, các nhà lập trình đưa ra dạng lưu đồ để minh họa từng bước quá trình xử lý một vấn đề (bài toán).

| Hình dạng (symbol) | Hành động (Activity) |
|--------------------|---|
| | Dữ liệu vào (Input) |
| | Xử lý (Process) |
| | Dữ liệu ra (Output) |
| | Quyết định (Decision), sử dụng điều kiện |
| | Luồng xử lý (Flow lines) |
| | Gọi CT con, hàm... (Procedure, Function...) |
| | Bắt đầu, kết thúc (Begin, End) |
| | Điểm ghép nối (Connector) |

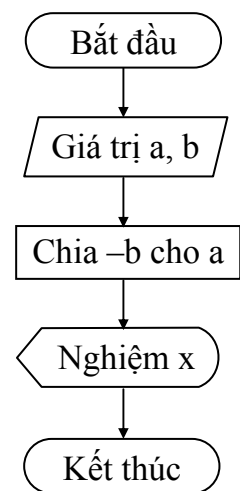
Ví dụ 6: Chuẩn bị cà phê



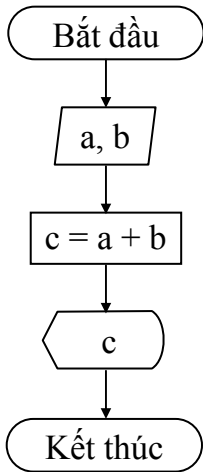
Ví dụ 7: Mô tả ví dụ 3



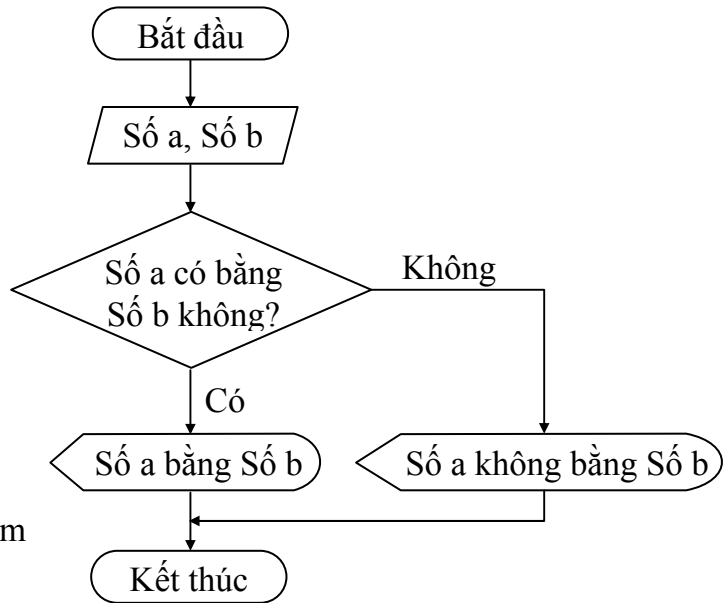
Ví dụ 8: Mô tả ví dụ 4



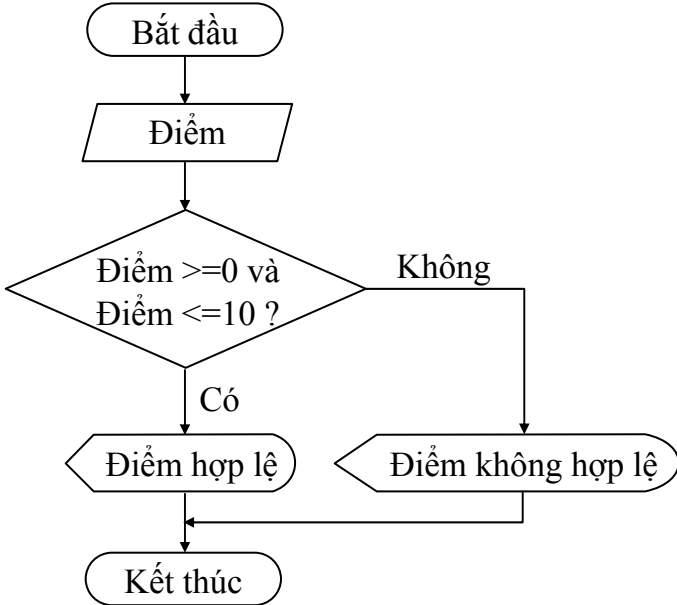
Ví dụ 9: Cộng 2 số



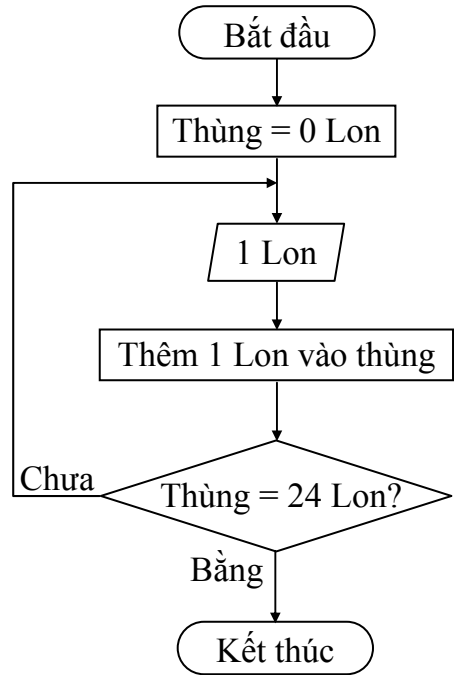
Ví dụ 10: so sánh 2 số



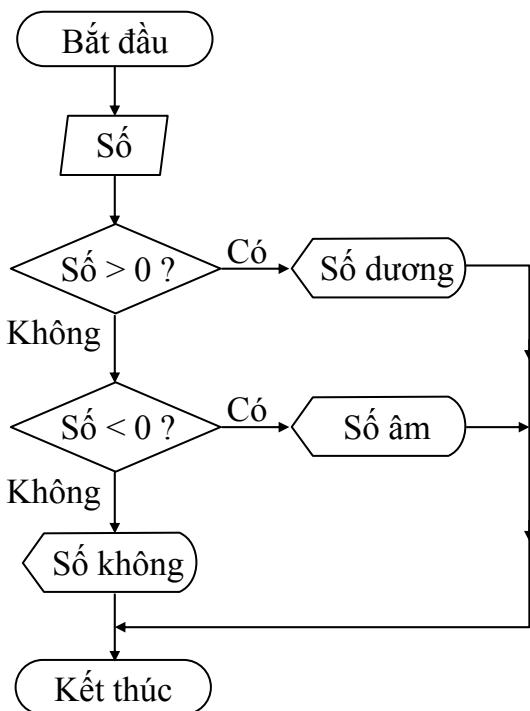
Ví dụ 11: Kiểm tra tính hợp lệ của điểm



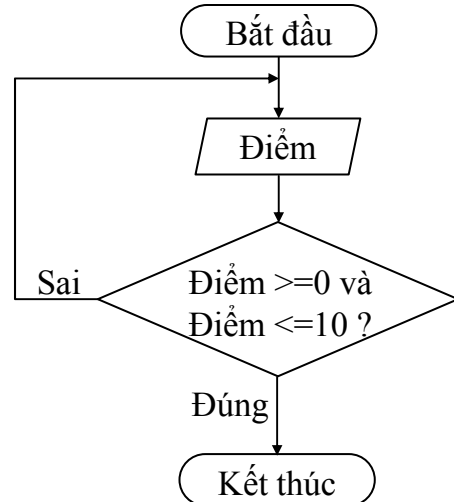
Ví dụ 12: Xếp lon vào thùng



Ví dụ 13: Kiểm tra loại số



Ví dụ 14: Kiểm tra tính hợp lệ của điểm



 **Bài tập**

Vẽ lưu đồ cho các chương trình sau:

1. Đổi từ tiền VND sang tiền USD.
2. Tính điểm trung bình của học sinh gồm các môn Toán, Lý, Hóa.
3. Giải phương trình bậc 2: $ax^2 + bx + c = 0$
4. Đổi từ độ sang radian và đổi từ radian sang độ
(công thức $\alpha/\pi = a/180$, với α : radian, a: độ)
5. Kiểm tra 2 số a, b giống nhau hay khác nhau.



Bài 2 :

LÀM QUEN LẬP TRÌNH C QUA CÁC VÍ DỤ ĐƠN GIẢN

2.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Ngôn ngữ C.
- Một số thao tác cơ bản của trình soạn thảo C.
- Cách lập trình trên C.
- Tiếp cận một số lệnh đơn giản thông qua các ví dụ.
- Nắm bắt được một số kỹ năng đơn giản.

2.2 Nội dung

2.2.1 Khởi động và thoát BorlandC

2.2.1.1 Khởi động

■ Nhập lệnh tại dấu nhắc DOS: **gõ BC ↵ (Enter)** (nếu đường dẫn đã được cài đặt bằng lệnh **path** trong đó có chứa đường dẫn đến thư mục chứa tập tin BC.EXE). Nếu đường dẫn chưa được cài đặt ta tìm xem thư mục BORLANDC nằm ở ổ đĩa nào. Sau đó ta gõ lệnh sau:

<ổ đĩa>:\BORLANDC\BIN\BC ↵ (Enter)

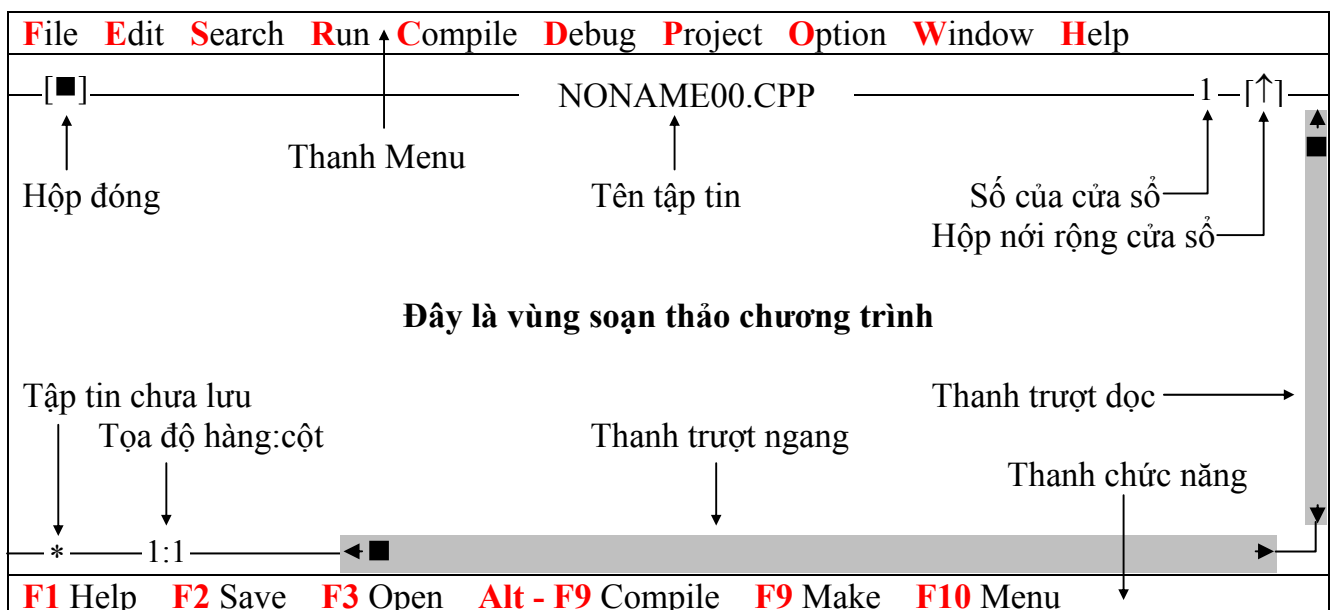
Nếu bạn muốn vừa khởi động BC vừa soạn thảo chương trình với một tập tin có tên do chúng ta đặt, thì gõ lệnh: **BC [đường dẫn]<tên file cần soạn thảo>**, nếu tên file cần soạn thảo đã có thì được nạp lên, nếu chưa có sẽ được tạo mới.

■ Khởi động tại Windows: Bạn vào menu Start, chọn Run, bạn gõ vào hộp Open 1 trong các dòng lệnh như nhập tại DOS. Hoặc bạn vào Window Explorer, chọn ổ đĩa chứa thư mục BORLANDC, vào thư mục BORLANDC, vào thư mục BIN, khởi động tập tin BC.EXE.

Ví dụ: Bạn gõ **D:\BORLANDC\BIN\BC E:\BAITAP_BC\VIDU1.CPP**

Câu lệnh trên có nghĩa khởi động BC và nạp tập tin VIDU1.CPP chứa trong thư mục BAITAP_BC trong ổ đĩa E. Nếu tập tin này không có sẽ được tạo mới.

Màn hình sau khi khởi động thành công



2.2.1.2 Thoát

- Ấn phím **F10** (kích hoạt Menu), chọn menu **File**, chọn **Quit**;
- Hoặc ấn tổ hợp phím **Alt – X**.

2.2.2 Các ví dụ đơn giản

2.2.2.1 Ví dụ 1

| | |
|------|---|
| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
| 1 | /* Chuong trinh in ra cau bai hoc C dau tien */ |
| 2 | #include <stdio.h> |
| 3 | |
| 4 | void main(void) |
| 5 | { |
| 6 | printf("Bai hoc C dau tien."); |
| 7 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

 **Kết quả in ra màn hình**

Bai hoc C dau tien. _

■ Dòng thứ 1: bắt đầu bằng **/*** và kết thúc bằng ***/** cho biết hàng này là hàng diễn giải (chú thích). Khi dịch và chạy chương trình, dòng này không được dịch và cũng không thi hành lệnh gì cả. Mục đích của việc ghi chú này giúp chương trình rõ ràng hơn. Sau này bạn đọc lại chương trình biết chương trình làm gì.


■ Dòng thứ 2: chứa phát biểu tiền xử lý **#include <stdio.h>**. Vì trong chương trình này ta sử dụng hàm thư viện của C là **printf**, do đó bạn cần phải có khai báo của hàm thư viện này để báo cho trình biên dịch C biết. **Nếu không khai báo chương trình sẽ báo lỗi.**

■ Dòng thứ 3: hàng trắng viết ra với ý đồ làm cho bảng chương trình thoáng, dễ đọc.

■ Dòng thứ 4: **void main(void)** là thành phần chính của mọi chương trình C (bạn có thể viết main() hoặc void main() hoặc main(void)). Tuy nhiên, bạn nên viết theo dạng **void main(void)** để chương trình rõ ràng hơn. Mọi chương trình C đều bắt đầu thi hành từ hàm main. Cặp dấu ngoặc () cho biết đây là khối hàm (function). Hàm void main(void) có từ khóa void đầu tiên cho biết hàm này không trả về giá trị, từ khóa void trong ngoặc đơn cho biết hàm này không nhận vào đối số.

■ Dòng thứ 5 và 7: cặp dấu ngoặc móc {} giới hạn thân của hàm. Thân hàm bắt đầu bằng dấu { và kết thúc bằng dấu }.

■ Dòng thứ 6: **printf("Bai hoc C dau tien.");**, chỉ thị cho máy in ra chuỗi ký tự nằm trong nháy kép ("). Hàng này được gọi là một câu lệnh, kết thúc một câu lệnh trong C phải là dấu chấm phẩy (;).

 **Chú ý:**

- ✓ Các từ include, stdio.h, void, main, printf phải viết bằng chữ thường.
- ✓ Chuỗi trong nháy kép cần in ra "Bạn có thể viết chữ HOA, thường tùy, ý".
- ✓ Kết thúc câu lệnh phải có dấu chấm phẩy.
- ✓ Kết thúc tên hàm không có dấu chấm phẩy hoặc bất cứ dấu gì.
- ✓ Ghi chú phải đặt trong cặp /* */.
- ✓ Thân hàm phải được bao bởi cặp { }.
- ✓ Các câu lệnh trong thân hàm phải viết thụt vào.



Bạn nhập đoạn chương trình trên vào máy. Dịch, chạy và quan sát kết quả.

Ctrl – F9: Dịch và chạy chương trình. Alt – F5: Xem màn hình kết quả.



Sau khi bạn nhập xong đoạn chương trình vào máy. Bạn Ấn và giữ phím Ctrl, gõ F9 để dịch và chạy chương trình. Khi đó bạn thấy chương trình chớp rất nhanh và không thấy kết quả gì cả. Bạn Ấn và giữ phím Alt, gõ F5 để xem kết quả, khi xem xong, bạn ấn phím bất kỳ để quay về màn hình soạn thảo chương trình.



Bây giờ bạn sửa lại dòng thứ 6 bằng câu lệnh `printf("Bai hoc C dau tien.\n");`, sau đó dịch và chạy lại chương trình, quan sát kết quả.

Kết quả in ra màn hình

Bai hoc C dau tien.

Ở dòng bạn vừa sửa có thêm `\n`, `\n` là ký hiệu xuống dòng sử dụng trong lệnh `printf`. Sau đây là một số ký hiệu khác.

+ Các kí tự điều khiển:

- `\n` : Nhảy xuống dòng kế tiếp canh về cột đầu tiên.
- `\t` : Canh cột tab ngang.
- `\r` : Nhảy về đầu hàng, không xuống hàng.
- `\a` : Tiếng kêu bip.

+ Các kí tự đặc biệt:

- `\\` : In ra dấu `\`
- `\"` : In ra dấu `"`
- `'` : In ra dấu `'`



Bây giờ bạn sửa lại dòng thứ 6 bằng câu lệnh `printf("\tBai hoc C dau tien.\a\n");`, sau đó dịch và chạy lại chương trình, quan sát kết quả.

Kết quả in ra màn hình

Bai hoc C dau tien.

Khi chạy chương trình bạn nghe tiếng bip phát ra từ loa.



Mỗi khi chạy chương trình bạn thấy rất bất tiện trong việc xem kết quả phải ấn tổ hợp phím `Alt – F5`. Để khắc phục tình trạng này bạn sửa lại chương trình như sau:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | <code>/* Chuong trinh in ra cau bai hoc C dau tien */</code> |
| 2 | <code>#include <stdio.h></code> |
| 3 | <code>#include <conio.h></code> |
| 4 | |
| 5 | <code>void main(void)</code> |
| 6 | <code>{</code> |
| 7 | <code>printf("\t\tBai hoc C \rdau tien.\n");</code> |
| 8 | <code>getch();</code> |
| 9 | <code>}</code> |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |


 **Kết quả in ra màn hình**

```
dau tien.      Bai hoc C
-

```

■ Dòng thứ 3: chứa phát biểu tiên xử lý **#include <conio.h>**. Vì trong chương trình này ta sử dụng hàm thư viện của C là **getch**, do đó bạn cần phải có khai báo của hàm thư viện này để báo cho trình biên dịch C biết. **Nếu không khai báo chương trình sẽ báo lỗi.**

■ Dòng thứ 8: **getch()**; chờ nhận 1 ký tự bất kỳ từ bàn phím, nhưng không in ra màn hình. Vì thế ta sử dụng hàm này để khi chạy chương trình xong sẽ dừng lại ở màn hình kết quả, sau đó ta ấn phím bất kỳ sẽ quay lại màn hình soạn thảo.

 Bạn nhập đoạn chương trình trên vào máy. Dịch, chạy và quan sát kết quả.

2.2.2.2 Ví dụ 2

| Dòng | File | Edit | Search | Run | Compile | Debug | Project | Option | Window | Help | |
|------|---|---------|--------|-----------------|---------|-----------------|---------|------------------|--------|---------|----------|
| 1 | /* Chuong trinh nhap va in ra man hinh gia tri bien*/ | | | | | | | | | | |
| 2 | #include <stdio.h> | | | | | | | | | | |
| 3 | #include <conio.h> | | | | | | | | | | |
| 4 | | | | | | | | | | | |
| 5 | void main(void) | | | | | | | | | | |
| 6 | { | | | | | | | | | | |
| 7 | int i; | | | | | | | | | | |
| 8 | printf("Nhap vao mot so: "); | | | | | | | | | | |
| 9 | scanf("%d", &i); | | | | | | | | | | |
| 10 | printf("So ban vua nhap la: %d.\n", i); | | | | | | | | | | |
| 11 | getch(); | | | | | | | | | | |
| 12 | } | | | | | | | | | | |
| | | F1 Help | | Alt-F8 Next Msg | | Alt-F7 Prev Msg | | Alt - F9 Compile | | F9 Make | F10 Menu |

 **Kết quả in ra màn hình**


```
Nhap vao mot so: 15
So ban vua nhap la: 15.
-

```

■ Dòng thứ 7: **int i**; là lệnh khai báo, mẫu tự i gọi là tên biến. Biến là một vị trí trong bộ nhớ dùng lưu trữ giá trị nào đó mà chương trình sẽ lấy để sử dụng. Mỗi biến phải thuộc một kiểu dữ liệu. Trong trường hợp này ta sử dụng biến i kiểu số nguyên (integer) viết tắt là int.


■ Dòng thứ 9: **scanf("%d", &i)**. Sử dụng hàm scanf để nhận từ người sử dụng một trị nào đó. Hàm scanf trên có 2 đối mục. Đối mục **"%d"** được gọi là chuỗi định dạng, cho biết loại dữ kiện mà người sử dụng sẽ nhập vào. Chẳng hạn, ở đây phải nhập vào là số nguyên. Đối mục thứ 2 **&i** có dấu & đi đầu gọi là address operator, dấu & phối hợp với tên biến cho hàm scanf biến đem trị gõ từ bàn phím lưu vào biến i.

■ Dòng thứ 10: **printf("So ban vua nhap la: %d.\n", i)**; Hàm này có 2 đối mục. Đối mục thứ nhất là một chuỗi định dạng có chứa chuỗi văn bản **So ban vua nhap la:** và **%d** (ký hiệu khai báo chuyển đổi dạng thức) cho biết số nguyên sẽ được in ra. Đối mục thứ 2 là i cho biết giá trị lấy từ biến i để in ra màn hình.

 Bạn nhập đoạn chương trình trên vào máy. Dịch, chạy và quan sát kết quả.

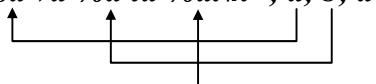
2.2.2.3 Ví dụ 3

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Chuong trinh nhap vao 2 so a, b in ra tong*/ |
| 2 | #include <stdio.h> |
| 3 | #include <conio.h> |
| 4 | |
| 5 | void main(void) |
| 6 | { |
| 7 | int a, b; |
| 8 | printf("Nhap vao so a: "); |
| 9 | scanf("%d", &a); |
| 10 | printf("Nhap vao so b: "); |
| 11 | scanf("%d", &b); |
| 12 | printf("Tong cua 2 so %d va %d la %d.\n", a, b, a+b); |
| 13 | getch(); |
| 14 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

 **Kết quả in ra màn hình**

```
Nhap vao so a: 4
Nhap vao so b: 14
Tong cua 2 so 4 va 14 la 18.
_
```

■ Dòng thứ 12: `printf("Tong cua 2 so %d va %d la %d.\n", a, b, a+b);`



Bạn nhập đoạn chương trình trên vào máy. Dịch, chạy và quan sát kết quả.

2.2.2.4 Ví dụ 4

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Chuong trinh nhap vao ban kinh hinh tron. Tinh dien tich */ |
| 2 | #include <stdio.h> |
| 3 | #include <conio.h> |
| 4 | |
| 5 | #define PI 3.14 |
| 6 | |
| 7 | void main(void) |
| 8 | { |
| 9 | float fR; |
| 10 | printf("Nhap vao ban kinh hinh tron: "); |
| 11 | scanf("%f", &fR); |
| 12 | printf("Dien tich hinh tron: %.2f.\n", 2*PI*fR); |
| 13 | getch(); |
| 14 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

👉 Kết quả in ra màn hình

Nhap vao ban kinh hinh tron: 1

Dien tich hinh tron: 6.28

—

■ Dòng thứ 5: ***#define PI 3.14***, dùng chỉ thị define để định nghĩa hằng số PI có giá trị 3.14. *Trước define phải có dấu # và cuối dòng không có dấu chấm phẩy.*

■ Dòng thứ 12: ***printf("Dien tich hinh tron: %.2f\n", 2*PI*fR);***. Hàm này có 2 đối mục. Đối mục thứ nhất là một chuỗi định dạng có chứa chuỗi văn bản ***Dien tich hinh tron:*** và ***%.2f*** (ký hiệu khai báo chuyển đổi dạng thức) cho biết dạng số chấm động sẽ được in ra, trong đó ***.2*** nghĩa là in ra với 2 số lẻ. Đối mục thứ 2 là biểu thức hằng ***2*PI*fR***;



Bạn nhập đoạn chương trình trên vào máy. Dịch, chạy và quan sát kết quả.



Bài 3 :

CÁC THÀNH PHẦN TRONG NGÔN NGỮ C

3.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Khái niệm từ khóa
- Các kiểu dữ liệu
- Cách ghi chú
- Đặt tên biến
- Khai báo biến.
- Phạm vi sử dụng biến.

3.2 Nội dung

3.2.1 Từ khóa

Từ khóa là từ có ý nghĩa xác định dùng để khai báo dữ liệu, viết câu lệnh... Trong C có các từ khóa sau:

| | | | | | | | |
|-------|----------|--------|------|-----------|--------|----------|----------|
| asm | const | else | for | interrupt | return | sizeof | void |
| break | continue | enum | goto | long | short | switch | volatile |
| case | default | extern | huge | near | static | typedef | while |
| cdecl | do | far | if | pascal | struct | union | |
| char | double | float | int | register | signed | unsigned | |

☞ Các từ khóa phải viết bằng **chữ thường**

3.2.2 Tên

Khái niệm tên rất quan trọng trong quá trình lập trình, nó không những thể hiện rõ ý nghĩa trong chương trình mà còn dùng để xác định các đại lượng khác nhau khi thực hiện chương trình. Tên thường được đặt cho hằng, biến, mảng, con trỏ, nhãn... Chiều dài tối đa của tên là 32 ký tự.

Tên biến hợp lệ là một chuỗi ký tự liên tục gồm: **Ký tự chữ, số và dấu gạch dưới**. Ký tự đầu của tên phải là **chữ hoặc dấu gạch dưới**. Khi đặt tên không được đặt trùng với các từ khóa.

Ví dụ 1 :

■ Các tên đúng: delta, a_1, Num_ODD, Case

■ Các tên sai:

| | |
|---------|-----------------------------|
| 3a_1 | (ký tự đầu là số) |
| num-odd | (sử dụng dấu gạch ngang) |
| int | (đặt tên trùng với từ khóa) |
| del ta | (có khoảng trắng) |
| f(x) | (có dấu ngoặc tròn) |

Lưu ý: Trong C, tên phân biệt chữ hoa, chữ thường

Ví dụ 2 : number khác Number

case khác Case

(case là từ khóa, do đó bạn đặt tên là Case vẫn đúng)

3.2.3 Kiểu dữ liệu

Có 4 kiểu dữ liệu cơ bản trong C là: char, int, float, double.

| TT | Kiểu dữ liệu (Type) | Kích thước (Length) | Miền giá trị (Range) |
|----|------------------------|------------------------|--|
| 1 | unsigned char | 1 byte | 0 đến 255 |
| 2 | char | 1 byte | - 128 đến 127 |
| 3 | enum | 2 bytes | - 32,768 đến 32,767 |
| 4 | unsigned int | 2 bytes | 0 đến 65,535 |
| 5 | short int | 2 bytes | - 32,768 đến 32,767 |
| 6 | int | 2 bytes | - 32,768 đến 32,767 |
| 7 | unsigned long | 4 bytes | 0 đến 4,294,967,295 |
| 8 | long | 4 bytes | - 2,147,483,648 đến 2,147,483,647 |
| 9 | float | 4 bytes | $3.4 * 10^{-38}$ đến $3.4 * 10^{38}$ |
| 10 | double | 8 bytes | $1.7 * 10^{-308}$ đến $1.7 * 10^{308}$ |
| 11 | long double | 10 bytes | $3.4 * 10^{-4932}$ đến $1.1 * 10^{4932}$ |

3.2.4 Ghi chú

Trong khi lập trình cần phải ghi chú để giải thích các biến, hằng, thao tác xử lý giúp cho chương trình rõ ràng dễ hiểu, dễ nhớ, dễ sửa chữa và để người khác đọc vào dễ hiểu. Trong C có các ghi chú sau: // hoặc /* nội dung ghi chú */

Ví dụ 3 :

```
void main()
{
    int a, b;           //khai bao bien t kieu int
    a = 1;             //gan 1 cho a
    b =3;              //gan 3 cho b
    /* thuat toan tim so lon nhat la
       neu a lon hon b thi a lon nhat
       nguoc lai b lon nhat */
    if (a > b) printf("max: %d", a);
    else printf("max: %d", b);
}
```

Khi biên dịch chương trình, C gặp cặp dấu ghi chú sẽ không dịch ra ngôn ngữ máy.

Tóm lại, đối với ghi chú dạng // dùng để ghi chú một hàng và dạng /* */ có thể ghi chú một hàng hoặc nhiều hàng.

3.2.5 Khai báo biến

3.2.5.1 Tên biến

Cách đặt tên biến như mục 2.

3.2.5.2 Khai báo biến

■ Cú pháp

Kiểu dữ liệu *Danh sách tên biến*;

☞ Kiểu dữ liệu: 1 trong các kiểu ở mục 3

Danh sách tên biến: gồm các tên biến có cùng kiểu dữ liệu, mỗi tên biến cách nhau dấu phẩy (,), cuối cùng là dấu chấm phẩy (;).

☞ Khi khai báo biến nên đặt tên biến theo **quy tắc Hungarian Notation**

Ví dụ 4 :

```
int ituoi;           //khai báo biến ituoi có kiểu int
float fTrongluong;   //khai báo biến fTrongluong có kiểu long
char ckitu1, ckitu2; //khai báo biến ckitu1, ckitu2 có kiểu char
```

Các biến khai báo trên theo quy tắc Hungarian Notation. Nghĩa là biến `itoui` là kiểu `int`, bạn thêm chữ `i` (kí tự đầu của kiểu) vào đầu tên biến **tuoi** để trong quá trình lập trình hoặc sau này xem lại, sửa chữa... bạn dễ dàng nhận ra biến **itoui** có kiểu `int` mà không cần phải di chuyển đến phần khai báo mới biết kiểu của biến này. Tương tự cho biến **fTrongluong**, bạn nhìn vào là biết ngay biến này có kiểu `float`.

3.2.5.3 Vừa khai báo vừa khởi gán

Có thể kết hợp việc khai báo với toán tử gán để biến nhận ngay giá trị cùng lúc với khai báo.

Ví dụ 5 :

- **Khai báo trước, gán giá trị sau:**

```
void main()
{
    int a, b, c;
    a = 1;
    b = 2;
    c = 5;
    ...
}
```

- **Vừa khai báo vừa gán giá trị:**

```
void main()
{
    int a = 1, b = 2, c = 5;
    ...
}
```

3.2.5.4 Phạm vi của biến

Khi lập trình, bạn phải nắm rõ phạm vi của biến. Nếu khai báo và sử dụng không đúng, không rõ ràng sẽ dẫn đến sai sót khó kiểm soát được, vì vậy bạn cần phải xác định đúng vị trí, phạm vi sử dụng biến trước khi sử dụng biến.

- Khai báo biến ngoài (biến toàn cục): Vị trí biến đặt bên ngoài tất cả các hàm, cấu trúc... Các biến này có ảnh hưởng đến toàn bộ chương trình. Chu trình sống của nó là bắt đầu chạy chương trình đến lúc kết thúc chương trình.

- Khai báo biến trong (biến cục bộ): Vị trí biến đặt bên trong hàm, cấu trúc.... Chỉ ảnh hưởng nội bộ bên trong hàm, cấu trúc đó.... Chu trình sống của nó bắt đầu từ lúc hàm, cấu trúc được gọi thực hiện đến lúc thực hiện xong.



Bài 4 :

NHẬP / XUẤT DỮ LIỆU

4.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Ý nghĩa, cách sử dụng hàm printf, scanf
- Sử dụng khuôn dạng, ký tự đặc biệt, ký tự điều khiển trong printf, scanf.

4.2 Nội dung

4.2.1 Hàm printf

Kết xuất dữ liệu được định dạng.

■ Cú pháp

printf ("chuỗi định dạng"[, đối mục 1, đối mục 2,...]);

☞ Khi sử dụng hàm phải khai báo tiền xử lý **#include <stdio.h>**

- printf: tên hàm, **phải viết bằng chữ thường**.
- đối mục 1,...: là các mục dữ kiện cần in ra màn hình. Các đối mục này có thể là biến, hằng hoặc biểu thức phải được định trị trước khi in ra.
- chuỗi định dạng: được đặt trong cặp nháy kép (" "), gồm 3 loại:
 - + Đối với chuỗi kí tự ghi như thế nào in ra giống như vậy.
 - + Đối với những kí tự chuyên đổi dạng thức cho phép kết xuất giá trị của các đối mục ra màn hình tạm gọi là mã định dạng. Sau đây là các dấu mô tả định dạng:

%c : Kí tự đơn

%s : Chuỗi

%d : Số nguyên thập phân có dấu

%f : Số chấm động (ký hiệu thập phân)

%e : Số chấm động (ký hiệu có số mũ)

%g : Số chấm động (%f hay %g)

%x : Số nguyên thập phân không dấu

%u : Số nguyên hex không dấu

%o : Số nguyên bát phân không dấu

l : Tiền tố dùng kèm với %d, %u, %x, %o để chỉ số nguyên dài (ví dụ %ld)

+ Các ký tự điều khiển và ký tự đặc biệt

\n : Nhảy xuống dòng kế tiếp canh về cột đầu tiên.

\t : Canh cột tab ngang.

\r : Nhảy về đầu hàng, không xuống hàng.

\a : Tiếng kêu bip.

\\ : In ra dấu \

\" : In ra dấu "

\' : In ra dấu '

%%: In ra dấu %

Ví dụ 1: printf("Bai hoc C dau tien. \n");

└─> ký tự điều khiển
└─> chuỗi ký tự

Kết quả in ra màn hình

Bai hoc C dau tien.

```

Ví dụ 2: printf("Ma dinh dang \\\" in ra dau \".\n");

```

Kết quả in ra màn hình

Ma dinh dang \" in ra dau \".

```

Ví dụ 3: giả sử biến i có giá trị = 5
                xuất giá trị biến i
printf("So ban vua nhap la: %d .\n", i);

```

Kết quả in ra màn hình

So ban vua nhap la: 5.

```

Ví dụ 4: giả sử biến a có giá trị = 7 và b có giá trị = 4
                xuất giá trị biểu thức a+b
                xuất giá trị biến b
                xuất giá trị biến a
printf("Tong cua 2 so %d va %d la %d .\n", a, b, a+b);

```

Kết quả in ra màn hình

Tong cua 2 so 7 va 4 la 11.

```

Ví dụ 5: sửa lại ví dụ 4
                printf("Tong cua 2 so %5d va %3d la %1d .\n", a, b, a+b);

```

👉 Kết quả in ra màn hình

| |
|---|
| Tong cua 2 so <u>7</u> va <u>4</u> la <u>11</u> . |
| — |

→ 2 kí tự (mặc dù định dạng là 1)
 → 3 kí tự
 → 5 kí tự

Ví dụ 6: sửa lại ví dụ 5

```
printf("Tong cua 2 so %-5d va %-3d la %-1d . \n", a, b, a+b);
```

👉 Kết quả in ra màn hình

| |
|---|
| Tong cua 2 so <u>7</u> va <u>4</u> la <u>11</u> . |
| — |

→ 2 kí tự (mặc dù định dạng là 1)
 → 3 kí tự
 → 5 kí tự

👉 Dấu trừ trước bề rộng trường sẽ kéo kết quả sang trái

Ví dụ 7: sửa lại ví dụ 4

```
printf("Tong cua 2 so %02d va %02d la %04d . \n", a, b, a+b);
```

👉 Kết quả in ra màn hình

| |
|---|
| Tong cua 2 so <u>07</u> va <u>04</u> la <u>0011</u> . |
| — |

→ thêm 2 số 0 trước -> đủ 4 kí tự
 → thêm 1 số 0 trước -> đủ 2 kí tự
 → thêm 1 số 0 trước -> đủ 2 kí tự

Ví dụ 8: giả sử int a = 6, b = 1234, c = 62

```
printf("%7d%7d%7d.\n", a, b, c);
printf("%7d%7d%7d.\n", 165, 2, 965);
```

👉 Kết quả in ra màn hình

| | |
|-----------|-------------------------------------|
| 6 1234 62 | Số canh về bên phải bề rộng trường. |
| 165 2 965 | |
| — | |

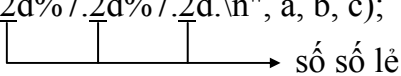
```
printf("%-7d%-7d%-7d.\n", a, b, c);
printf("%-7d%-7d%-7d.\n", 165, 2, 965);
```

👉 Kết quả in ra màn hình

| | |
|-----------|-------------------------------------|
| 6 1234 62 | Số canh về bên trái bề rộng trường. |
| 165 2 965 | |
| — | |

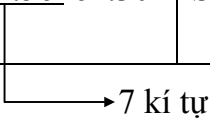
Ví dụ 9: giả sử float a = 6.4, b = 1234.56, c = 62.3

```
printf("%7.2d%7.2d%7.2d\n", a, b, c);
```



☞ **Kết quả in ra màn hình**

| | | | |
|------|---------|-------|-------------------------------------|
| 6.40 | 1234.56 | 62.30 | Số canh về bên phải bề rộng trường. |
| — | | | |



☞ **Bề rộng trường bao gồm: phần nguyên, phần lẻ và dấu chấm động**

Ví dụ 10: giả sử float a = 6.4, b = 1234.55, c = 62.34

```
printf("%10.1d%10.1d%10.1d\n", a, b, c);
printf("%10.1d%10.1d%10.1d\n", 165, 2, 965);
```

☞ **Kết quả in ra màn hình**

| | | | |
|-------|--------|-------|-------------------------------------|
| 6.4 | 1234.6 | 62.3 | Số canh về bên phải bề rộng trường. |
| 165.0 | 2.0 | 965.0 | |

```
printf("%-10.2d%-10.2d%-10.2d\n", a, b, c);
printf("%-10.2d%-10.2d%-10.2d\n", 165, 2, 965);
```

☞ **Kết quả in ra màn hình**

| | | | |
|--------|---------|--------|-------------------------------------|
| 6.40 | 1234.55 | 62.34 | Số canh về bên trái bề rộng trường. |
| 165.00 | 2.00 | 965.00 | |

4.2.2 Hàm scanf

Định dạng khi nhập liệu.

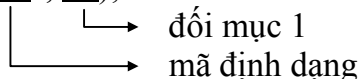
■ **Cú pháp**

```
scanf ("chuỗi định dạng" [, đối mục 1, đối mục 2, ...]);
```

☞ Khi sử dụng hàm phải khai báo tiền xử lý `#include <stdio.h>`

- scanf: tên hàm, **phải viết bằng chữ thường**.
- khung định dạng: được đặt trong cặp nháy kép (" ") là hình ảnh dạng dữ liệu nhập vào.
- đối mục 1, ..., là danh sách các đối mục cách nhau bởi dấu phẩy, mỗi đối mục sẽ tiếp nhận giá trị nhập vào.

Ví dụ 11: `scanf("%d", &i);`



☞ Nhập vào 12abc, biến i chỉ nhận giá trị 12. Nhập 3.4 chỉ nhận giá trị 3.

Ví dụ 12: `scanf("%d%d", &a, &b);`

☞ Nhập vào 2 số a, b phải cách nhau bằng **khoảng trắng** hoặc **enter**.

Ví dụ 13: `scanf("%d/%d/%d", &ngay, &thang, &nam);`

☞ Nhập vào ngày, tháng, năm theo dạng ngày/thang/nam (20/12/2002)

Ví dụ 14: `scanf("%d%*c%d%*c%d", &ngay, &thang, &nam);`

☞ Nhập vào ngày, tháng, năm với dấu phân cách /, -, ...; ngoại trừ số.

Ví dụ 15: `scanf("%2d%2d%4d", &ngay, &thang, &nam);`

☞ Nhập vào ngày, tháng, năm theo dạng dd/mm/yyyy.

4.3 Bài tập

1. *Viết chương trình đổi một số nguyên hệ 10 sang hệ 2.*
2. *Viết chương trình đổi một số nguyên hệ 10 sang hệ 16.*
3. *Viết chương trình đọc và 2 số nguyên và in ra kết quả của phép (+), phép trừ (-), phép nhân (*), phép chia (/). Nhận xét kết quả chia 2 số nguyên.*
4. *Viết chương trình nhập vào bán kính hình cầu, tính và in ra diện tích, thể tích của hình cầu đó.*

Hướng dẫn: $S = 4\pi R^2$ và $V = (4/3)\pi R^3$.

5. *Viết chương trình nhập vào một số a bất kỳ và in ra giá trị bình phương (a^2), lập phương (a^3) của a và giá trị a^4 .*
6. *Viết chương trình đọc từ bàn phím 3 số nguyên biểu diễn ngày, tháng, năm và xuất ra màn hình dưới dạng "ngày/thang/nam" (chỉ lấy 2 số cuối của năm).*
7. *Viết chương trình nhập vào số giây từ 0 đến 86399, đổi số giây nhập vào thành dạng "gio:phut:giay", mỗi thành phần là một số nguyên có 2 chữ số.*
Ví dụ: 02:11:05



Bài 5 :

CẤU TRÚC Rẽ NHÁNH CÓ ĐIỀU KIỆN

(Cấu trúc chọn)

5.1 Mục tiêu

- Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:
- Ý nghĩa lệnh, khối lệnh.
 - Cú pháp, ý nghĩa, cách sử dụng lệnh if, lệnh switch.
 - Một số bài toán sử dụng lệnh if, switch thông qua các ví dụ.
 - So sánh, đánh giá một số bài toán sử dụng lệnh if hoặc switch.
 - Cách sử dụng các cấu trúc lồng nhau.

5.2 Nội dung

5.2.1 Lệnh và khối lệnh

5.2.1.1 Lệnh

Là một tác vụ, biểu thức, hàm, cấu trúc điều khiển...

Ví dụ 1:

```
x = x + 2;
printf("Day la mot lenh\n");
```

5.2.1.2 Khối lệnh

Là một dãy các câu lệnh được bọc bởi cặp dấu { }, các lệnh trong khối lệnh phải viết thụt vô 1 tab so với cặp dấu { }

Ví dụ 2:

```
{ //đau khoi
  a = 5;
  b = 6;
  printf("Tong %d + %d = %d", a, b, a+b);
} //cuoi khoi
```

} viết thụt vô 1 tab so với cặp { }

☞ **Quên dùng cặp dấu { } bao bọc khi sử dụng khối lệnh, hoặc mở dấu { và quên đóng dấu }**

5.2.2 Lệnh if

Câu lệnh if cho phép lựa chọn một trong hai nhánh tùy thuộc vào giá trị của biểu thức luận lý là đúng (true) hay sai (false) hoặc khác không hay bằng không.

5.2.2.1 Dạng 1 (if thiếu)

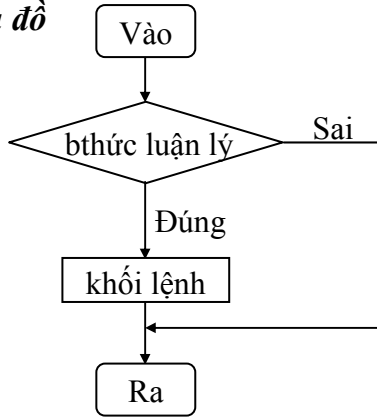
Quyết định sẽ thực hiện hay không một khối lệnh.

- **Cú pháp lệnh**

**if (biểu thức luận lý)
khối lệnh;**

- ☞ từ khóa **if** phải viết bằng chữ thường
- ☞ kết quả của **biểu thức luận lý** phải là **đúng (≠ 0)** hoặc **sai (= 0)**

• Lưu đồ



☞ nếu **biểu thức luận lý** đúng thì thực hiện khối lệnh và thoát khỏi if, ngược lại không làm gì cả và thoát khỏi if.

☞ Nếu **khối lệnh** bao gồm từ 2 lệnh trở lên thì phải đặt trong dấu { }

Diễn giải:

+ Khối lệnh là một lệnh ta viết lệnh if như sau:

```
if (biểu thức luận lý)
    lệnh;
```

+ Khối lệnh bao gồm nhiều lệnh: lệnh 1, lệnh 2..., ta viết lệnh if như sau:

```
if (biểu thức luận lý)
{
    lệnh 1;
    lệnh 2;
    ...
}
```

☞ Không đặt dấu chấm phẩy sau câu lệnh if.

Ví dụ: `if(biểu thức luận lý);`

→ trình biên dịch không báo lỗi nhưng khối lệnh không được thực hiện cho dù điều kiện đúng hay sai.

Ví dụ 3: Viết chương trình nhập vào 2 số nguyên a, b. Tìm và in ra số lớn nhất.

a. Phác họa lời giải

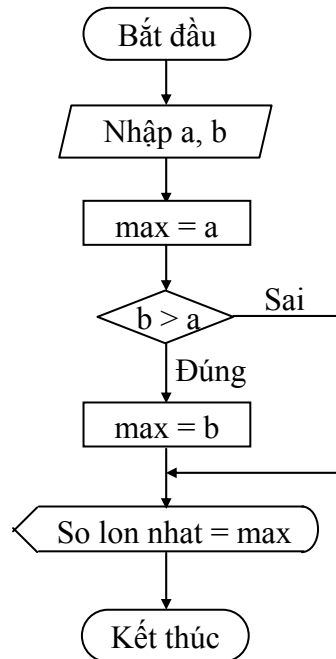
Trước tiên ta cho giá trị **a là giá trị lớn nhất bằng cách gán a cho max** (max là biến được khai báo cùng kiểu dữ liệu với a, b). Sau đó so sánh b với a, **nếu b lớn hơn a ta gán b cho max** và cuối cùng ta **được kết quả max là giá trị lớn nhất**.

b. Mô tả quy trình xử lý (giải thuật)

| Ngôn ngữ tự nhiên | Ngôn ngữ C |
|--|---|
| - Khai báo 3 biến a, b, max kiểu số nguyên | - int ia, ib, imax; |
| - Nhập vào giá trị a | - printf("Nhap vao so a: "); scanf("%d", &ia); |
| - Nhập vào giá trị b | - printf("Nhap vao so b: "); scanf("%d", &ib); |
| - Gán a cho max | - imax = ia; |
| - Nếu b > a thì gán b cho max | - if (ib > ia) imax = ib; |
| - In ra kết quả max | - printf("So lon nhat = %d.\n", imax); |

☞ **Biểu thức luận lý phải đặt trong cặp dấu (). if ib > ia → báo lỗi**

c. Mô tả bằng lưu đồ



d. Viết chương trình

```

File Edit Search Run Compile Debug Project Option Window Help
/* Chương trình tìm số lớn nhất từ 2 số nguyên a, b */
#include <stdio.h>
#include <conio.h>

void main(void)
{
    int ia, ib, imax;
    printf("Nhập vào số a: ");
    scanf("%d", &ia);
    printf("Nhập vào số b: ");
    scanf("%d", &ib);
    imax = ia;
    if (ib>ia)
        imax = ib;
    printf("Số lớn nhất = %d.\n", imax);
    getch();
}
F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu
  
```

☞ Kết quả in ra màn hình

| | |
|--|---|
| Nhập vào số a : 10 Nhập vào số b : 8 Số lớn nhất = 10. | Cho chạy lại chương trình và thử lại với: a = 7, b = 9 a = 5, b = 5 Quan sát và nhận xét kết quả |
|--|---|

Ví dụ 4: Viết chương trình nhập vào 2 số nguyên a, b. Nếu a lớn hơn b thì hoán đổi giá trị a và b, ngược lại không hoán đổi. In ra giá trị a, b.

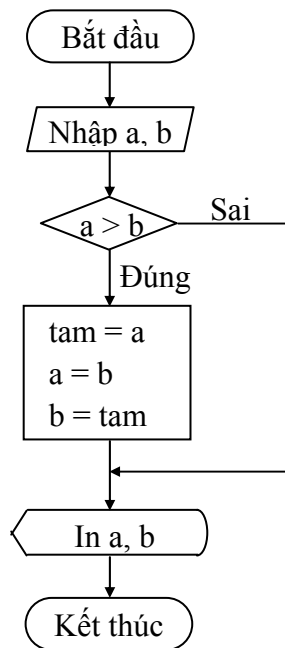
a. Phác họa lời giải

Nếu giá trị a lớn hơn giá trị b, bạn phải hoán chuyển 2 giá trị này cho nhau (nghĩa là a sẽ mang giá trị b và b mang giá trị a) bằng cách đem **giá trị a gởi (gán) cho biến tam** (biến tam được khai báo theo kiểu dữ liệu của a, b), kế đến bạn **gán giá trị b cho a** và cuối cùng bạn **gán giá trị tam cho b**, rồi in ra a, b.

b. Mô tả quy trình thực hiện (giải thuật)

| Ngôn ngữ tự nhiên | Ngôn ngữ C |
|---|--|
| - Khai báo 3 biến a, b, tam kiểu số nguyên - Nhập vào giá trị a - Nhập vào giá trị b - Nếu $a > b$ thì tam = a; a = b; b = tam; - In ra a, b | - int ia, ib, itam; - printf("Nhap vao so a: "); scanf("%d", &ia); - printf("Nhap vao so b: "); scanf("%d", &ib); - if (ia > ib) { itam = ia; ia = ib; ib = itam; } - printf("%d, %d\n", ia, ib); |

c. Mô tả bằng lưu đồ



d. Viết chương trình

```

File Edit Search Run Compile Debug Project Option Window Help
/* Chuong trinh hoan vi 2 so a, b neu a > b */

#include <stdio.h>
#include <conio.h>

void main(void)
{
    int ia, ib, itam;
    printf("Nhap vao so a: ");
  
```

```
scanf("%d", &ia);
printf("Nhập vào số b: ");
scanf("%d", &ib);
if (ia>ib)
{
    itam = ia;    //hoan vi a va b
    ia = ib;
    ib = itam;
}
printf("%d, %d.\n", ia, ib);
getch();
}
```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

☞ **Kết quả in ra màn hình**

| | |
|--|---|
| Nhập vào số a : 10 Nhập vào số b : 8 8, 10 | Cho chạy lại chương trình và thử lại với: a = 1, b = 8 a = 2, b = 2 Quan sát và nhận kết quả |
|--|---|

5.2.2.2 Dạng 2 (if đ ử)

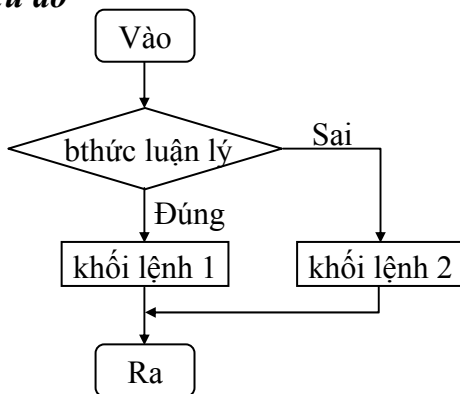
Quyết định sẽ thực hiện 1 trong 2 khối lệnh cho trước.

• **Cú pháp lệnh**

```
if (biểu thức luận lý)
    khối lệnh 1;
else
    khối lệnh 2;
```

- ☞ từ khóa **if, else** phải viết bằng chữ thường
- ☞ kết quả của **biểu thức luận lý** phải là **đúng (≠ 0)** hoặc **sai (= 0)**

• **Lưu đồ**



- ☞ nếu **biểu thức luận lý** đúng thì thực hiện khối lệnh 1 và thoát khỏi if ngược lại thực hiện khối lệnh 2 và thoát khỏi if.
- ☞ Nếu **khối lệnh 1, khối lệnh 2** bao gồm từ 2 lệnh trở lên thì phải đặt trong dấu { }

Ví dụ 5: Viết chương trình nhập vào 2 số nguyên a, b. In ra thông báo "a bằng b" nếu a = b, ngược lại in ra thông báo "a khác b".

a. Phác họa lời giải

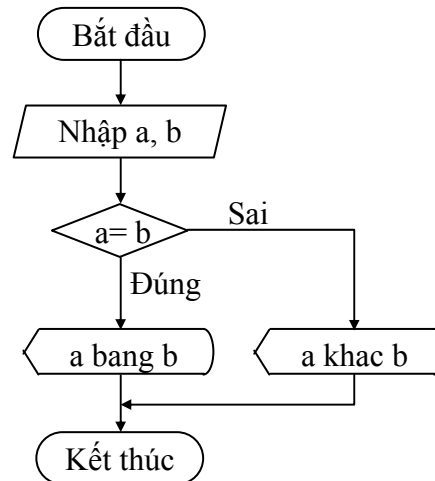
Số sánh a với b, nếu a bằng b thì in ra câu thông báo "a bằng b", ngược lại in ra thông báo "a khác b".

b. Mô tả quy trình xử lý (giải thuật)

| | |
|-------------------|------------|
| Ngôn ngữ tự nhiên | Ngôn ngữ C |
|-------------------|------------|

| | |
|---|---|
| <ul style="list-style-type: none"> - Khai báo 2 biến a, b kiểu số nguyên - Nhập vào giá trị a - Nhập vào giá trị b - Nếu a = b thì in ra thông báo "a bằng b" Ngược lại (còn không thì) in ra thông báo "a khác b" | <pre>- int ia, ib; - printf("Nhap vao so a: "); scanf("%d", &ia); - printf("Nhap vao so b: "); scanf("%d", &ib); - if (ia == ib) printf("a bang b\n"); else printf("a khac b\n");</pre> |
|---|---|

c. Mô tả bằng lưu đồ



d. Viết chương trình

| |
|--|
| File Edit Search Run Compile Debug Project Option Window Help |
| <pre>/* Chương trình in ra thông báo "a bằng b" nếu a = b, ngược lại in ra "a khác b" */ #include <stdio.h> #include <conio.h> void main(void) { int ia, ib; printf("Nhap vao so a: "); scanf("%d", &ia); printf("Nhap vao so b: "); scanf("%d", &ib); if (ia == ib) printf("a bang b\n"); else printf("a khac b\n"); getch(); }</pre> |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

👉 Kết quả in ra màn hình

| | |
|--|---|
| Nhập vào số a : 10 Nhập vào số b : 8 a khác b. | Cho chạy lại chương trình và thử lại với: a = 6, b = 6 a = 1, b = 5 |
|--|---|

Quan sát và nhận xét kết quả

☞ Sau else không có dấu chấm phẩy.
 Ví dụ: `else; printf('a khac b\n');`
 → trình biên dịch không báo lỗi, lệnh `printf("a khac b\n");` không thuộc else

Ví dụ 6: Viết chương trình nhập vào kí tự c. Kiểm tra xem nếu kí tự nhập vào là kí tự thường trong khoảng từ 'a' đến 'z' thì đổi sang chữ in hoa và in ra, ngược lại in ra thông báo "Kí tự bạn vừa nhập là: c".

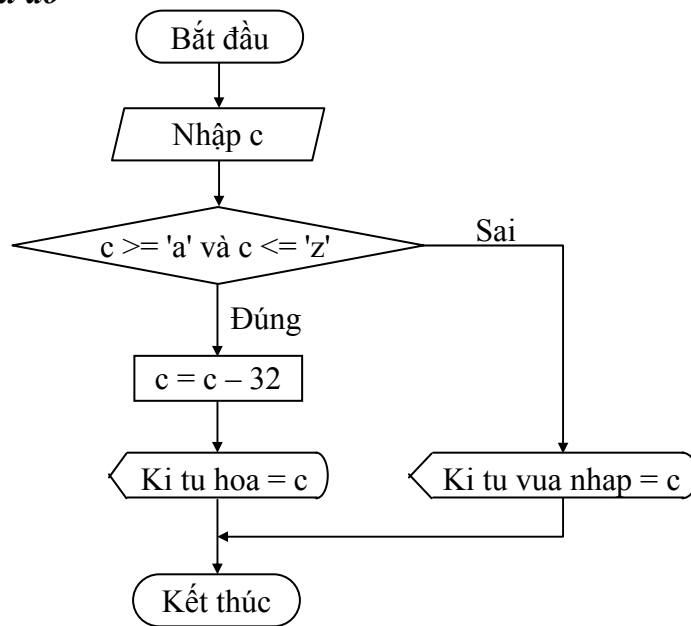
a. Phác họa lời giải

Trước tiên bạn phải kiểm tra xem nếu kí tự c thuộc khoảng 'a' và 'z' thì đổi kí tự c thành chữ in hoa bằng cách lấy kí tự c - 32 rồi gán lại cho chính nó (`c = c - 32`) (vì giữa kí tự thường và in hoa trong bảng mã ASCII cách nhau 32, ví dụ: A trong bảng mã ASCII là 65, B là 66..., còn a là 97, b là 98...), sau khi đổi xong bạn in kí tự c ra. Ngược lại, in câu thông báo "Kí tự bạn vừa nhập là: c".

b. Mô tả quy trình xử lý (giải thuật)

| Ngôn ngữ tự nhiên | Ngôn ngữ C |
|--|--|
| - Khai báo biến c kiểu kí tự - Nhập vào kí tự c - Nếu <code>c >= a</code> và <code>c <= z</code> thì <code>c = c - 32</code> in c ra màn hình Ngược lại in ra thông báo " Kí tự bạn vừa nhập là: c " | <pre> - char c; - printf("Nhap vao 1 ki tu: "); scanf("%c", &c); - if (c >= 'a' && c <= 'z') { c = c - 32; printf("Ki tu hoa la: %c.\n", c); }; else printf("Ki tu ban vua nhap la: %c.\n", c); </pre> |

c. Mô tả bằng lưu đồ



d. Viết chương trình

File Edit Search Run Compile Debug Project Option Window Help


```

/* Chương trình nhập vào ký tự c, nếu c là chữ thường in ra chữ IN HOA */
#include <stdio.h>
#include <conio.h>

void main(void)
{
    char c;
    printf("Nhập vào 1 ký tự: ");
    scanf("%c", &c);
    if (c >= 'a' && c <= 'z') // hoặc if(c >= 97 && c <= 122)
    {
        c = c - 32; // đổi thành chữ in hoa
        printf("Ký tự hoa là: %c.\n", c);
    };
    else
        printf("Ký tự bạn vừa nhập là: %c.\n", c);
    getch();
}
    
```

F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

☞ **Kết quả in ra màn hình**

| | |
|---|---|
| Nhập vào một ký tự: g Ký tự hoa là: G. | Cho chạy lại chương trình và thử lại với: c = '!', c = '2', c = 'A', c = 'u' Quan sát và nhận xét kết quả |
|---|---|

5.2.2.3 Cấu trúc else if

Quyết định sẽ thực hiện 1 trong n khối lệnh cho trước.

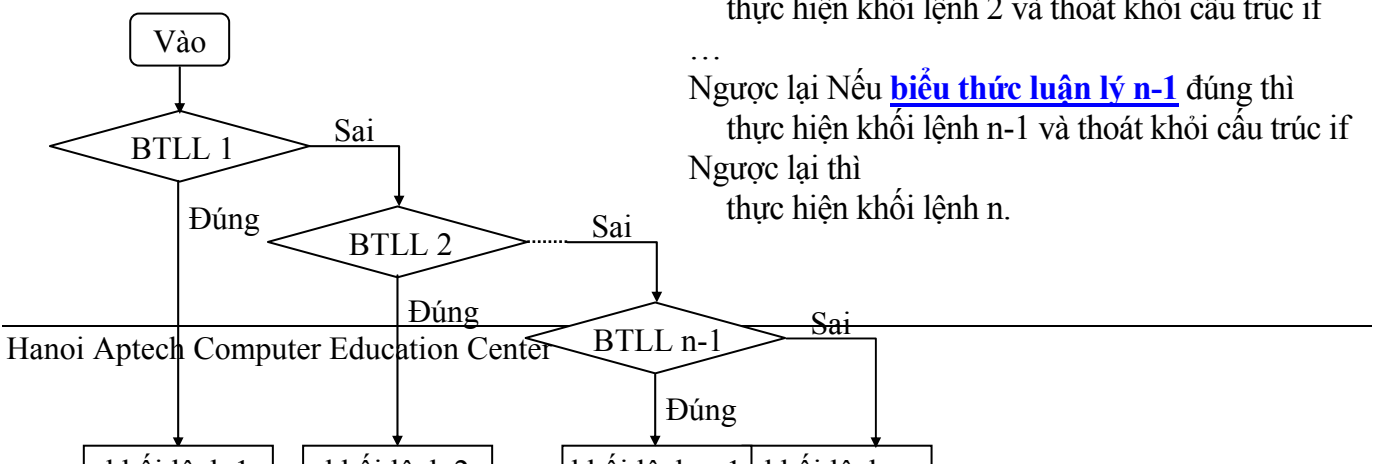
• **Cú pháp lệnh**

```

if (biểu thức luận lý 1)
    khối lệnh 1;
else if (biểu thức luận lý 2)
    khối lệnh 2;
...
else if (biểu thức luận lý n-1)
    khối lệnh n-1;
else
    khối lệnh n;
    
```

- ☞ từ khóa **if, else if, else** phải viết bằng chữ thường
- ☞ kết quả của **biểu thức luận lý 1, 2..n** phải là **đúng (≠ 0)** hoặc **sai (= 0)**
- ☞ Nếu **khối lệnh 1, 2...n** bao gồm từ 2 lệnh trở lên thì phải đặt trong dấu { }

• **Lưu đồ**



Nếu **biểu thức luận lý 1** đúng thì thực hiện khối lệnh 1 và thoát khỏi cấu trúc if
 Ngược lại Nếu **biểu thức luận lý 2** đúng thì thực hiện khối lệnh 2 và thoát khỏi cấu trúc if
 ...
 Ngược lại Nếu **biểu thức luận lý n-1** đúng thì thực hiện khối lệnh n-1 và thoát khỏi cấu trúc if
 Ngược lại thì thực hiện khối lệnh n.

Ví dụ 7: Viết chương trình nhập vào 2 số nguyên a, b. In ra thông báo "a lớn hơn b" nếu $a > b$, in ra thông báo "a nhỏ hơn b" nếu $a < b$, in ra thông báo "a bằng b" nếu $a = b$.

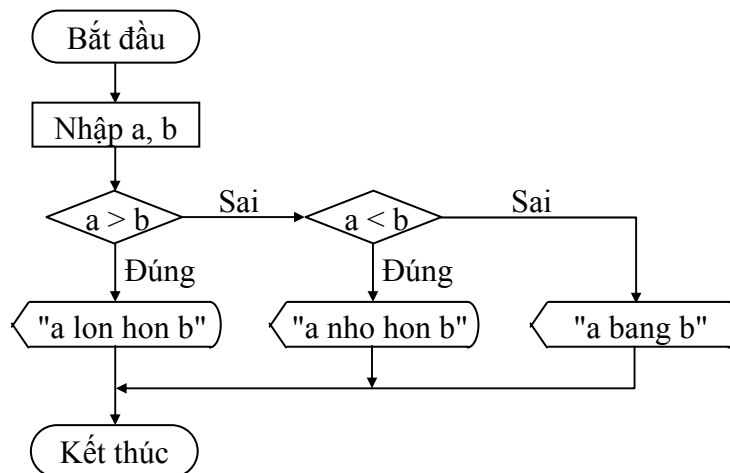
a. Phác họa lời giải

Trước tiên so sánh a với b. Nếu $a > b$ thì in ra thông báo "a lớn hơn b", ngược lại nếu $a < b$ thì in ra thông báo "a nhỏ hơn b", ngược với 2 trường hợp trên thì in ra thông báo "a bằng b".

b. Mô tả quy trình thực hiện (giải thuật)

| Ngôn ngữ tự nhiên | Ngôn ngữ C |
|--|---|
| - Khai báo 2 biến a, b kiểu số nguyên - Nhập vào giá trị a - Nhập vào giá trị b - Nếu $a > b$ thì in ra thông báo "a lớn hơn b" Ngược lại Nếu $a < b$ thì in ra thông báo "a nhỏ hơn b" Ngược lại thì in ra thông báo "a bằng b" | <pre> - int ia, ib; - printf("Nhap vao so a: "); scanf("%d", &ia); - printf("Nhap vao so b: "); scanf("%d", &ib); - if (ia > ib) printf("a lon hon b.\n"); else if (ia < ib) printf("a nho hon b.\n"); else printf("a bang b.\n"); </pre> |

c. Mô tả bằng lưu đồ



d. Viết chương trình

```

File Edit Search Run Compile Debug Project Option Window Help
/* Chuong trinh nhap vao 2 so nguyen a, b. In ra thong bao a > b, a < b, a = b */
#include <stdio.h>
#include <conio.h>

void main(void)
    
```

```

{
    int ia, ib;
    printf("Nhap vao so a: ");
    scanf("%d", &ia);
    printf("Nhap vao so b: ");
    scanf("%d", &ib);
    if (ia>ib)
        printf("a lon hon b.\n");
    else if (ia<ib)
        printf("a nho hon b.\n");
    else
        printf("a bang b.\n");
    getch();
}
    
```

F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

 **Kết quả in ra màn hình**

| | |
|---|---|
| Nhap vao so a : 5 Nhap vao so b : 7 a nho hon b | Cho chạy lại chương trình và thử lại với: a = 8, b = 4 a = 2, b = 2 Quan sát và nhận xét kết quả |
|---|---|

Ví dụ 8: Viết chương trình nhập vào kí tự c. Kiểm tra xem nếu kí tự nhập vào là kí tự thường trong khoảng từ 'a' đến 'z' thì đổi sang chữ in hoa và in ra, nếu kí tự in hoa trong khoảng A đến Z thì đổi sang chữ thường và in ra, nếu kí tự là số từ 0 đến 9 thì in ra câu "Kí tự bạn vừa nhập là số ...(in ra kí tự c)", còn lại không phải 3 trường hợp trên in ra thông báo "Bạn đã nhập kí tự ...(in ra kí tự c)".

a. Phác họa lời giải

Nhập kí tự c vào, kiểm tra xem nếu kí tự c thuộc khoảng 'a' và 'z' đổi kí tự c thành chữ in hoa bằng cách lấy kí tự c – 32 rồi gán lại cho chính nó (c = c – 32) (vì giữa kí tự thường và in hoa trong bảng mã ASCII cách nhau 32, ví dụ: A trong bảng mã ASCII là 65, B là 66..., còn a là 97, b là 98...), sau khi đổi xong bạn in kí tự c ra. Ngược lại Nếu kí tự c thuộc khoảng 'A' và 'Z', đổi kí tự c thành chữ thường (theo cách ngược lại) và in ra. Ngược lại Nếu kí tự c thuộc khoảng '0' và '9' thì in ra thông báo "Kí tự bạn vừa nhập là số...". Ngược lại, in câu thông báo "Bạn đã nhập kí tự...".

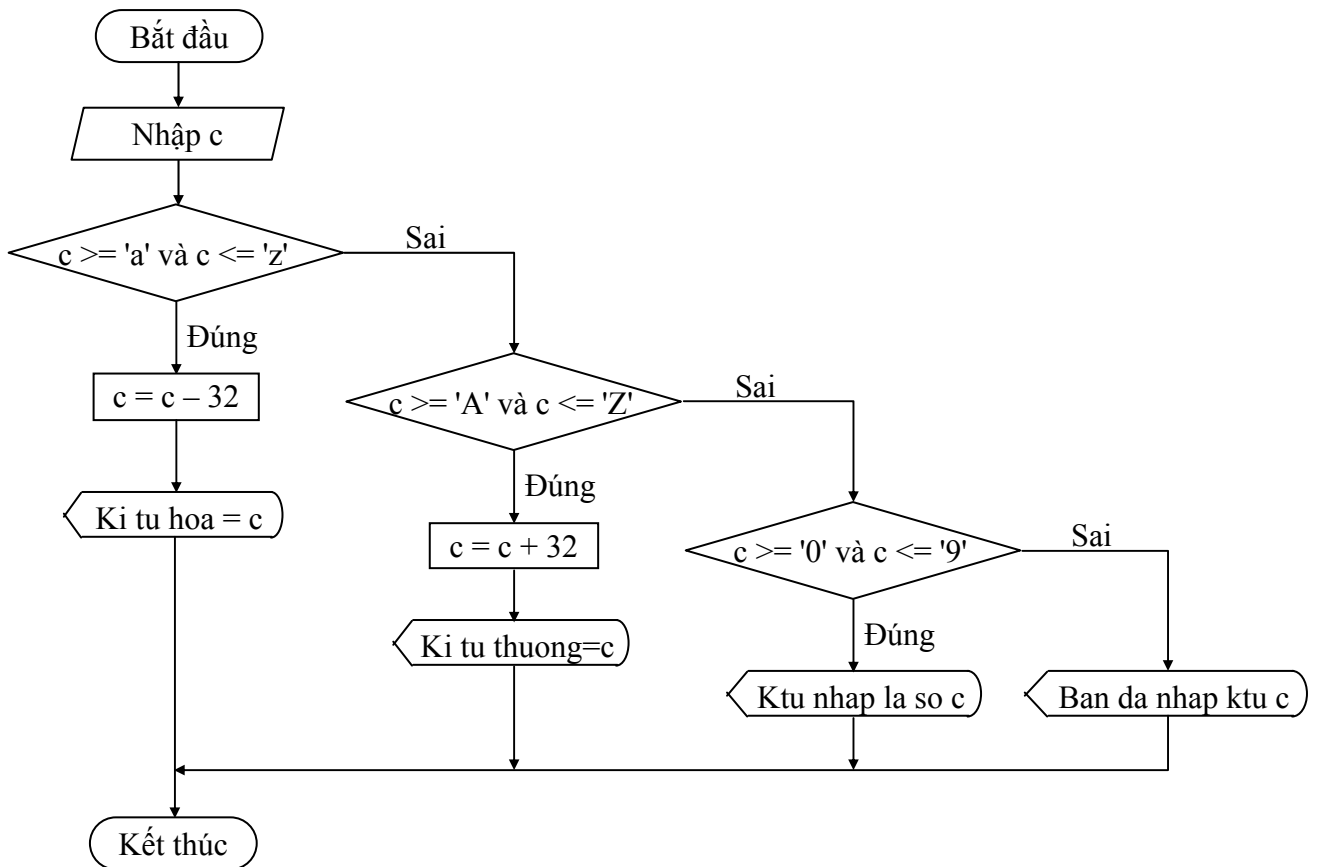
b. Mô tả quy trình xử lý (giải thuật)

| Ngôn ngữ tự nhiên | Ngôn ngữ C |
|--|---|
| - Khai báo biến c kiểu kí tự - Nhập vào kí tự c - Nếu c >= a và c <= z thì c = c – 32 in c ra màn hình Ngược lại Nếu c >= A và c <= Z thì c = c + 32 in c ra màn hình | - char c; - printf("Nhap vao 1 ki tu: "); scanf("%c", &c); - if (c >= 'a' && c <= 'z') { c = c – 32; printf("Ki tu hoa la: %c.\n", c); }; else if(c >= 'A' && c <= 'Z') { c = c + 32; printf("Ki tu thuong la: %c.\n", c); |

| | |
|--|---|
| Ngược lại Nếu $c \geq 0$ và $c \leq 9$ thì in thông báo "Kí tự bạn vừa nhập là số c" Ngược lại thì in thông báo "Bạn đã nhập kí tự c" | <pre>}; else if(c >= '0' && c <= '9') printf("Kí tự Ban vua nhap la so %c.\n", c); else printf("Ban da nhap ki tu %c.\n", c);</pre> |
|--|---|

☞ Cũng như if, không đặt dấu chấm phẩy sau câu lệnh else if.
 Ví dụ: `else if(c >= 'A' && c <= 'Z');`
 → trình biên dịch không báo lỗi nhưng khối lệnh sau else if không được thực hiện.

c. Mô tả bằng lưu đồ



e. Viết chương trình

```

File Edit Search Run Compile Debug Project Option Window Help
/* Chuong trinh nhap vao ki tu c. Doi ra hoa, thuong */

#include <stdio.h>
#include <conio.h>

void main(void)
{
    char c;
    printf("Nhap vao 1 ki tu: ");
    scanf("%c", &c);
    if (c >= 'a' && c <= 'z')           //hoac if(c >= 97 && c <= 122)
    {
        c = c - 32;                    //doi thanh chu in hoa
    }
}
    
```

```

printf("Ki tu hoa la: %c.\n", c);
};
else if(c >= 'A' && c <= 'Z') //hoac if(c >= 65 && c <= 90)
{
    c = c + 32; //doi thanh chu thuong
    printf("Ki tu thuong la: %c.\n", c);
};
else if(c >= '0' && c <= '9') //hoac if(c >= 48 && c <= 57)
    printf("Ki tu Ban vua nhap la so %c.\n", c);
else
    printf("Ban da nhap ki tu %c.\n", c);
getch();
}

```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

☞ Kết quả in ra màn hình

Nhap vao mot ki tu: g
Ki tu hoa la: G.

Cho chạy lại chương trình và thử lại với:
c = '!', c = '2', c = 'a', c = 'Z'
Quan sát và nhận xét kết quả

5.2.2.4 Cấu trúc if lồng

Quyết định sẽ thực hiện 1 trong n khối lệnh cho trước.

- **Cú pháp lệnh**

Cú pháp là một trong 3 dạng trên, nhưng trong 1 hoặc nhiều khối lệnh bên trong phải chứa ít nhất một trong 3 dạng trên gọi là cấu trúc if lồng nhau. Thường cấu trúc if lồng nhau càng nhiều cấp độ phức tạp càng cao, chương trình chạy càng chậm và trong lúc lập trình dễ bị nhầm lẫn.

Lưu ý: Các lệnh **if...else** lồng nhau thì **else** sẽ luôn luôn kết hợp với **if** nào chưa có **else** gần nhất. Vì vậy khi gặp những lệnh if không có else, Bạn phải đặt chúng trong những **khối lệnh rõ ràng** để tránh bị hiểu sai câu lệnh.

Ví dụ 9: Bạn viết các dòng lệnh sau:

```

...
if (n > 0)
    if (a > b)
        x = a;
else
    x = b;

```

Mặc dù Bạn viết lệnh else thẳng hàng với if (n > 0), nhưng lệnh else ở đây được hiểu đi kèm với if (a > b), vì nó nằm gần với if (a > b) nhất và if (a > b) chưa có else. Để dễ nhìn và dễ hiểu hơn Bạn viết lại như sau:

```

...
if (n > 0)
    if (a > b)
        x = a;
    else
        x = b;

```

Còn nếu Bạn muốn lệnh else là của if (n > 0) thì Bạn phải đặt if (a > b) x = a trong một khối lệnh. Bạn viết lại như sau:

...

```

if (n > 0)
{
    if (a > b)
        x = a;
}
else
    x = b;
...
    
```

• **Lưu đồ**

Tương tự 3 dạng trên. Nhưng trong mỗi khối lệnh có thể có một (nhiều) cấu trúc if ở 3 dạng trên.

Ví dụ 10: Viết chương trình nhập vào điểm của một học sinh. In ra xếp loại học tập của học sinh đó. (Cách xếp loại. Nếu điểm ≥ 9 , Xuất sắc. Nếu điểm từ 8 đến cận 9, Giỏi. Nếu điểm từ 7 đến cận 8, Khá. Nếu điểm từ 6 đến cận 7, TBKhá. Nếu điểm từ 5 đến cận 6, TBình. Còn lại là Yếu).

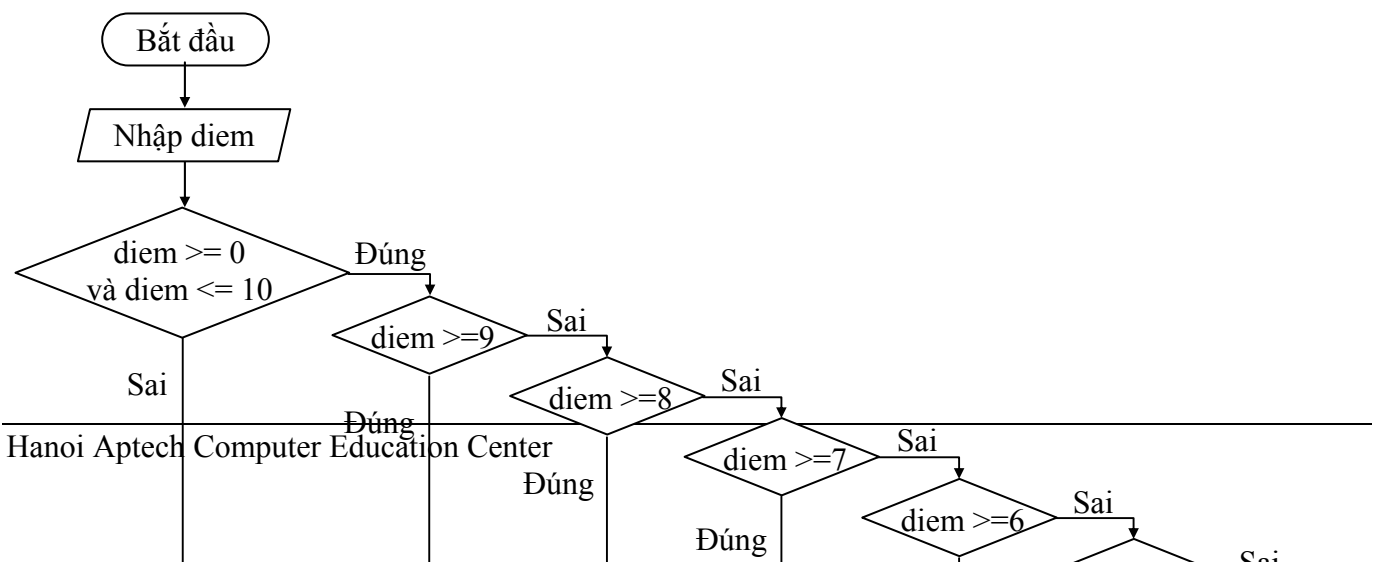
a. Phác họa lời giải

Điểm số nhập vào nếu hợp lệ ($0 \leq \text{điểm} \leq 10$), bạn tiếp tục công việc xếp loại, ngược lại thông báo "Nhập điểm không hợp lệ". Việc xếp loại bạn sử dụng cấu trúc else if.

b. Mô tả quy trình xử lý (giải thuật)

| Ngôn ngữ tự nhiên | Ngôn ngữ C |
|---|---|
| - Khai báo biến diem kiểu số thực - Nhập vào điểm số - Nếu diem ≥ 0 và diem ≤ 10 thì - Nếu diem ≥ 9 thì in ra xếp loại = Xuất sắc Ngược lại Nếu diem ≥ 8 thì in ra xếp loại = Giỏi Ngược lại Nếu diem ≥ 7 thì in ra xếp loại = Khá Ngược lại Nếu diem ≥ 6 thì in ra xếp loại = TBKhá Ngược lại Nếu diem ≥ 5 thì in ra xếp loại = TBình Ngược lại thì in ra xếp loại = Yếu Ngược lại thì in ra "Bạn nhập điểm không hợp lệ" | <pre> - float fdiem; - printf("Nhap vao diem so: "); scanf("%f", &fdiem); - if (fdiem >= 0 && fdiem <= 10) - if (fdiem >= 9) printf("Xep loai = Xuat sac.\n"); else if (fdiem >= 8) printf("Xep loai = Gioi.\n"); else if (fdiem >= 7) printf("Xep loai = Kha.\n"); else if (fdiem >= 6) printf("Xep loai = TBKha.\n"); else if (fdiem >= 5) printf("Xep loai = TBinh.\n"); else printf("Xep loai = Yeu.\n"); else printf("Ban nhap diem khong hop le.\n"); </pre> |

c. Mô tả bằng lưu đồ



d. Viết chương trình

| File Edit Search Run Compile Debug Project Option Window Help |
|--|
| <pre> /* Chuong trinh nhap vao 2 so nguyen a, b. In ra thong bao a > b, a < b, a = b */ #include <stdio.h> #include <conio.h> void main(void) { float fdiem; printf("Nhap vao diem so: "); scanf("%f", &fdiem); if (fdiem >=0 && fdiem <=10) if (fdiem >=9) printf("Xep loai = Xuat sac.\n"); else if (fdiem >=8) printf("Xep loai = Gioi.\n"); else if (fdiem >=7) printf("Xep loai = Kha.\n"); else if (fdiem >=6) printf("Xep loai = TBKha.\n"); else if (fdiem >=5) printf("Xep loai = TBinh.\n"); else printf("Xep loai = Yeu.\n"); else //if (fdiem>=0 && fdiem<=10) printf("Nhap diem khong hop le.\n"); getch(); } </pre> |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

☞ Kết quả in ra màn hình

| | |
|--|--|
| Nhap vao diem so: 6.5 Xep loai = TBKha. | Cho chạy lại chương trình và thử lại với: diem = 4, diem = 9, diem = 7, diem = 12 Quan sát và nhận xét kết quả |
|--|--|

e. Bàn thêm về chương trình

Trong chương trình trên cấu trúc **else if được lồng vào trong cấu trúc dạng 2**, trong cấu trúc else if ta không cần đặt trong khối vì tất cả các if trong cấu trúc này đều có else, nên `else printf("Nhap diem khong hop le.\n")` đương nhiên là thuộc về if (`fdiem >= 0 && fdiem <= 10`). Giả sử trong cấu trúc else if không có dòng `else printf("Xep loai = Yeu.\n")` thì khi đó dòng `else printf("Nhap diem khong hop le.\n")` sẽ thuộc về cấu trúc else if chứ không thuộc về if (`fdiem >=0 && fdiem <= 10`). Đối với trường hợp đó bạn cần phải đặt cấu trúc else if vào trong {}, thì khi đó dòng `else printf("Nhap diem khong hop le.\n")` sẽ thuộc về if (`fdiem >= 0 && fdiem <= 10`).

Ví dụ 11: Viết chương trình nhập vào 3 số nguyên a, b, c. Tìm và in ra số lớn nhất.

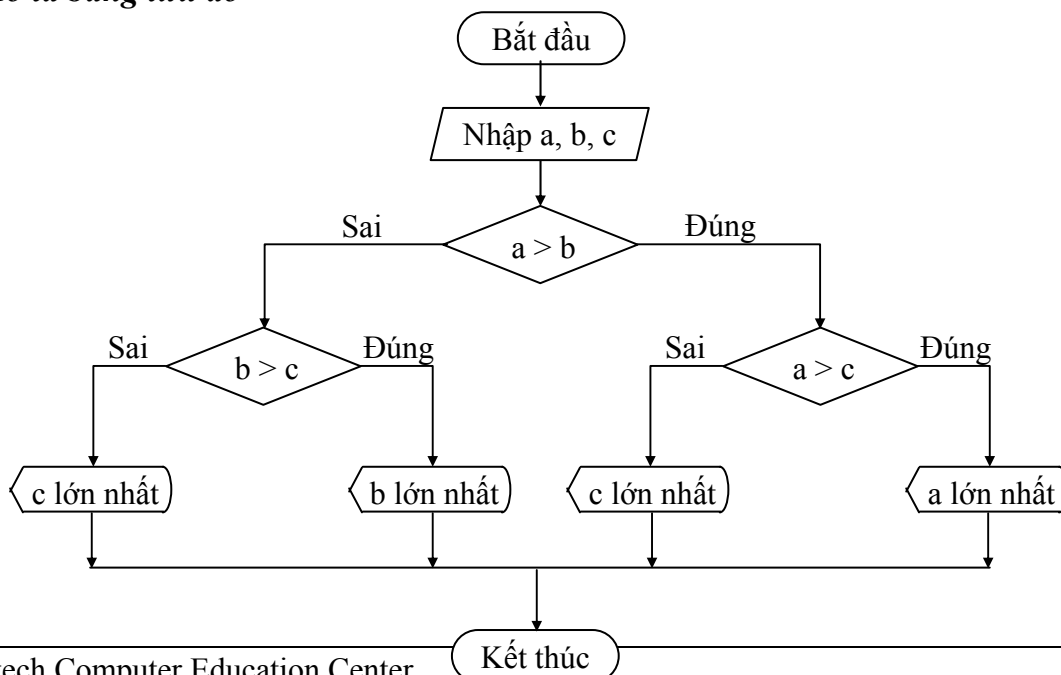
a. Phác họa lời giải

Trước tiên bạn so nếu $a > b$, mà $a > c$ thì a lớn nhất, ngược lại c lớn nhất, còn nếu $a \leq b$, mà $c > b$ thì b lớn nhất, ngược lại c lớn nhất.

b. Mô tả quy trình xử lý (giải thuật)

| Ngôn ngữ tự nhiên | Ngôn ngữ C |
|---|--|
| - Khai báo 3 biến a, b, c kiểu số nguyên - Nhập vào số a - Nhập vào số b - Nhập vào số c - Nếu $a > b$ thì - Nếu $a > c$ thì a lớn nhất Ngược lại thì c lớn nhất Ngược lại - Nếu $b > c$ thì b lớn nhất Ngược lại thì c lớn nhất | - int ia, ib, ic; - printf("Nhap vao so a: "); scanf("%d", &ia); - printf("Nhap vao so b: "); scanf("%d", &ib); - printf("Nhap vao so c: "); scanf("%d", &ic); - if (ia > ib) - if (ia > ic) printf("%d lon nhat.\n", ia); else printf("%d lon nhat.\n", ic); else - if (ib > ic) printf("%d lon nhat.\n", ib); else printf("%d lon nhat.\n", ic); |

c. Mô tả bằng lưu đồ



d. Viết chương trình

| |
|---|
| File Edit Search Run Compile Debug Project Option Window Help |
| <pre> /* Chương trình nhập vào 2 số nguyên a, b, c. Tìm, in ra số lớn nhất */ #include <stdio.h> #include <conio.h> void main(void) { int ia, ib, ic; printf("Nhập vào số a: "); scanf("%d", &ia); printf("Nhập vào số b: "); scanf("%d", &ib); printf("Nhập vào số c: "); scanf("%d", &ic); if (ia > ib) if (ia > ic) printf("%d lớn nhất.\n", ia); else printf("%d lớn nhất.\n", ic); else if (ib > ic) printf("%d lớn nhất.\n", ib); else printf("%d lớn nhất.\n", ic); getch(); } </pre> |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

👉 Kết quả in ra màn hình

| | |
|---|---|
| Nhập vào số a: 4 Nhập vào số b: 5 Nhập vào số c: 3 5 lớn nhất. | Cho chạy lại chương trình và thử lại với: a = 5, b = 4, c = 2 a = 2, b = 1, c = 10 a = 5, b = 5, c = 5 Quan sát và nhận xét kết quả |
|---|---|

e. Bàn thêm về chương trình

Trong chương trình trên cấu trúc **dạng 2** được lồng vào trong cấu trúc **dạng 2**.

5.2.3 Lệnh switch

Lệnh switch cũng giống cấu trúc else if, nhưng nó mềm dẻo hơn và linh động hơn nhiều so với sử dụng if. Tuy nhiên, nó cũng có mặt hạn chế là kết quả của biểu thức phải là giá trị hằng nguyên (có giá trị cụ thể). Một bài toán sử dụng lệnh switch thì cũng có thể sử dụng if, nhưng ngược lại còn tùy thuộc vào giải thuật của bài toán.

5.2.3.1 Cấu trúc switch...case (switch thiếu)

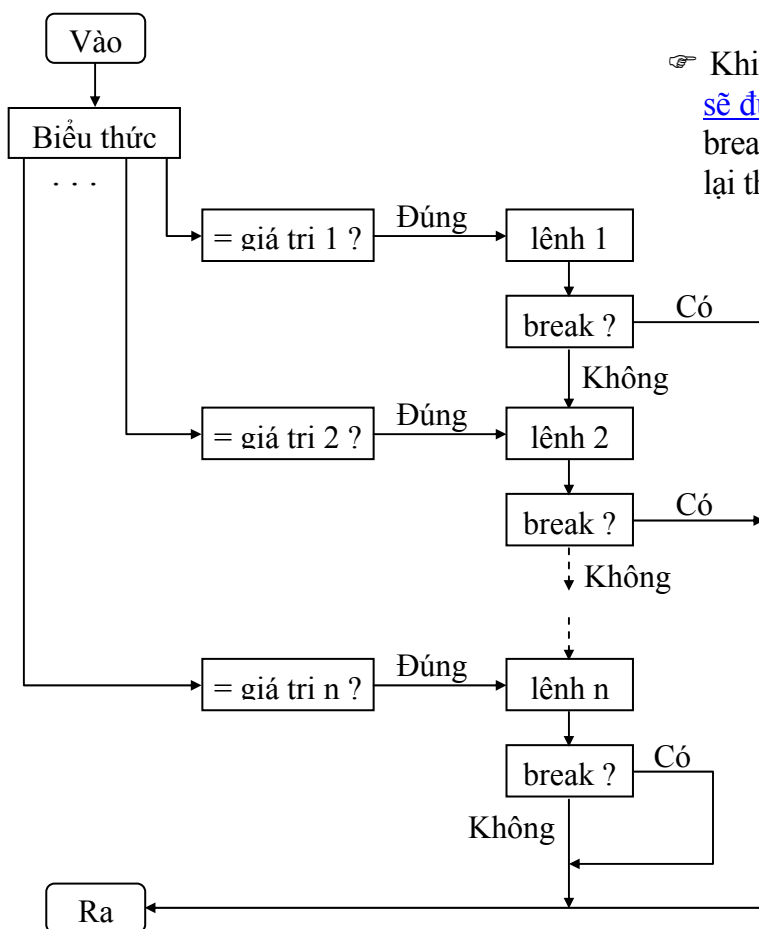
Chọn thực hiện 1 trong n lệnh cho trước.

- **Cú pháp lệnh**

```
switch (biểu thức)
{
    case giá trị 1 : lệnh 1;
                  break;
    case giá trị 2 : lệnh 2;
                  break;
    ...
    case giá trị n : lệnh n;
                  [break;]
}
```

- ☞ từ khóa **switch, case, break** phải viết bằng chữ thường
- ☞ **biểu thức** phải là có kết quả là **giá trị hằng nguyên (char, int, long,...)**
- ☞ **Lệnh 1, 2...n** có thể gồm nhiều lệnh, nhưng không cần đặt trong cặp dấu { }

- **Lưu đồ**



- ☞ Khi **giá trị của biểu thức bằng giá trị i** thì **lệnh i sẽ được thực hiện**. Nếu sau lệnh i không có lệnh break thì sẽ tiếp tục thực hiện lệnh i + 1... Ngược lại thoát khỏi cấu trúc switch.

Ví dụ 12: Viết chương trình nhập vào số 1, 2, 3. In ra tương ứng 1, 2, 3 sao.

a. Viết chương trình

```
File Edit Search Run Compile Debug Project Option Window Help
```

```
/* Chương trình nhập vào số 1, 2, 3. In ra số sao tương ứng */
#include <stdio.h>
#include <conio.h>
void main(void)
{
    int i;
    printf("Nhập vào số 1, 2 hoặc 3: ");
```

```
scanf("%d", &i);
switch(i)
{
    case 3: printf("*");
    case 2: printf("*");
    case 1: printf("*");
};
printf("An phim bat ky de ket thuc!\n");
getch();
}
```

F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

☞ Kết quả in ra màn hình

Nhap vao so 1, 2 hoặc 3: 2
**

Cho chạy lại chương trình và thử lại với:
i = 1, i = 3, i = 0, i = 4
Quan sát và nhận xét kết quả

b. Bàn thêm về chương trình

Trong chương trình trên khi nhập vào i = 2 lệnh printf("*") ở dòng case 2 được thi hành, nhưng do không có lệnh break sau đó nên lệnh printf("*") ở dòng case 1 tiếp tục được thi hành. Kết quả in ra **.

☞ Không đặt dấu chấm phẩy sau câu lệnh switch.

Ví dụ: switch(i);

→ trình biên dịch không báo lỗi nhưng các lệnh trong switch không được thực hiện.

Ví dụ 13: Viết chương trình nhập vào tháng và in ra quý. (tháng 1 -> quý 1, tháng 10 -> quý 4)

a. Phác họa lời giải

Nhập vào giá trị tháng, kiểm tra xem tháng có hợp lệ (trong khoảng 1 đến 12). Nếu hợp lệ in ra quý tương ứng (1->3: quý 1, 4->6: quý 2, 7->9: quý 3, 10->12: quý 4).

b. Viết chương trình

File Edit Search Run Compile Debug Project Option Window Help

```
/* Chuong trinh nhap vao thang. In ra quy tuong ung */
```

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
void main(void)
```

```
{
    int ithang;
    printf("Nhap vao thang: ");
    scanf("%d", &ithang);
    if (ithang > 0 && ithang <= 12)
        switch(ithang)
        {
            case 1:
            case 2:
            case 3: printf("Quy 1.\n");
                    break;

            case 4:
            case 5:
            case 6: printf("Quy 2.\n");
                    break;

            case 7:
            case 8:
```

```

    case 9: printf("Quy 3.\n");
            break;
    case 10:
    case 11:
    case 12: printf("Quy 4.\n");
            break;
};
else
    printf("Thang khong hop le.\n");
getch();
}

```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

☞ **Kết quả in ra màn hình**

| | |
|-----------------------------|---|
| Nhap vao thang: 4 Quy 2. | Cho chạy lại chương trình và thử lại với: thang = 7, thang = 1, thang = 13, thang = -4 Quan sát và nhận xét kết quả |
|-----------------------------|---|

c. Bàn thêm về chương trình

Trong chương trình trên cấu trúc **switch...case** được lồng vào trong cấu trúc **if** dạng 2.

5.2.3.2 Cấu trúc switch...case...default (switch đủ)

Chọn thực hiện 1 trong n + 1 lệnh cho trước.

• **Cú pháp lệnh**

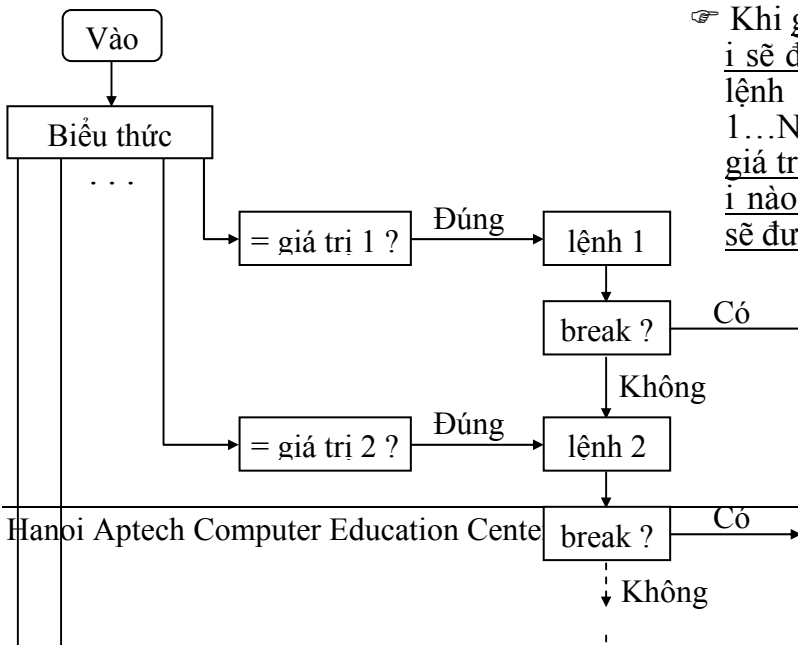
```

switch (biểu thức)
{
    case giá trị 1 : lệnh 1;
                  break;
    case giá trị 2 : lệnh 2;
                  break;
    ...
    case giá trị n : lệnh n;
                  break;
    default      : lệnh;
                  [break;]
}

```

- ☞ từ khóa **switch, case, break, default** phải viết bằng chữ thường
- ☞ **biểu thức** phải là có kết quả là giá trị nguyên (char, int, long,...)
- ☞ **Lệnh 1, 2...n** có thể gồm nhiều lệnh, nhưng không cần đặt trong cặp dấu { }

• **Lưu đồ**



☞ Khi giá trị của biểu thức bằng giá trị i thì lệnh i sẽ được thực hiện. Nếu sau lệnh i không có lệnh break thì sẽ tiếp tục thực hiện lệnh i + 1... Ngược lại thoát khỏi cấu trúc switch. Nếu giá trị biểu thức không trùng với bất kỳ giá trị i nào thì lệnh tương ứng với từ khóa default sẽ được thực hiện.

Ví dụ 14: Viết lại chương trình ở **Ví dụ 12**

a. Viết chương trình

| File Edit Search Run Compile Debug Project Option Window Help |
|---|
| <pre> /* Chuong trinh nhap vao so 1, 2, 3. In ra so sao tuong ung */ #include <stdio.h> #include <conio.h> void main(void) { int i; printf("Nhap vao so 1, 2 hoặc 3: "); scanf("%d", &i); switch(i) { case 3: printf("*"); case 2: printf("*"); case 1: printf("*"); break; default: printf("Ban nhap phai nhap vao so 1, 2 hoac 3.\n"); }; getch(); } </pre> |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

 **Kết quả in ra màn hình**

| | |
|--|---|
| Nhap vao so 1, 2 hoặc 3: 3 *** _ | Cho chạy lại chương trình và thử lại với: i = 1, i = 3, i = 0, i = 4 Quan sát kết quả |
|--|---|

b. Bàn thêm về chương trình

Trong chương trình trên. Nếu bạn nhập vào 1, 2, 3 sẽ in ra số sao tương ứng. Ngoài các số này chương trình sẽ in ra câu thông báo "Bạn phải nhập vào số 1, 2 hoặc 3".

Ví dụ 15: Viết lại chương trình ở **Ví dụ 13**

a. Viết chương trình

```

File Edit Search Run Compile Debug Project Option Window Help

/* Chuong trinh nhap vao thang. In ra quy tuong ung */

#include <stdio.h>
#include <conio.h>

void main(void)
{
    int ithang;
    printf("Nhap vao thang: ");
    scanf("%d", &ithang);
    switch(ithang)
    {
        case 1: case 2: case 3 :    printf("Quy 1.\n");
                                   break;
        case 4: case 5: case 6:    printf("Quy 2.\n");
                                   break;
        case 7: case 8: case 9:    printf("Quy 3.\n");
                                   break;
        case 10: case 11: case 12: printf("Quy 4.\n");
                                   break;
        default                    : printf("Ban phai nhap vao so trong khoang 1..12\n");
    };
    getch();
}
F1 Help  Alt-F8 Next Msg  Alt-F7 Prev Msg  Alt - F9 Compile  F9 Make  F10 Menu
    
```

👉 Kết quả in ra màn hình

| | |
|-----------------------------|---|
| Nhập vào thang: 4 Quy 2. | Cho chạy lại chương trình và thử lại với: thang = 7, thang = 1, thang = 13, thang = -4 Quan sát kết quả |
|-----------------------------|---|

c. Bàn thêm về chương trình

Trong chương trình trên. Nếu bạn nhập vào 1 đến 12 sẽ in quý tương ứng. Ngoài các số này chương trình sẽ in ra câu thông báo "Bạn phải nhập vào số trong khoảng 1..12".

5.2.3.3 Cấu trúc switch lồng

Quyết định sẽ thực hiện 1 trong n khối lệnh cho trước.

• **Cú pháp lệnh**

Cú pháp là một trong 2 dạng trên, nhưng trong 1 hoặc nhiều lệnh bên trong phải chứa ít nhất một trong 2 dạng trên gọi là cấu trúc switch lồng nhau. Thường cấu trúc switch lồng nhau càng nhiều cấp độ phức tạp càng cao, chương trình chạy càng chậm và trong lúc lập trình dễ bị nhầm lẫn.

• **Lưu đồ**

Tương tự 2 dạng trên. Nhưng trong mỗi lệnh có thể có một (nhiều) cấu trúc switch ở 2 dạng trên.

Ví dụ 16: Viết chương trình menu 2 cấp

a. Viết chương trình

| File Edit Search Run Compile Debug Project Option Window Help |
|---|
| <pre> /* Chuong trinh menu 2 cap */ #include <stdio.h> #include <conio.h> void main(void) { int imenu, isubmenu; printf("-----\n"); printf(" MAIN MENU \n"); printf("-----\n"); printf("1. File\n"); printf("2. Edit\n"); printf("3. Search\n"); printf("Chon muc tuong ung: "); scanf("%d", &imenu); switch(imenu) { case 1: printf("-----\n"); printf(" MENU FILE \n"); printf("-----\n"); printf("1. New\n"); printf("2. Open\n"); printf("Chon muc tuong ung: "); scanf("%d", &isubmenu); switch(isubmenu) { case 1: printf("Ban da chon chuc nang New File\n"); break; case 2: printf("Ban da chon chuc nang Open File\n"); break; } break; //break cua case 1 – switch(imenu) case 2: printf("Ban da chon chuc nang Edit\n"); break; case 3: printf("Ban da chon chuc nang Search\n"); }; getch(); } </pre> |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

| | |
|----------------------------|--|
| <pre>----- MAIN MENU</pre> | Cho chạy lại chương trình và thử lại với: mục chọn chức năng khác |
|----------------------------|--|

| | |
|--|--|
| <p>----- 1. File 2. Edit 3. Search Chọn mục tương ứng: 1 ----- MENU FILE ----- 1. New 2. Open Chọn mục tương ứng: 2 Bạn đã chọn chức năng Open File</p> | <p>Quan sát kết quả. * Thêm các thành phần sau vào chương trình: - Thêm mục Save vào menu File. - Tạo menu Edit gồm 4 chức năng: Copy, Cut, Paste, Clear. - Tạo menu Search gồm 2 chức năng: Find, Replace. Chạy lại chương trình và thử với nhiều mục chọn khác nhau. Quan sát kết quả.</p> |
|--|--|

5.3 Bài tập

5.3.1 Sử dụng lệnh if

1. *Viết lại chương trình ví dụ 3, sử dụng cấu trúc if dạng 2.*
2. *Viết lại chương trình ví dụ 11, sử dụng cấu trúc if dạng 1.*
3. *Viết lại chương trình ví dụ 11, sử dụng cấu trúc if dạng 2.*
4. *Viết chương trình nhập vào số nguyên dương, in ra thông báo số chẵn hay lẻ.*

Hướng dẫn: Nhập vào số nguyên dương x. Kiểm tra nếu x chia chẵn cho hai thì x là số chẵn (hoặc chia cho 2 dư 0) ngược lại là số lẻ.

5. *Viết chương trình nhập vào 4 số nguyên. Tìm và in ra số lớn nhất.*

Hướng dẫn: Ta có 4 số nguyên a, b, c, d. Tìm 2 số nguyên lớn nhất x, y của 2 cặp (a, b) và (c, d). Sau đó so sánh 2 số nguyên x, y để tìm ra số nguyên lớn nhất.

6. *Viết chương trình giải phương trình bậc 2: $ax^2 + bx + c = 0$, với a, b, c nhập vào từ bàn phím.*

Hướng dẫn: Nhập vào 3 biến a, b, c.

Tính Delta = $b^2 - 4*a*c$

Nếu Delta < 0 thì

Phương trình vô nghiệm

Ngược lại

Nếu Delta = 0 thì

$x_1 = x_2 = -b/(2*a)$

Ngược lại

$x_1 = (-b - \sqrt{\text{Delta}})/(2*a)$

$x_2 = (-b + \sqrt{\text{Delta}})/(2*a)$

Hết Nếu

Hết Nếu

7. *Viết chương trình nhập vào giờ phút giây (hh:mm:ss). Cộng thêm số giây nhập vào và in ra kết quả dưới dạng hh:mm:ss.*

Hướng dẫn: Nhập vào giờ phút giây vào 3 biến gio, phut, giay và nhập và giây công thêm vào biến them:

Nếu giay + them < 60 thì

giay = giay + them

Ngược lại

giay = (giay + them) - 60

phut = phut + 1

Nếu phut >= 60 thì

phut = phut - 60

gio = gio + 1

Hết nếu

Hết nếu

5.3.2 Sử dụng lệnh switch

8. *Viết chương trình nhập vào tháng, in ra tháng đó có bao nhiêu ngày.*

Hướng dẫn: Nhập vào tháng

Nếu là tháng 1, 3, 5, 7, 8, 10, 12 thì có 30 ngày

Nếu là tháng 4, 6, 9, 11 thì có 31 ngày

Nếu là tháng 2 và là năm nhuận thì có 29 ngày ngược lại 28 ngày

(Năm nhuận là năm chia chắn cho 4)

9. *Viết chương trình trò chơi One-Two-Three ra cái gì ra cái này theo điều kiện:*

- Búa (B) thắng Kéo, thua Giấy.

- Kéo (K) thắng Giấy, thua Búa.

- Giấy (G) thắng Búa, thua Kéo.

Hướng dẫn: Dùng lệnh switch lồng nhau

10. *Viết chương trình xác định biến ký tự color rồi in ra thông báo*

- RED, nếu color = 'R' hoặc color = 'r'

- GREEN, nếu color = 'G' hoặc color = 'g'

- BLUE, nếu color = 'B' hoặc color = 'b'

- BLACK, nếu color có giá trị khác.

11. *Viết chương trình nhập vào 2 số x, y và 1 trong 4 toán tử +, -, *, /. Nếu là + thì in ra kết quả x + y, nếu là - thì in ra x - y, nếu là * thì in ra x * y, nếu là / thì in ra x / y (nếu y = 0 thì thông báo không chia được)*

5.4 Bài tập làm thêm

12. *Viết lại bài tập 8, 9, 10, 11 sử dụng lệnh if.*

13. *Viết chương trình nhập vào điểm 3 môn thi: Toán, Lý, Hóa của học sinh. Nếu tổng điểm >= 15 và không có môn nào dưới 4 thì in kết quả đậu. Nếu đậu mà các môn đều lớn hơn 5 thì in ra lời phê "Học đều các môn", ngược lại in ra "Học chưa đều các môn", các trường hợp khác là "Thì hỏng".*

14. *Viết chương trình nhập vào ngày tháng năm (dd:mm:yy), cho biết đó là thứ mấy trong tuần.*

15. *Viết chương trình nhập số giờ làm và lương giờ rồi tính số tiền lương tổng cộng. Nếu số giờ làm lớn hơn 40 thì những giờ làm dôi ra được tính 1,5 lần.*

16. *Viết chương trình nhập vào 3 giá trị nguyên dương a, b, c. Kiểm tra xem a, b, c có phải là 3 cạnh của tam giác không? Nếu là 3 cạnh của tam giác thì tính diện tích của tam giác theo công thức sau:*

17. $S = \sqrt{p * (p - a) * (p - b) * p - c}$, với p là 1/2 chu vi của tam giác.

Hướng dẫn: a, b, c là 3 cạnh của tam giác phải thỏa điều kiện sau:

(a + b) > c và (a + c) > b và (b + c) > a

18. Viết chương trình nhập vào 3 số nguyên rồi in ra màn hình theo thứ tự tăng dần.

19. Viết chương trình tính tiền điện gồm các khoảng sau:

- Tiền thuê bao điện kê: 1000đ/tháng
- Định mức sử dụng điện cho mỗi hộ là: 50 KW với giá 230đ/KW
- Nếu phần vượt định mức $\leq 50KW$ thì tính giá 480đ/KW
- Nếu $50KW < \text{phần vượt định mức} < 100KW$ thì tính giá 700đ/KW
- Nếu phần vượt định mức $\leq 100KW$ thì tính giá 900đ/KW

Chỉ số mới và cũ được nhập vào từ bàn phím

- In ra màn hình chỉ số cũ, chỉ số mới, tiền trả định mức, tiền trả vượt định mức, tổng tiền phải trả.

Bài 6 :

CẤU TRÚC VÒNG LẶP

6.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Ý nghĩa, cách hoạt động của vòng lặp.
- Cú pháp, ý nghĩa, cách sử dụng lệnh for, while, do...while.
- Ý nghĩa và cách sử dụng lệnh break, continue.
- Một số bài toán sử dụng lệnh for, while, do...while thông qua các ví dụ.
- So sánh, đánh giá một số bài toán sử dụng lệnh for, while hoặc do...while.
- Cấu trúc vòng lặp lồng nhau.

6.2 Nội dung

6.2.1 Lệnh for

Vòng lặp xác định thực hiện lặp lại một số lần xác định của một (chuỗi hành động)

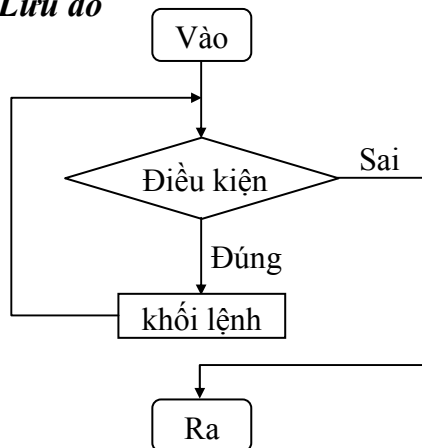
- **Cú pháp lệnh**

**for (biểu thức 1; biểu thức 2; biểu thức 3)
khối lệnh;**

☞ từ khóa **for** phải viết bằng chữ thường

☞ Nếu **khối lệnh** bao gồm từ 2 lệnh trở lên thì phải đặt trong dấu { }

- **Lưu đồ**



☞ kiểm tra **điều kiện**
nếu **đúng** đúng thì
thực hiện khối lệnh;
lặp lại kiểm tra điều kiện
nếu **sai**
thoát khỏi vòng lặp.

Giải thích:

- + Biểu thức 1: khởi tạo giá trị ban đầu cho biến điều khiển.
- + Biểu thức 2: là quan hệ logic thể hiện điều kiện tiếp tục vòng lặp.
- + Biểu thức 3: phép gán dùng thay đổi giá trị biến điều khiển.

Nhận xét:

- + Biểu thức 1 bao giờ cũng chỉ được tính toán một lần khi gọi thực hiện for.
- + Biểu thức 2, 3 và thân for có thể thực hiện lặp lại nhiều lần.

Lưu ý:

- + **Biểu thức 1, 2, 3 phải phân cách bằng dấu chấm phẩy (;)**

- + Nếu biểu thức 2 không có, vòng for được xem là luôn luôn **đúng**. Muốn thoát khỏi vòng lặp for phải dùng một trong 3 lệnh **break**, **goto** hoặc **return**.
- + Với mỗi biểu thức có thể viết thành một dãy biểu thức con phân cách nhau bởi dấu phẩy. Khi đó các biểu thức con được xác định từ trái sang phải. Tính đúng sai của dãy biểu thức con trong biểu thức thứ 2 được xác định bởi biểu thức con cuối cùng.
- + Trong thân for (khối lệnh) có thể chứa một hoặc nhiều cấu trúc điều khiển khác.
- + Khi gặp lệnh **break**, cấu trúc lặp sâu nhất sẽ thoát ra.
- + Trong thân for có thể dùng lệnh **goto** để thoát khỏi vòng lặp đến vị trí mong muốn.
- + Trong thân for có thể sử dụng **return** để trở về một hàm nào đó.
- + Trong thân for có thể sử dụng lệnh continue để chuyển đến đầu vòng lặp (bỏ qua các câu lệnh còn lại trong thân).

Ví dụ 1: Viết chương trình in ra câu "Vi du su dung vong lap for" 3 lần.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | /* Chuong trinh in ra cau "Vi du su dung vong lap for" 3 lan */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | #define MSG "Vi du su dung vong lap for.\n" |
| 7 | |
| 8 | void main(void) |
| 9 | { |
| 10 | int i; |
| 11 | for(i = 1; i<=3; i++) /hoac for(i = 1; i<=3; i+=1) |
| 12 | printf("%s", MSG); |
| 13 | getch(); |
| 14 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

| | |
|---|---|
| Ví dụ su dung vong lap for. Ví dụ su dung vong lap for. Ví dụ su dung vong lap for. | Bạn thay 2 dòng 11 và 12 bằng câu lệnh for(i=1; i<=3; i++, printf("%s", MSG)); Chạy lại chương trình, quan sát và nhận xét kết quả. |
|---|---|

Có dấu chấm phẩy sau lệnh for(i=1; i<=3; i++); → các lệnh thuộc vòng lặp for sẽ không được thực hiện.

Ví dụ 2: Viết chương trình nhập vào 3 số nguyên. Tính và in ra tổng của chúng.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Chuong trinh nhap vao 3 so va tinh tong */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | int i, in, is; |
| 9 | is = 0; |

```

10  for(i = 1; i<=3; i++)
11  {
12      printf("Nhap vao so thu %d :", i);
13      scanf("%d", &in);
14      is = is + in;
15  }
16  printf("Tong: %d", is);
17  getch();
18  }
    
```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

☞ Kết quả in ra màn hình

| | |
|---|---|
| Nhap vao so thu 1: 5 Nhap vao so thu 2: 4 Nhap vao so thu 3: 2 Tong: 11. | Bạn thay các dòng từ 9 đến 15 bằng câu lệnh: for(is=0, i=1; i<=3; printf("Nhap vao so thu %d: ", i), scanf("%d", &in), i++, is=is+in); Chạy lại chương trình, quan sát và nhận xét kết quả. |
|---|---|

☞ Trong vòng lặp for có sử dụng từ 2 lệnh trở lên, nhớ sử dụng cặp ngoặc { } để bọc các lệnh đó lại. Dòng 12, 13, 14 thuộc vòng for dòng 10 do được bọc bởi cặp ngoặc { }. Nếu 3 dòng này không bọc bởi cặp ngoặc { }, thì chỉ dòng 12 thuộc vòng lặp for, còn 2 dòng còn lại không thuộc vòng lặp for.

Ví dụ 3: Viết chương trình nhập vào số nguyên n. Tính tổng các giá trị lẻ từ 0 đến n.

```

Dòng  File Edit Search Run Compile Debug Project Option Window Help
1  /* Chuong trinh nhap vao 3 so va tinh tong */
2
3  #include <stdio.h>
4  #include <conio.h>
5
6  void main(void)
7  {
8      int i, in, is = 0;
9      printf("Nhap vao so n: ");
10     scanf("%d", &in);
11     is = 0;
12     for(i = 0; i<=in; i++)
13     {
14         if (i % 2 != 0)           //neu i la so le
15             is = is + i;         //hoac is += i;
16     }
17     printf("Tong: %d", is);
18     getch();
19 }
    
```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

☞ Kết quả in ra màn hình

| | |
|-------------------------------|---|
| Nhap vao so n : 5 Tong: 9. | Bạn thay các dòng từ 11 đến 16 bằng câu lệnh: for(is=0, i=1; i<=n; is=is+i, i+=2); Chạy lại chương trình, quan sát và nhận xét kết quả. |
|-------------------------------|---|

☞ Bạn có thể viết gộp các lệnh trong thân for vào trong lệnh for. Tuy nhiên, khi lập trình bạn nên viết lệnh for có đủ 3 biểu thức đơn và các lệnh thực hiện trong thân for mỗi lệnh một dòng để sau này có thể đọc lại dễ hiểu, dễ sửa chữa.

Ví dụ 4: Một vài ví dụ thay đổi biến điều khiển vòng lặp.

- Thay đổi biến điều khiển từ 1 đến 100, mỗi lần tăng 1:
for(i = 1; i <= 100; i++)
- Thay đổi biến điều khiển từ 100 đến 1, mỗi lần giảm 1:
for(i = 100; i >= 1; i--)
- Thay đổi biến điều khiển từ 7 đến 77, mỗi lần tăng 7:
for(i = 7; i <= 77; i += 7)
- Thay đổi biến điều khiển từ 20 đến 2, mỗi lần giảm 2:
for(i = 20; i >= 2; i -= 2)

Ví dụ 5: Đọc vào một loạt kí tự trên bàn phím. Kết thúc khi gặp dấu chấm '!'.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | /* Doc vao 1 loat ktu tren ban phim. Ket thuc khi gap dau cham */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | |
| 5 | #define DAU_CHAM ' !' |
| 6 | |
| 7 | void main(void) |
| 8 | { |
| 9 | char c; |
| 10 | for(; (c = getchar()) != DAU_CHAM;) |
| 11 | putchar(c); |
| 12 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

☞ Kết quả in ra màn hình

| | |
|---|---|
| a | Bạn thay các dòng từ 10 đến 11 bằng câu lệnh: for(; (c = getchar()) != DAU_CHAM; putchar(c)); Chạy lại chương trình, quan sát và nhận xét kết quả. |
| a | |
| 4 | |
| 4 | |
| . | |

☞ Vòng lặp for vắng mặt biểu thức 1 và 3.

Ví dụ 6: Đọc vào một loạt kí tự trên bàn phím, đếm số kí tự nhập vào. Kết thúc khi gặp dấu chấm '!'.

| Dòng | File Edit Search Run Cmpile Debug Project Option Window Help |
|------|--|
| 1 | /* Doc vao 1 loat ktu tren ban phim, dem so ktu nhap vao. Ket thuc khi gap dau cham */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | #define DAU_CHAM ' !' |
| 7 | |
| 8 | void main(void) |

| | |
|--|--|
| 9 | { |
| 10 | char c; |
| 11 | int idem; |
| 12 | for(idem = 0; (c = getchar()) != DAU_CHAM;) |
| 13 | idem++; |
| 14 | printf("So ki tu: %d.\n", idem); |
| 15 | getch(); |
| 16 | } |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | |

☞ **Kết quả in ra màn hình**

| | |
|------------------------|---|
| afser. So ki tu: 5. | Bạn thay các dòng từ 12 đến 13 bằng câu lệnh: for(idem = 0; (c = getchar()) != DAU_CHAM; idem++); Chạy lại chương trình, quan sát và nhận xét kết quả. |
|------------------------|---|

☞ **Vòng lặp for vắng mặt biểu thức 3.**

Ví dụ 7: Đọc vào một loạt kí tự trên bàn phím, đếm số kí tự nhập vào. Kết thúc khi gặp dấu chấm '.'.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|--|--|
| 1 | /* Doc vao 1 loat ktu tren ban phim, dem so ktu nhap vao. Ket thuc khi gap dau cham */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | #define DAU_CHAM '.' |
| 7 | |
| 8 | void main(void) |
| 9 | { |
| 10 | char c; |
| 11 | int idem = 0; |
| 12 | for(;;) |
| 13 | { |
| 14 | c = getchar(); |
| 15 | if (c == DAU_CHAM) //nhap vao dau cham |
| 16 | break; //thoat vong lap |
| 17 | idem++; |
| 18 | } |
| 19 | printf("So ki tu: %d.\n", idem); |
| 20 | getch(); |
| 21 | } |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | |

☞ **Kết quả in ra màn hình**

| | |
|------------------------|--|
| afser. So ki tu: 5. | Chạy lại chương trình, quan sát và nhận xét kết quả. |
|------------------------|--|

☞ **Vòng lặp for vắng mặt cả ba biểu thức.**

Ví dụ 8: Nhập vào 1 dãy số nguyên từ bàn phím đến khi gặp số 0 thì dừng. In ra tổng các số nguyên dương.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
|------|---|

```

1  /* Nhập vào 1 dãy số nguyên từ bàn phím đến khi gặp số 0 thì dừng. In ra tổng các số
2  nguyên dương */
3
4  #include <stdio.h>
5  #include <conio.h>
6
7  void main(void)
8  {
9      int in, itong = 0;
10     for(;;)
11     {
12         printf("Nhập vào 1 số nguyên: ");
13         scanf("%d", &in);
14         if (in < 0)
15             continue;    //in < 0 quay ngược lên đầu vòng lặp
16         if (in == 0)
17             break;       //in = 0 thoát vòng lặp
18         itong += in;
19     }
20     printf("Tổng: %d.\n", itong);
21     getch();
22 }

```

F1 Help **Alt-F8 Next Msg** **Alt-F7 Prev Msg** **Alt - F9 Compile** **F9 Make** **F10 Menu**

☞ Kết quả in ra màn hình

| | |
|---|---|
| Nhập vào 1 số nguyên: -8 Nhập vào 1 số nguyên: 9 Nhập vào 1 số nguyên: -7 Nhập vào 1 số nguyên: 3 Nhập vào 1 số nguyên: 0 Tổng: 12 | Chạy lại chương trình với số liệu khác Quan sát và nhận xét kết quả. |
|---|---|

6.2.2 Lệnh break

Thông thường lệnh break dùng để thoát khỏi vòng lặp không xác định điều kiện dừng hoặc bạn muốn dừng vòng lặp theo điều kiện do bạn chỉ định. Việc dùng lệnh break để thoát khỏi vòng lặp thường sử dụng phối hợp với lệnh if. Lệnh break dùng trong for, while, do...while, switch. Lệnh break thoát khỏi vòng lặp chứa nó.

Ví dụ 9 : Như ví dụ 7, 8

Sử dụng lệnh break trong switch để nhảy bỏ các câu lệnh kế tiếp còn lại.

6.2.3 Lệnh continue

Được dùng trong vòng lặp for, while, do...while. Khi lệnh continue thi hành quyền điều khiển sẽ trao qua cho biểu thức điều kiện của vòng lặp gần nhất. Nghĩa là lộn ngược lên đầu vòng lặp, tất cả những lệnh đi sau trong vòng lặp chứa continue sẽ bị bỏ qua không thi hành.

Ví dụ 10 : Như ví dụ 8

6.2.4 Lệnh while

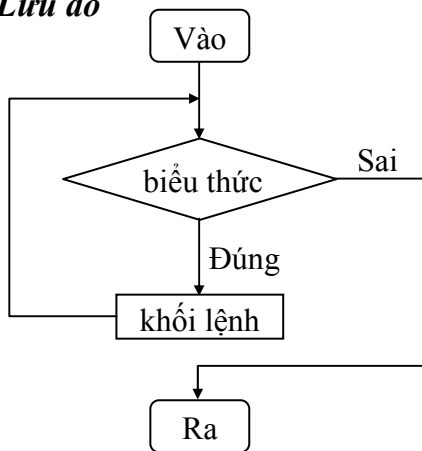
Vòng lặp thực hiện lặp lại trong khi biểu thức còn đúng.

- **Cú pháp lệnh**

**while (biểu thức)
khối lệnh;**

- ☞ từ khóa **while** phải viết bằng chữ thường
- ☞ Nếu **khối lệnh** bao gồm từ 2 lệnh trở lên thì phải đặt trong dấu { }

• Lưu đồ



- ☞ Trước tiên **biểu thức** được kiểm tra nếu **sai** thì kết thúc vòng lặp while (khối lệnh không được thi hành 1 lần nào) nếu **đúng** thực hiện khối lệnh; lặp lại kiểm tra biểu thức

+ Biểu thức: có thể là một biểu thức hoặc nhiều biểu thức con. Nếu là nhiều biểu thức con thì cách nhau bởi dấu phẩy (,) và tính đúng sai của biểu thức được quyết định bởi biểu thức con cuối cùng.

+ Trong thân while (khối lệnh) có thể chứa một hoặc nhiều cấu trúc điều khiển khác.

+ Trong thân while có thể sử dụng lệnh continue để chuyển đến đầu vòng lặp (bỏ qua các câu lệnh còn lại trong thân).

+ Muốn thoát khỏi vòng lặp while tùy ý có thể dùng các lệnh **break, goto, return** như lệnh **for**.

Ví dụ 11: Viết chương trình in ra câu "Vi dụ sử dụng vòng lặp while" 3 lần.

| | |
|------|---|
| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
| 1 | /* Chuong trinh in ra cau "Vi dụ sử dụng vòng lặp while" 3 lan */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | #define MSG "Vi dụ sử dụng vòng lặp while.\n" |
| 7 | |
| 8 | void main(void) |
| 9 | { |
| 10 | int i = 0; |
| 11 | while (i++ < 3) |
| 12 | printf("%s", MSG); |
| 13 | getch(); |
| 14 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

☞ **Kết quả in ra màn hình**

| | |
|---|---|
| Vi dụ sử dụng vòng lặp while. Vi dụ sử dụng vòng lặp while. Vi dụ sử dụng vòng lặp while. | Bạn thay 2 dòng 11 và 12 bằng câu lệnh while(printf("%s", MSG), ++i < 3); Chạy lại chương trình và quan sát kết quả. |
|---|---|

Ví dụ 12: Viết chương trình tính tổng các số nguyên từ 1 đến n, với n được nhập vào từ bàn phím.

| | |
|------|---|
| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
| 1 | /* Chuong trình tính tong cac so nguyen tu 1 den n */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | int i = 0, in, is = 0; |
| 9 | printf("Nhap vao so n: "); |
| 10 | scanf("%d", &in); |
| 11 | while (i++ < in) |
| 12 | is = is + i; //hoac is += i; |
| 13 | printf("Tong: %d", is); |
| 14 | getch(); |
| 15 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

| | |
|--------------------------------|--|
| Nhap vao so n : 5 Tong: 15. | Bạn thay các dòng từ 11 đến 12 bằng câu lệnh: while(is = is+i, i++ < in); Chạy lại chương trình, quan sát và nhận xét kết quả. |
|--------------------------------|--|

Ví dụ 13: Thay dòng `for(; (c = getchar()) != DAU_CHAM;)` ở ví dụ 5 thành dòng `while ((c = getchar()) != DAU_CHAM)`

Chạy lại chương trình, quan sát và nhận xét kết quả.

Ví dụ 14: Ở ví dụ 6, thay dòng `int dem;` thành dòng `int dem = 0;`, thay dòng `for(idem=0; (c = getchar()) != DAU_CHAM;)` thành dòng `while ((c = getchar()) != DAU_CHAM)`

Chạy lại chương trình, quan sát và nhận xét kết quả.

Ví dụ 15: Ở ví dụ 7 và 8, thay dòng `for(; ;)` thành dòng `while(1)`

Chạy lại chương trình, quan sát và nhận xét kết quả.

6.2.5 Lệnh do...while

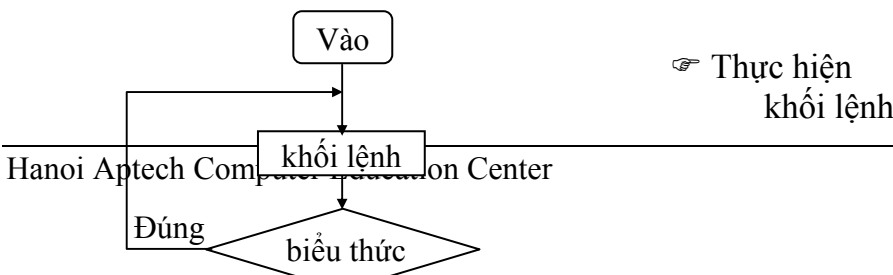
Vòng lặp thực hiện lặp lại cho đến khi biểu thức sai.

- Cú pháp lệnh**

```
do
    khối lệnh;
while (biểu thức);
```

- từ khóa **do, while** phải viết bằng chữ thường
- Nếu **khối lệnh** bao gồm từ 2 lệnh trở lên thì phải đặt trong dấu { }

- Lưu đồ**



Kiểm tra biểu thức

Nếu **đúng** thì
lặp lại thực hiện khối lệnh

Nếu **sai** thì
kết thúc vòng lặp
(khối lệnh được thi hành 1 lần)

+ Biểu thức: có thể là một biểu thức hoặc nhiều biểu thức con. Nếu là nhiều biểu thức con thì cách nhau bởi dấu phẩy (,) và tính đúng sai của biểu thức được quyết định bởi biểu thức con cuối cùng.

+ Trong thân do...while (khối lệnh) có thể chứa một hoặc nhiều cấu trúc điều khiển khác.

+ Trong thân do...while có thể sử dụng lệnh continue để chuyển đến đầu vòng lặp (bỏ qua các câu lệnh còn lại trong thân).

+ Muốn thoát khỏi vòng lặp do...while tùy ý có thể dùng các lệnh **break, goto, return**.

Ví dụ 16: Viết chương trình kiểm tra password.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | /* Chuong trinh kiem tra mat khau */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | |
| 5 | # define PASSWORD 12345 |
| 6 | |
| 7 | void main(void) |
| 8 | { |
| 9 | int in; |
| 10 | do |
| 11 | { |
| 12 | printf("Nhap vao password: "); |
| 13 | scanf("%d", &in); |
| 14 | } while (in != PASSWORD) |
| 15 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

 **Kết quả in ra màn hình**

| | |
|---|---|
| Nhap vao password: 1123 Nhap vao password: 12346 Nhap vao password: 12345 | Bạn thay các dòng từ 10 đến 14 bằng câu lệnh: do{}while(printf("Nhap vao password: "), scanf("%d", &in), in != PASSWORD); Chạy lại chương trình và quan sát kết quả. |
|---|---|

Ví dụ 17: Viết chương trình nhập vào năm hiện tại, năm sinh. In ra tuổi.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Chuong trinh in tuoi */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | |
| 5 | # define CHUC "Chuc ban vui ve (: >\n" |
| 6 | |
| 7 | void main(void) |
| 8 | { |

| | |
|---|---|
| 9 | unsigned char choi; |
| 10 | int inamhtai, inamsinh; |
| 11 | do |
| 12 | { |
| 13 | printf("Nhap vao nam hien tai: "); |
| 14 | scanf("%d", inamhtai); |
| 15 | printf("Nhap vao nam sinh: "); |
| 16 | scanf("%d", inamsinh); |
| 17 | printf("Ban %d tuoi, %s", inamhtai – inamsinh, CHUC); |
| 18 | printf("Ban co muon tiep tục? (Y/N)\n"); |
| 19 | choi = getch(); |
| 20 | } while (choi == 'y' choi == 'Y'); |
| 21 | } |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | |

☞ **Kết quả in ra màn hình**

| | |
|--|---|
| Nhap vao nam hien tai: 2002 Nhap vao nam sinh: 1980 Ban 22 tuoi, chuc ban vui ve (:> Ban co muon tiep tục? (Y/N) _ (nếu gõ y hoặc Y tiếp tục thực hiện chương trình, ngược lại gõ các phím khác chương trình sẽ thoát) | Bạn lại chương trình với số liệu khác. Quan sát, đánh giá và nhận xét kết quả. |
|--|---|

6.2.6 Vòng lặp lồng nhau

Ví dụ 18: Vẽ hình chữ nhật đặc bằng các dấu '*'

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|---|--|
| 1 | /* Ve hình chu nhật đặc */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | int i, ij, idai, irong; |
| 9 | printf("Nhap vao chieu dai: "); |
| 10 | scanf("%d", &idai); |
| 11 | printf("Nhap vao chieu rong: "); |
| 12 | scanf("%d", &irong); |
| 13 | for (i = 1; i <= irong; i++) |
| 14 | { |
| 15 | for (ij = 1; ij <= idai; ij++) //in mot hàng voi chieu dai dau * |
| 16 | printf("*"); |
| 17 | printf("\n"); //xuống dòng khi in xong 1 hàng |
| 18 | } |
| 19 | getch(); |
| 20 | } |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | |

☞ **Kết quả in ra màn hình**

| | |
|---|---|
| Nhập vào chiều dài: 10 Nhập vào chiều rộng: 5 * | Bạn lại chương trình với số liệu khác. Quan sát, đánh giá và nhận xét kết quả. |
|---|---|

Ví dụ 19: Vẽ hình chữ nhật đặc có chiều rộng = 10 hàng. Hàng thứ 1 = 10 số 0, hàng thứ 2 = 10 số 1...

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Vẽ hình chữ nhật bằng các số từ 0 đến 9 */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | int i = 0, ij; |
| 9 | while (i <= 9) |
| 10 | { |
| 11 | ij = 0; //khởi tạo lại ij = 0 cho lần in kế tiếp |
| 12 | while (ij++ <= 9) //in 1 hàng 10 số i |
| 13 | printf("%d", i); |
| 14 | printf("\n"); //xuống dòng khi in xong 1 hàng |
| 15 | i++; //tăng i lên 1 cho vòng lặp kế tiếp |
| 16 | } |
| 17 | getch(); |
| 18 | } |
| 19 | |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

👉 Kết quả in ra màn hình

| | |
|--|---|
| 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 6 6 6 6 6 6 6 6 6 6 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8 9 9 9 9 9 9 9 9 9 9 | Thay dòng 11, 12 thành câu lệnh for (ij = 0; ij <= 9; ij++) Chạy lại chương trình. Quan sát, đánh giá và nhận xét kết quả. |
|--|---|

👉 Các lệnh lặp for, while, do...while có thể lồng vào chính nó, hoặc lồng vào lẫn nhau. Nếu không cần thiết không nên lồng vào nhiều cấp để gây nhầm lẫn khi lập trình cũng như kiểm soát chương trình.

6.2.7 So sánh sự khác nhau của các vòng lặp

- Vòng lặp for thường sử dụng khi biết được số lần lặp xác định.
- Vòng lặp thường while, do...while sử dụng khi không biết rõ số lần lặp.

- Khi gọi vòng lặp while, do...while, nếu biểu thức sai vòng lặp while sẽ không được thực hiện lần nào nhưng vòng lặp do...while thực hiện được 1 lần.

☞ Số lần thực hiện ít nhất của while là 0 và của do...while là 1

6.3 Bài tập

1. *Viết chương trình in ra bảng mã ASCII*
2. *Viết chương trình tính tổng bậc 3 của N số nguyên đầu tiên.*
3. *Viết chương trình nhập vào một số nguyên rồi in ra tất cả các ước số của số đó.*
4. *Viết chương trình vẽ một tam giác cân bằng các dấu **
5. *Viết chương trình tính tổng nghịch đảo của N số nguyên đầu tiên theo công thức*

$$S = 1 + 1/2 + 1/3 + \dots + 1/N$$
6. *Viết chương trình tính tổng bình phương các số lẻ từ 1 đến N.*
7. *Viết chương trình nhập vào N số nguyên, tìm số lớn nhất, số nhỏ nhất.*
8. *Viết chương trình nhập vào N rồi tính giai thừa của N.*
9. *Viết chương trình tìm USCLN, BSCNN của 2 số.*
10. *Viết chương trình vẽ một tam giác cân rộng bằng các dấu *.*
11. *Viết chương trình vẽ hình chữ nhật rộng bằng các dấu *.*
12. *Viết chương trình nhập vào một số và kiểm tra xem số đó có phải là số nguyên tố hay không?*
13. *Viết chương trình tính số hạng thứ n của dãy Fibonacci.*
 Dãy Fibonacci là dãy số gồm các số hạng p(n) với:
 $p(n) = p(n-1) + p(n-2)$ với $n > 2$ và $p(1) = p(2) = 1$
 Dãy Fibonacci sẽ là: 1 1 2 3 5 8 13 21 34 55 89 144...
14. *Viết chương trình tính giá trị của đa thức*

$$P_n = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x^1 + a_0$$
 Hướng dẫn đa thức có thể viết lại

$$P_n = (\dots(a_n x + a_{n-1})x + a_{n-2})x + \dots + a_0$$
 Như vậy trước tiên tính $a_n x + a_{n-1}$, lấy kết quả nhân với x, sau đó lấy kết quả nhân với x cộng thêm a_{n-2} , lấy kết quả nhân với x ... n gọi là bậc của đa thức.
15. *Viết chương trình tính x^n với x, n được nhập vào từ bàn phím.*
16. *Viết chương trình nhập vào 1 số từ 0 đến 9. In ra chữ số tương ứng. Ví dụ: nhập vào số 5, in ra "Năm".*
17. *Viết chương trình phân tích một số nguyên N thành tích của các thừa số nguyên tố.*
18. *Viết chương trình lặp lại nhiều lần công việc nhập một ký tự và in ra mã ASCII của ký tự đó, khi nào nhập số 0 thì dừng.*
19. *Viết chương trình tìm ước số chung lớn nhất và bội số chung nhỏ nhất của 2 số nguyên.*
20. *Viết chương trình in lá cờ nước Mỹ.*
21. *Viết chương trình tính dân số của một thành phố sau 10 năm nữa, biết rằng dân số hiện nay là 6.000.000, tỉ lệ tăng dân số hàng năm là 1.8%.*

22. **Viết chương trình tìm các số nguyên gồm 3 chữ số sao cho tích của 3 chữ số bằng tổng 3 chữ số.** Ví dụ: $1*2*3 = 1+2+3$.

23. **Viết chương trình tìm các số nguyên a, b, c, d khác nhau trong khoảng từ 0 tới 10 thỏa mãn điều kiện $a*d*d = b*c*c*c$**

24. **Viết chương trình tính tổ hợp N chập K (với $K \leq N$)**

$$C = \frac{(N-k+1) * (N-k+2) * \dots * N}{1*2*3* \dots * k}$$

Trong đó C là một tích gồm k phần tử với phần tử thứ I là $(N-k+1)/I$. Để viết chương trình này, bạn dùng vòng lặp For với biến điều khiển I từ giá trị đầu là 1 tăng đến giá trị cuối là k kết hợp với việc nhân dồn vào kết quả C .

25. **Viết chương trình giải bài toán cổ điển sau:**

Trăm trâu, trăm cỏ
 Trâu đứng ăn năm
 Trâu nằm ăn ba,
 Ba trâu già ăn một
 Hỏi mỗi loại trâu có bao nhiêu con.

26. **Viết chương trình giải bài toán cổ điển sau:**

Vừa gà vừa chó 36 con
 Bó lại cho tròn, đếm đủ 100 chân
 Hỏi có bao nhiêu gà, bao nhiêu chó

27. **Viết chương trình in ra bảng cửu chương**

28. **Viết chương trình xác định xem một tờ giấy có độ dày 0.1 mm. Phải gấp đôi tờ giấy bao nhiêu lần để nó có độ dày 1m.**

29. **Viết chương trình tìm các số nguyên tố từ 2 đến N , với N được nhập vào.**

30. **Viết chương trình lặp đi lặp lại các công việc sau:**

- Nhập vào một ký tự trên bàn phím.
- Nếu là chữ thường thì in ra chính nó và chữ HOA tương ứng.
- Nếu là chữ HOA thì in ra chính nó và chữ thường tương ứng.
- Nếu là ký số thì in ra chính nó.
- Nếu là một ký tự điều khiển thì kết thúc chương trình

31. **Viết chương trình nhập vào x, n tính:**

$$- \sqrt{x + \sqrt{x + \dots + \sqrt{x}}} \quad (n \text{ dấu căn})$$

$$- 1 + \frac{x}{2} + \dots + \frac{x^n}{n+1}$$

32. **Viết chương trình nhập vào N số nguyên, đếm xem có bao nhiêu số âm, bao nhiêu số dương và bao nhiêu số không.**



Bài 7 :

HÀM

7.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Khái niệm, cách khai báo về hàm.
- Cách truyền tham số, tham biến, tham trị.
- Sử dụng biến cục bộ, toàn cục trong hàm.
- Sử dụng tiền xử lý #define

7.2 Nội dung

Hàm là một chương trình con thực hiện một khối công việc được lặp đi lặp lại nhiều lần trong khi chạy chương trình hoặc dùng tách một khối công việc cụ thể để chương trình đỡ phức tạp.

7.2.1 Các ví dụ về hàm

Ví dụ 1:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | #include <stdio.h> |
| 2 | #include <conio.h> |
| 3 | |
| 4 | // khai bao prototype |
| 5 | void line(); |
| 6 | |
| 7 | // ham in 1 dong dau |
| 8 | void line() |
| 9 | { |
| 10 | int i; |
| 11 | for(i = 0; i < 19; i++) |
| 12 | printf("*"); |
| 13 | printf("\n"); |
| 14 | } |
| 15 | |
| 16 | void main(void) |
| 17 | { |
| 18 | line(); |
| 19 | printf("* Minh hoa ve ham *"); |
| 20 | line(); |
| 21 | getch(); |
| 22 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

```
*****
* Minh hoa ve ham *
*****
_
```

Giải thích chương trình

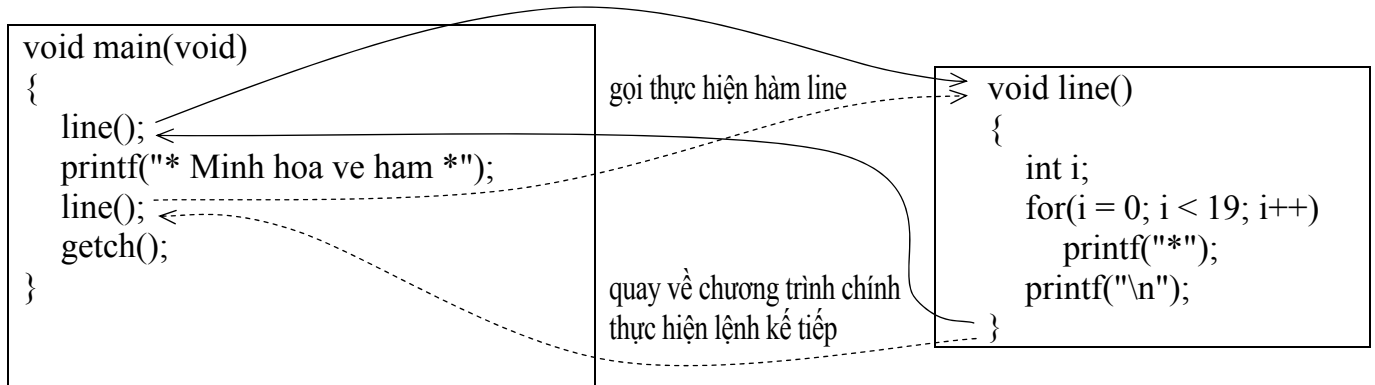
Dòng 8 đến dòng 14: định nghĩa hàm **line**, hàm này không trả về giá trị, thực hiện công việc in ra 19 dấu sao.


Dòng 5: khai báo prototype, sau tên hàm phải có dấu chấm phẩy

Trong hàm line có sử dụng biến i, biến i là biến cục bộ chỉ sử dụng được trong phạm vi hàm line.

Dòng 18 và 20: gọi thực hiện hàm line.

* Trình tự thực hiện chương trình



 **Không có dấu chấm phẩy sau tên hàm, phải có cặp dấu ngoặc () sau tên hàm nếu hàm không có tham số truyền vào. Phải có dấu chấm phẩy sau tên hàm khai báo prototype. Nên khai báo prototype cho dù hàm được gọi nằm trước hay sau câu lệnh gọi nó.**

Ví dụ 2:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | #include <stdio.h> |
| 2 | #include <conio.h> |
| 3 | |
| 4 | // khai bao prototype |
| 5 | int power(int, int); |
| 6 | |
| 7 | // ham tinh so mu |
| 8 | int power(int ix, int in) |
| 9 | { |
| 10 | int i, ip = 1; |
| 11 | for(i = 1; i <= in; i++) |
| 12 | ip *= ix; |
| 13 | return ip; |
| 14 | } |
| 15 | |
| 16 | void main(void) |
| 17 | { |
| 18 | printf("2 mu 2 = %d.\n", power(2, 2)); |
| 19 | printf("2 mu 3 = %d.\n", power(2, 3)); |
| 20 | getch(); |
| | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

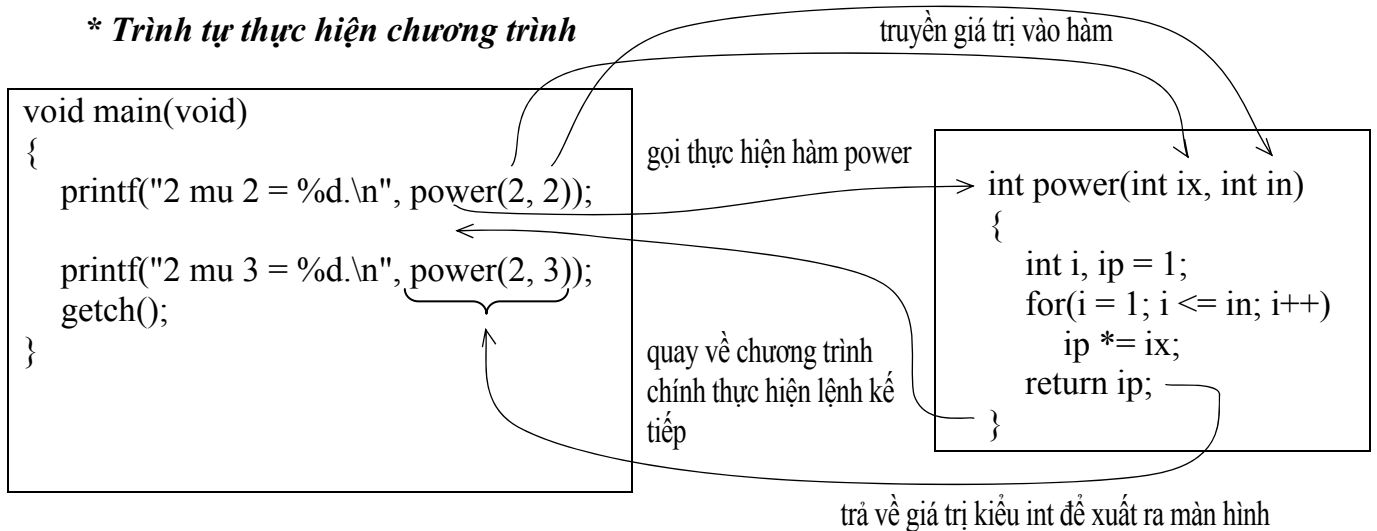
👉 Kết quả in ra màn hình

```
2 mu 2 = 4.
2 mu 3 = 8.
-
```

🔗 Giải thích chương trình

Hàm **power** có hai tham số truyền vào là ix, in có kiểu int và kiểu trả về cũng có kiểu int.
 Dòng 13: return ip: trả về giá trị sau khi tính toán
 Dòng 18: đối mục 2 và 3 có kiểu trả về là int sau khi thực hiện gọi power.
 Hai tham số ix, in của hàm power là dạng truyền tham trị.

*** Trình tự thực hiện chương trình**



👉 Quy tắc đặt tên hàm giống tên biến, hằng... Mỗi đối số cách nhau = dấu phẩy kèm theo kiểu dữ liệu tương ứng.

Ví dụ 3:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | #include <stdio.h> |
| 2 | #include <conio.h> |
| 3 | |
| 4 | // khai bao prototype |
| 5 | void time(int &, int &); |
| 6 | |
| 7 | // ham doi phut thanh gio:phut |
| 8 | void time(int &ig, int &ip) |
| 9 | { |
| 10 | ig = ip / 60; |
| 11 | ip %= 60; |
| 12 | } |
| 13 | |
| 14 | void main(void) |
| 15 | { |
| 16 | int igio, iphut; |

```

17  printf("Nhap vao so phut : ");
18  scanf("%d", &iphut);
19  time(igio, iphut);
20  printf("%02d:%02d\n", igio, iphut);
21  getch();
22  }
    
```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

👉 Kết quả in ra màn hình

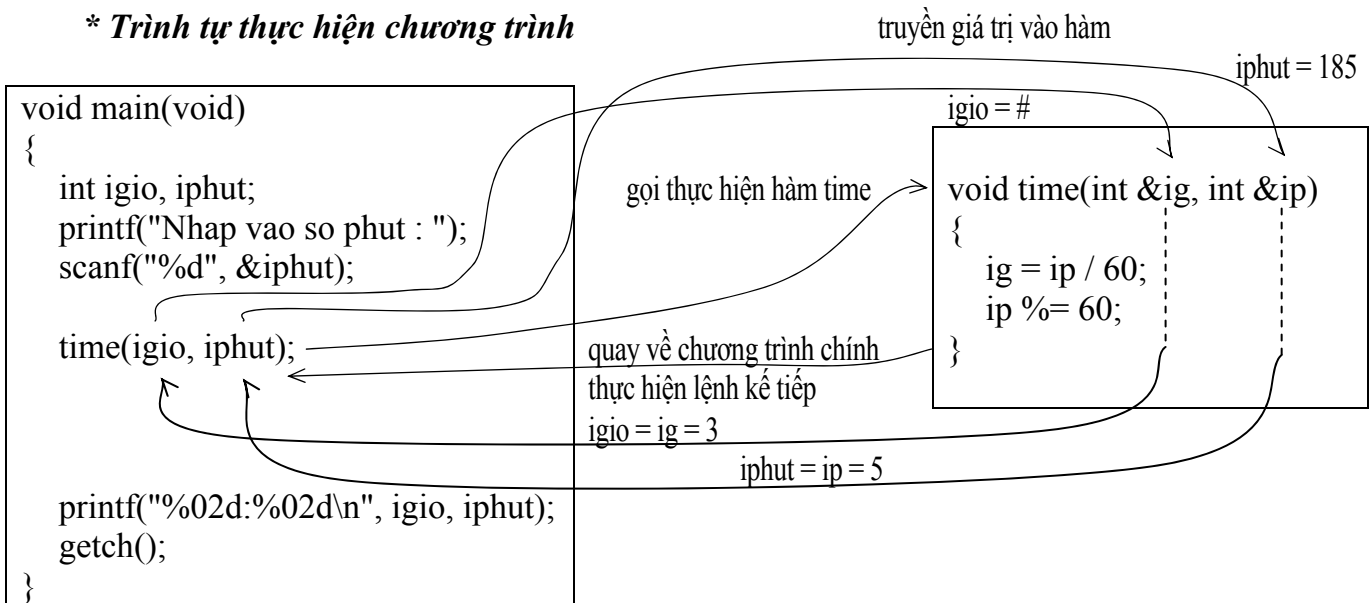
```

Nhap vao so phut: 185
03:05
_
    
```

🔗 Giải thích chương trình

Hàm **time** có hai tham số truyền vào là ix, in có kiểu int. 2 tham số này có toán tử địa chỉ & đi trước cho biết 2 tham số này là dạng truyền tham biến.

*** Trình tự thực hiện chương trình**



7.2.2 Tham số dạng tham biến và tham trị

Ví dụ 4:

| | | |
|---|---|--|
| <pre> void thamtri(int ix, int iy) { ix += 1; //cong ix them 1 iy += 1; //cong iy them 1 } void thambien(int &ix, int &iy) { ix += 1; //cong ix them 1 iy += 1; //cong iy them 1 } </pre> | <pre> void main(void) { int ia = 5, ib = 5; thamtri(ia, ib); printf("a = %d, b = %d", ia, ib); thambien(ia, ib); printf("a = %d, b = %d", ai, ib); } </pre> | <p>Kết quả in ra:</p> <p>a = 5, b = 5</p> <p>a = 6, b = 6</p> |
|---|---|--|

👉 Đối với hàm sử dụng lệnh return bạn chỉ có thể trả về duy nhất 1 giá trị mà thôi. Để có thể trả về nhiều giá trị sau khi gọi hàm bạn sử dụng hàm truyền nhiều tham số dạng tham biến.

7.2.3 Sử dụng biến toàn cục

Ví dụ 5:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | #include <stdio.h> |
| 2 | #include <conio.h> |
| 3 | |
| 4 | // khai bao prototype |
| 5 | void oddeven(); |
| 6 | void negative(); |
| 7 | |
| 8 | //khai bao bien toan cuc |
| 9 | int inum; |
| 10 | |
| 11 | void main(void) |
| 12 | { |
| 13 | printf("Nhap vao 1 so nguyen : "); |
| 14 | scanf("%d", &inum); |
| 15 | oddeven(); |
| 16 | negative(); |
| 17 | getch(); |
| 18 | } |
| 19 | |
| 20 | // hamkiem tra chan le |
| 21 | void oddeven() |
| 22 | { |
| 23 | if (inum % 2) |
| 24 | printf("%d la so le.\n", inum); |
| 25 | else |
| 26 | printf("%d la so chan.\n", inum); |
| 27 | } |
| 28 | |
| 29 | //hamkiem tra so am |
| 30 | void negative() |
| 31 | { |
| 32 | if (inum < 0) |
| 33 | printf("%d la so am.\n", inum); |
| 34 | else |
| 35 | printf("%d la so duong.\n", inum); |
| 36 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

```
Nhap vao 1 so nguyen: 3
3 la so le.
3 la so duong.
```

Giải thích chương trình

Chương trình trên gồm 2 hàm **oddeven** và **negative**, 2 hàm này bạn thấy không có tham số để truyền biến inum vào xử lý nhưng vẫn cho kết quả đúng. Do chương trình sử dụng biến **inum** toàn cục (dòng.9) nên biến này có ảnh hưởng đến toàn bộ chương trình mỗi khi gọi và sử dụng nó. Xét tình huống sau: Giả sử trong hàm **negative** ta khai báo biến inum có kiểu **int** như sau:

```
void negative()
{
    int inum;
    ....
}
```

Khi đó chương trình sẽ cho kết quả sai! Do các câu lệnh trong hàm **negative** sử dụng biến **inum** sẽ sử dụng biến **inum** khai báo trong hàm **negative** và lúc này biến **inum** toàn cục không có tác dụng đối với các câu lệnh trong hàm này. Biến **inum** khai báo trong hàm **negative** chỉ có ảnh hưởng trong phạm vi hàm và chu trình sống của nó bắt đầu từ lúc gọi hàm đến khi thực hiện xong.

⚠️ Cần thận khi đặt tên biến, xác định rõ phạm vi của biến khi sử dụng để có thể dễ dàng kiểm soát chương trình.

Ví dụ 6:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | #include <stdio.h> |
| 2 | #include <conio.h> |
| 3 | |
| 4 | #define PI 3.14 |
| 5 | |
| 6 | // khai bao prototype |
| 7 | float area(); |
| 8 | |
| 9 | //khai bao bien toan cuc |
| 10 | float frad; |
| 11 | |
| 12 | void main(void) |
| 13 | { |
| 14 | printf("Nhap vao ban kinh hinh cau : "); |
| 15 | scanf("%f", &frad); |
| 16 | printf("Dien tich hinh cau: %10.3f.\n", area()); |
| 17 | getch(); |
| 18 | } |
| 19 | |
| 20 | // ham tinh dien tich hinh cau |
| 21 | float area() |
| 22 | { |
| 23 | return (4*PI*frad*frad); |
| 24 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

⚡ Kết quả in ra màn hình

Nhap vao ban kinh hinh cau: 3.2
 Dien tich hinh cau: 128.614

7.2.4 Dùng dẫn hướng #define

Sau đây là một vài ví dụ dùng dẫn hướng #define để định nghĩa hàm đơn giản

```
#define AREA_CIRCLE (frad) (4*PI*frad*frad) //tinh dien tich hinh cau
#define SUM (x, y) (x + y) //cong 2 so
#define SQR (x) (x*x) //tinh x binh phuong
#define MAX(x, y) (x > y) ? x : y //tim so lon nhat giua x va y
#define ERROR (s) printf("%s.\n", s) //in thong bao voi chuoì s
```

Ví dụ 7: Trong ví dụ 6 xóa từ dòng 20 đến dòng 24, xóa dòng 6, 7; thêm dòng **AREA_CIRCLE (frad) (4*PI+frad*frad)** vào sau dòng 5.

Sửa dòng **printf("Dien tich hinh cau: %10.3f.\n", area());** thành **printf("Dien tich hinh cau: %10.3f.\n", AREA_CIRCLE(frad));**

Chạy lại chương trình, quan sát và nhận xét kết quả.

Ví dụ 8:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|--|---|
| 1 | #include <stdio.h> |
| 2 | #include <conio.h> |
| 3 | |
| 4 | #define MAX(x, y) (x > y) ? x : y |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | float a = 4.5, b = 6.1; |
| 9 | printf("So lon nhat la: %5.2f.\n", MAX(a, b)); |
| 10 | getch(); |
| 11 | } |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | |

Kết quả in ra màn hình

| | |
|---------------------------|--|
| So lon nhat la: 6.10 — | Thêm vào dòng 8 giá trị c = 10 Sửa lại dòng 9: MAX(a, b) thành MAX(MAX(a, b), c) Chạy lại chương trình, quan sát và nhận xét kết quả |
|---------------------------|--|

7.3 Bài tập

1. *Viết hàm tính n!*
2. *Viết hàm tính tổng S = 1+2+....+n.*
3. *Viết hàm kiểm tra số nguyên tố.*
4. *Viết hàm tính số hạng thứ n trong dãy Fibonacci.*
5. *Viết hàm tìm số lớn nhất trong 2 số.*

Bài 8 :

MẢNG VÀ CHUỖI

8.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Ý nghĩa, cách khai báo mảng, chuỗi.
- Nhập, xuất mảng, chuỗi.
- Khởi tạo mảng chuỗi.
- Một số kỹ thuật thao tác trên mảng, chuỗi.
- Dùng mảng làm tham số cho hàm.
- Một số hàm xử lý chuỗi

8.2 Nội dung

8.2.1 Mảng

Là tập hợp các phần tử có cùng dữ liệu. Giả sử bạn muốn lưu n số nguyên để tính trung bình, bạn không thể khai báo n biến để lưu n giá trị rồi sau đó tính trung bình.

Ví dụ 1 : bạn muốn tính trung bình 10 số nguyên nhập vào từ bàn phím, bạn sẽ khai báo 10 biến: a, b, c, d, e, f, g, h, i, j có kiểu int và lập thao tác nhập cho 10 biến này như sau:

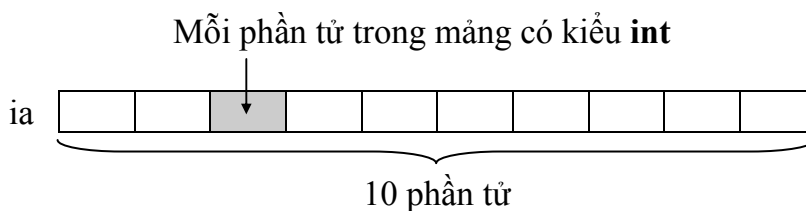
```
printf("Nhập vào biến a: ");
scanf("%d", &a);
```

10 biến bạn sẽ thực hiện 2 lệnh trên 10 lần, sau đó tính trung bình:
 $(a + b + c + d + e + f + g + h + i + j)/10$

☞ Điều này chỉ phù hợp với n nhỏ, còn đối với n lớn thì khó có thể thực hiện được. Vì vậy khái niệm mảng được sử dụng

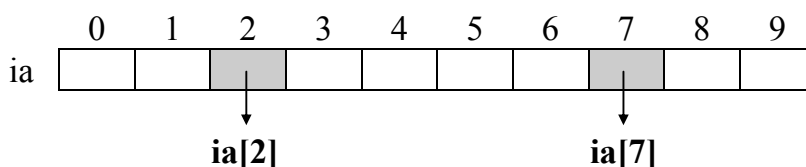
8.2.1.1 Cách khai báo mảng

Ví dụ 2 : `int ia[10];` với `int` là kiểu mảng, `ia` là tên mảng, 10 số phần tử mảng
 Ý nghĩa: *Khai báo một mảng số nguyên gồm 10 phần tử, mỗi phần tử có kiểu int.*



8.2.1.2 Tham chiếu đến từng phần tử mảng

Sau khi mảng được khai báo, mỗi phần tử trong mảng đều có chỉ số để tham chiếu. Chỉ số bắt đầu từ 0 đến n-1 (với n là kích thước mảng). Trong ví dụ trên, ta khai báo mảng 10 phần tử thì chỉ số bắt đầu từ 0 đến 9.



ia[2], ia[7]... là phần tử thứ 3, 8... trong mảng xem như là một biến kiểu **int**.

8.2.1.3 Nhập dữ liệu cho mảng

```
for (i = 0; i < 10; i++) //vòng for có giá trị i chạy từ 0 đến 9
{
    printf("Nhập vào phần tử thứ %d: ", i + 1);
    scanf("%d", &ia[i]);
}
```

8.2.1.4 Đọc dữ liệu từ mảng

```
for(i = 0; i < 10; i++)
    printf("%3d ", ia[i]);
```

Ví dụ 3 : Viết chương trình nhập vào n số nguyên. Tính và in ra trung bình cộng.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Tính trung bình cộng n số nguyên */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | int ia[50], i, in, isum = 0; |
| 9 | printf("Nhập vào giá trị n: "); |
| 10 | scanf("%d", &in); |
| 11 | |
| 12 | //Nhập dữ liệu vào mảng |
| 13 | for(i = 0; i < in; i++) |
| 14 | { |
| 15 | printf("Nhập vào phần tử thứ %d: ", i + 1); |
| 16 | scanf("%d", &ia[i]); //Nhập giá trị cho phần tử thứ i |
| 17 | } |
| 18 | |
| 19 | //Tính tổng giá trị các phần tử |
| 20 | for(i = 0; i < in; i++) |
| 21 | isum += ia[i]; //cộng dồn từng phần tử vào isum |
| 22 | |
| 23 | printf("Trung bình cộng: %.2f\n", (float) isum/in); |
| 24 | getch(); |
| 25 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

| | |
|---|---|
| Nhập vào giá trị n: 3 Nhập vào phần tử thứ 1: 7 Nhập vào phần tử thứ 2: 3 Nhập vào phần tử thứ 3: 6 Trung bình cộng: 5.33 | Bạn có thể gộp 2 lệnh for thành một vừa nhập vừa tính tổng, đưa hàng 21 sau hàng 16 và bỏ các hàng 19, 20, 21. Chạy và quan sát kết quả. |
|---|---|

| | |
|---|--|
| - | |
|---|--|

☹ Điều gì sẽ xảy ra cho đoạn chương trình trên nếu bạn nhập $n > 50$ trong khi bạn chỉ khai báo mảng ia tối đa là 50 phần tử. Bạn dùng lệnh if để ngăn chặn điều này trước khi vào thực hiện lệnh for. Thay dòng 9, 10 bằng đoạn lệnh sau :

```
do
{
    printf("Nhập vào giá trị n: ");
    scanf("%d", &in);
} while (in <= 0 || in > 50); //chỉ chấp nhận giá trị nhập vào trong khoảng 1..50
```

☞ Chạy chương trình và nhập n với các giá trị -6, 0, 51, 6. Quan sát kết quả.

8.2.1.5 Sử dụng biến kiểu khác

Ngoài kiểu int, bạn có thể khai báo mảng kiểu char, float, double...

Ví dụ 4 : char cloai[20]; float ftemp[10]; cách tham chiếu, nhập dữ liệu, đọc dữ liệu như trên.

8.2.1.6 Kỹ thuật Sentinel


Sử dụng kỹ thuật này để nhập liệu giá trị cho các phần tử mảng mà không biết rõ số lượng phần tử sẽ nhập vào là bao nhiêu (không biết số n).

Ví dụ 5 : Viết chương trình nhập vào 1 dãy số dương rồi in tổng các số dương đó.


Phác họa lời giải: Chương trình yêu cầu nhập vào dãy số dương mà không biết trước số lượng phần tử cần nhập là bao nhiêu, vì vậy để chấm dứt nhập liệu khi thỏa mãn bằng cách nhập vào số âm hoặc không.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | /* Nhập vào day so nguyen duong, in ra day chan, day le */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #define MAX 50 |
| 6 | |
| 7 | void main(void) |
| 8 | { |
| 9 | float fa[MAX], fsum = 0; |
| 10 | int i = 0; |
| 11 | do |
| 12 | { |
| 13 | printf("Nhập vào phần tử thứ %d: ", i + 1); |
| 14 | scanf("%f", &fa[i]); //Nhập giá trị cho phần tử thứ i |
| 15 | } while (fa[i++] > 0); //con nhập liệu khi giá trị phần tử > 0 |
| 16 | |
| 17 | i--; //giảm i đi 1 lần cuối cùng tăng 1 trước khi thoát |
| 18 | //Tính tổng |
| 19 | for(int ij = 0; ij < i; ij++) |
| 20 | fsum += fa[ij]; //cong don tung phần tử vào isum |
| 21 | |
| 22 | printf("Tổng : %5.2f\n", fsum); |
| 23 | getch(); |

| | | |
|----|---|---|
| 24 | } | |
| | | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

 **Kết quả in ra màn hình**

| | |
|---|---|
| Nhập vào phần tử thu 1: 1.2 Nhập vào phần tử thu 2: 3 Nhập vào phần tử thu 3: 4.6 Nhập vào phần tử thu 4: -9 Tổng : 8.80 _ | Bạn chạy lại chương trình và thử lại với số liệu khác. Quan sát kết quả. |
|---|---|

 Điều gì sẽ xảy ra cho đoạn chương trình trên nếu bạn nhập số lượng phần tử vượt quá 50 trong khi bạn chỉ khai báo mảng fa tối đa là MAX = 50 phần tử. Bạn dùng lệnh break để thoát khỏi vòng lặp do...while trước khi bước sang phần tử thứ 51. Thêm đoạn lệnh sau vào trước dòng 13:

```

if (i >= MAX)           //kiem tra phần tử bước sang 51
{
    printf("Mang da day!\n"); //thông báo "Mang da day"
    i++;                    //tăng i lên 1 do dòng 17 giảm i xuống 1
    break;                 //thoát khỏi vòng lặp do...while
}

```

 Sửa dòng 5 thành #define MAX 4. Chạy chương trình và nhập các số 1.2, 3.5, 6.5, 4. Quan sát kết quả.

8.2.1.7 Khởi tạo mảng

Ví dụ 6 : Có 4 loại tiền 1, 5, 10, 25 và 50 đồng. Hãy viết chương trình nhập vào số tiền sau đó cho biết số tiền trên gồm mấy loại tiền, mỗi loại bao nhiêu tờ.

Phác họa lời giải: Số tiền là 246 đồng gồm 4 tờ 50 đồng, 1 tờ 25 đồng, 2 tờ 10 đồng, 0 tờ 5 đồng và 1 tờ 1 đồng, Nghĩa là bạn phải xét loại tiền lớn trước, nếu hết khả năng mới xét tiếp loại kế tiếp.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | /* Nhập vào số tiền và in ra các loại 50, 25, 10, 5, 1 */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #define MAX 5 |
| 6 | |
| 7 | void main(void) |
| 8 | { |
| 9 | int itien[MAX] = {50, 25, 10, 5, 1}; //Khai báo và khởi tạo mảng với 5 phần tử |
| 10 | int i, isotien, ito; |

```

11 printf("Nhap vao so tien: ");
12 scanf("%d", &isotien); //Nhap vao so tien
13 for (i = 0; i < MAX; i++)
14 {
15     ito = isotien/itien[i]; //Tim so to cua loai tien thu i
16     printf("%4d to %2d dong\n", ito, itien[i]);
17     isotien = isotien%itien[i]; //So tien con lai sau khi da loai tru cac loai tien da co
18 }
19 getch();
20 }
    
```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

Kết quả in ra màn hình

| | |
|--|---|
| Nhap vao so tien: 246 4 tờ 50 đồng 1 tờ 25 đồng 2 tờ 10 đồng 0 tờ 5 đồng 1 tờ 1 đồng _ | Bạn chạy lại chương trình và thử lại với số liệu khác. Quan sát kết quả. |
|--|---|

⊗ Điều gì sẽ xảy nếu số phần tử mảng lớn hơn số mục, số phần tử dôi ra không được khởi tạo sẽ điền vào số 0. Nếu số phần tử nhỏ hơn số mục khởi tạo trình biên dịch sẽ báo lỗi.

Ví dụ 7 : int itien[5] = {50, 25}, phần tử itien[0] sẽ có giá trị 50, itien[1] có giá trị 25, itien[2], itien[3], itien[4] có giá trị 0.
 int itien[3] = {50, 25, 10, 5, 1} → trình biên dịch báo lỗi

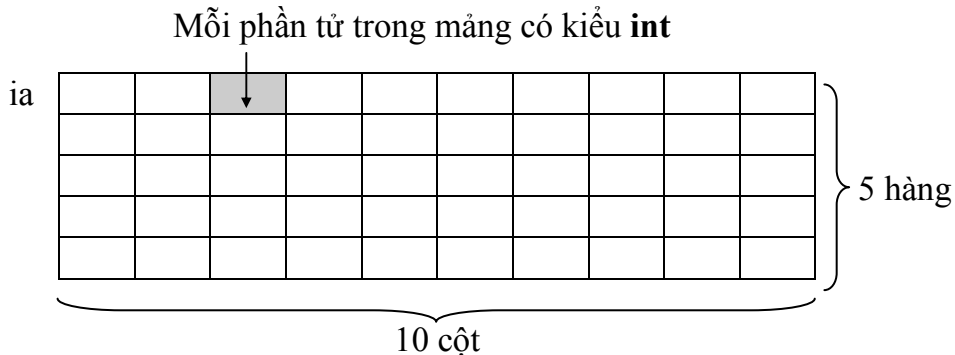
8.2.1.8 Khởi tạo mảng không bao hàm kích thước

Trong ví dụ trên giả sử ta khai báo int itien[] = {50, 25, 10, 5, 1}. Khi đó trình biên dịch sẽ đếm số mục trong danh sách khởi tạo và dùng con số đó làm kích thước mảng.

8.2.1.9 Mảng nhiều chiều

Ví dụ 8 : khai báo mảng 2 chiều int ia[5][10]; với int là kiểu mảng, ia là tên mảng, số phần tử mảng là 5 x 10.

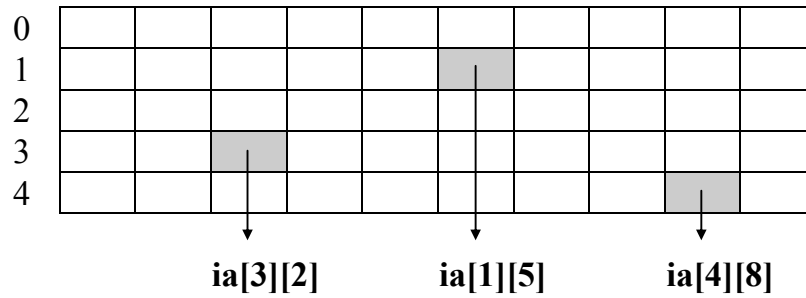
Ý nghĩa: *Khai báo một mảng 2 chiều số nguyên gồm 50 phần tử, mỗi phần tử có kiểu int.*



8.2.1.10 Tham chiếu đến từng phần tử mảng 2 chiều

Sau khi được khai báo, mỗi phần tử trong mảng 2 chiều đều có 2 chỉ số để tham chiếu, chỉ số hàng và chỉ số cột. Chỉ số hàng bắt đầu từ 0 đến số hàng - 1 và chỉ số cột bắt đầu từ 0 đến số cột - 1. Tham chiếu đến một phần tử trong mảng 2 chiều ia: ia[chỉ số hàng][chỉ số cột]

ia 0 1 2 3 4 5 6 7 8 9

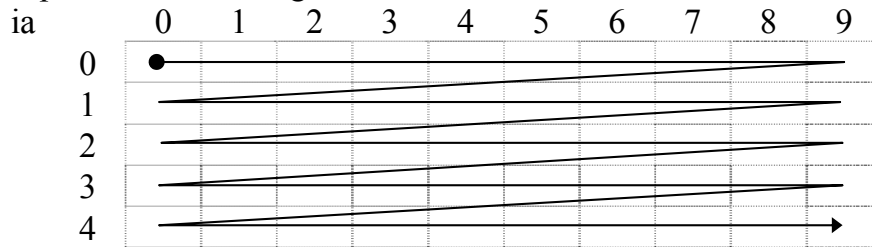


ia[3][2] là phần tử tại hàng 3 cột 2 trong mảng 2 chiều xem như là một biến kiểu **int**.

8.2.1.11 Nhập dữ liệu cho mảng 2 chiều

```
for (i = 0; i < 5; i++) //vòng for có giá trị i chạy từ 0 đến 4 cho hàng
    for (ij = 0; ij < 10; ij++) //vòng for có giá trị ij chạy từ 0 đến 9 cho cột
    {
        printf("Nhap vao phan tu ia[%d][%d]: ", i + 1, ij + 1);
        scanf("%d", &ia[i][ij]);
    }
```

* Thứ tự nhập dữ liệu vào mảng 2 chiều



8.2.1.12 Đọc dữ liệu từ mảng 2 chiều

Ví dụ 9 : in giá trị các phần tử mảng 2 chiều ra màn hình.

```
for (i = 0; i < 5; i++) //vòng for có giá trị i chạy từ 0 đến 4 cho hàng
{
    for (ij = 0; ij < 10; ij++) //vòng for có giá trị ij chạy từ 0 đến 9 cho cột
        printf("%3d ", ia[i][ij]);
    printf("\n"); //xuống dòng để in hàng kế tiếp
}
```

Ví dụ 10 : Viết chương trình nhập vào 1 ma trận số nguyên n x n. In ra ma trận vừa nhập vào và ma trận theo thứ tự ngược lại.


| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Tính trung bình công n số nguyên */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #define MAX 50; |
| 6 | |
| 7 | void main(void) |
| 8 | { |
| 9 | int ia[MAX][MAX], i, ij, in; |
| 10 | printf("Nhap vao cap ma tran: "); |
| 11 | scanf("%d", &in); |
| 12 | |
| 13 | //Nhap du lieu vao ma tran |

```

14  for (i = 0; i < in; i++)          //vòng for có giá trị i chạy từ 0 đến in-1 cho hàng
15      for (ij = 0; ij < in; ij++)    //vòng for có giá trị ij chạy từ 0 đến in-1 cho cột
16      {
17          printf("Nhap vao phan tu ia[%d][%d]: ", i + 1, ij + 1);
18          scanf("%d", &ia[i][ij]);
19      }
20
21  //In ma tran
22  for (i = 0; i < in; i++)          //vòng for có giá trị i chạy từ 0 đến in-1 cho hàng
23  {
24      for (ij = 0; ij < in; ij++)    //vòng for có giá trị ij chạy từ 0 đến in-1 cho cột
25          printf("%3d ", ia[i][ij]);
26      printf("\n");                //xuống dòng để in hàng kế tiếp
27  }
28  printf("\n");                    //Tạo khoảng cách giữa 2 ma tran
29
30  //In ma tran theo thu tu nguoc
31  for (i = in-1; i >= 0; i--)        //vòng for có giá trị i chạy từ in-1 đến 0 cho hàng
32  {
33      for (ij = in-1; ij >= 0 in; ij--) //vòng for có giá trị ij chạy từ in-1 đến 0 cho cột
34          printf("%3d ", ia[i][ij]);
35      printf("\n");                //xuống dòng để in hàng kế tiếp
36  }
37  getch();
38  }

```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

 **Kết quả in ra màn hình**

| | |
|--|---|
| <p>Nhap vao cap ma tran: 2 Nhap vao phan tu ia[1][1]: 7 Nhap vao phan tu ia[1][2]: 4 Nhap vao phan tu ia[2][1]: 6 Nhap vao phan tu ia[2][2]: 15 7 4 6 15 15 6 4 7 -</p> | <p>Chạy và thử lại chương trình với n = 3, 5. Quan sát kết quả. - Sửa lại chương trình trên cho phép nhập vào ma trận m x n. Nghĩa là ma trận có m hàng và n cột. Bạn sửa lại chương trình bằng cách cho nhập vào giá trị m và n và sửa lại vòng for cho hàng chạy m lần và vòng for cho cột chạy n lần.</p> |
|--|---|

 **Để khắc phục tình trạng người dùng nhập vào cấp ma trận > MAX, Bạn xem lại mục 3.1.4.**

8.2.1.13 Sử dụng biến kiểu khác trong mảng 2 chiều


Như mục 3.1.5.

8.2.1.14 Khởi tạo mảng 2 chiều

Ví dụ 11 : Vẽ chữ H lớn.

Dòng **File Edit Search Run Compile Debug Project Option Window Help**

| | |
|--|-------------------------------------|
| 1 | /* Chuong trinh ve chu H lon */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #define MAX 5 |
| 6 | |
| 7 | int H[MAX][MAX] = {{1, 0, 0, 0, 1}, |
| 8 | {1, 0, 0, 0, 1}, |
| 9 | {1, 1, 1, 1, 1}, |
| 10 | {1, 0, 0, 0, 1}, |
| 11 | {1, 0, 0, 0, 1}}; |
| 12 | void main(void) |
| 13 | { |
| 14 | int i , ij; |
| 15 | for (i = 0; i < MAX; i++) |
| 16 | { |
| 17 | for (ij = 0; ij < MAX; ij++) |
| 18 | if (H[i][ij]) |
| 19 | printf("!"); |
| 20 | else |
| 21 | printf(" "); |
| 22 | printf("\n"); |
| 23 | } |
| 24 | getch(); |
| 25 | } |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | |

 **Kết quả in ra màn hình**

| | |
|---------------------------------|---|
| <pre>! ! ! ! !!!! ! ! ! !</pre> | Bạn sửa lại chương trình để in ra chữ C, B... |
|---------------------------------|---|

8.2.1.15 Dùng mảng 1 chiều làm tham số cho hàm

Ví dụ 12 : Tìm số lớn nhất

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Chuong trinh tim so lon nhat su dung ham */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | #define MAX 20 |
| 7 | |
| 8 | //Khai bao prototype |
| 9 | int max(int, int); |
| 10 | |
| 11 | //ham tim so lon nhat trong mang 1 chieu |
| 12 | int max(int ia[], int in) |

```

13 {
14     int i, imax;
15     imax = ia[0];           //cho phan tu dau tien la max
16     for (i = 1; i < in; i++)
17         if (imax < ia[i]) //neu so dang xet > max
18             imax = ia[i]; //gan so nay cho max
19     return imax;          //tra ve ket qua so lon nhat
20 }
21
22 void main(void)
23 {
24     int ia[MAX];
25     int i = 0, inum;
26     do
27     {
28         printf("Nhap vao mot so: ");
29         scanf("%d", &ia[i]);
30     } while (ia[i++] != 0);
31     i--;
32     inum = max(ia, i);
33     printf("So lon nhat la: %d.\n", inum);
34     getch();
35 }

```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

☞ Kết quả in ra màn hình

| | |
|---|---|
| Nhap vao mot so: 12 Nhap vao mot so: 45 Nhap vao mot so: 3 Nhap vao mot so: 0 So lon nhat la: 45 — | Chạy lại chương trình và thử lại với số liệu khác. Thực hiện một số thay đổi sau: - Di chuyển dòng int a[MAX] ; lên sau dòng số 10 - Sửa dòng int max(int, int) ; thành int max(int) ; - Sửa dòng int max(int a[], int n) ; thành int max(int n) ; - Sửa dòng num = max(a, i) ; thành num = max(i) ; Chạy lại chương trình, quan sát, nhận xét và đánh giá kết quả. |
|---|---|

☞ Giải thích chương trình

Chương trình ban đầu hàm max có hai tham số truyền vào và kết quả trả về là giá trị max có kiểu nguyên, một tham số là mảng 1 chiều kiểu int và một tham số có kiểu int. Với chương trình sau khi sửa hàm max chỉ còn một tham số truyền vào nhưng cho kết quả như nhau. Do sau khi sửa chương trình mảng a[MAX] được khai báo lại là biến toàn cục nên hàm max không cần truyền tham số mảng vào cũng có thể sử dụng được. Tuy vậy, khi lập trình bạn nên viết như chương trình ban đầu là truyền tham số mảng vào (dạng tổng quát) để hàm max có thể thực hiện được trên nhiều mảng khác nhau. Còn với chương trình sửa lại bạn chỉ sử dụng hàm max được với mảng a mà thôi.

Ví dụ 13 : Bạn khai báo các mảng sau ia[MAX], ib[MAX], ic[MAX]. Để tìm giá trị lớn nhất của từng mảng. Bạn chỉ cần gọi hàm


- imax_a = max(ia, i);
- imax_b = max(ib, i);
- imax_c = max(ic, i);

Với chương trình sửa lại bạn không thể tìm được số lớn nhất của mảng b và c.

☞ **Bạn lưu ý rằng khi truyền mảng sang hàm, không tạo bản sao mảng mới. Vì vậy mảng truyền sang hàm có dạng tham biến. Nghĩa là giá trị của các phần tử trong mảng sẽ bị ảnh hưởng nếu có sự thay đổi trên chúng.**

Ví dụ 14 : Tìm số lớn nhất của 3 mảng a, b, c

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | /* Chương trình tìm số lớn nhất sử dụng hàm */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | #define MAX 20 |
| 7 | |
| 8 | //Khai báo prototype |
| 9 | int max(int, int); |
| 10 | int input(int); |
| 11 | |
| 12 | //hàm tìm phần tử lớn nhất trong mảng 1 chiều |
| 13 | int max(int ia[], int in) |
| 14 | { |
| 15 | int i, imax; |
| 16 | imax = ia[0]; //cho phần tử đầu tiên là max |
| 17 | for (i = 1; i < in; i++) |
| 18 | if (max < ia[i]) //nếu số đang xét > max |
| 19 | max = ia[i]; //gán số này cho max |
| 20 | return imax; //trả về kết quả số lớn nhất |
| 21 | } |
| 22 | |
| 23 | //hàm nhập liệu vào mảng 1 chiều |
| 24 | int input(int ia[]) |
| 25 | { |
| 26 | int i = 0; |
| 27 | do |
| 28 | { |
| 29 | printf("Nhập vào một số: "); |
| 30 | scanf("%d", &ia[i]); |
| 31 | } while (ia[i++] != 0); |
| 32 | i--; |
| 33 | return i; |
| 34 | } |
| 35 | |
| 36 | void main(void) |
| 37 | { |
| 38 | int ia[MAX], ib[MAX], ic[MAX]; |
| 39 | int inum1, inum2, inum3; |
| 40 | printf("Nhập liệu cho mảng a: \n"); |
| 41 | inum1 = max(ia, input(ia)); |
| 42 | printf("Nhập liệu cho mảng b: \n"); |
| 43 | inum2 = max(ib, input(ib)); |
| 44 | printf("Nhập liệu cho mảng c: \n"); |
| 45 | inum3 = max(ic, input(ic)); |
| 46 | printf("Số lớn nhất của mảng a: %d, b: %d, c: %d.\n", inum1, inum2, inum3); |
| 47 | getch(); |
| 48 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

 **Kết quả in ra màn hình**

| | |
|---|--|
| Nhap lieu cho mang a: Nhap vao mot so: 12 Nhap vao mot so: 45 Nhap vao mot so: 3 Nhap vao mot so: 0 Nhap lieu cho mang b: Nhap vao mot so: 5 Nhap vao mot so: 15 Nhap vao mot so: 0 | Nhap lieu cho mang c: Nhap vao mot so: 1 Nhap vao mot so: 5 Nhap vao mot so: 4 Nhap vao mot so: 0 So lon nhat cua mang a: 45, b: 15, c: 5. — |
| Chạy lại chương trình và thử lại với số liệu khác. Viết thêm hàm tìm số nhỏ nhất. | |

Giải thích chương trình

Hàm input có kiểu trả về là int thông qua biến i (cho biết số lượng phần tử đã nhập vào) và 1 tham số là mảng 1 chiều kiểu int. Dòng 41, 43, 45 lần lượt gọi hàm input với các tham số là mảng a, b, c. Khi hàm input thực hiện việc nhập liệu thì các phần tử trong mảng cũng được cập nhật theo.

8.2.1.16 Dùng mảng 2 chiều làm tham số cho hàm

Ví dụ 15 : Nhập vào 2 ma trận vuông cấp n số thập phân. Cộng 2 ma trận này lưu vào ma trận thứ 3 và tìm số lớn nhất trên ma trận thứ 3.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* cong ma tran */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | #define MAX 20 |
| 7 | |
| 8 | //Khai bao prototype |
| 9 | void input(float); |
| 10 | void output(float); |
| 11 | void add(float, float, float); |
| 12 | float max(float); |
| 13 | |
| 14 | //khai bao bien toan cuc |
| 15 | int in; |
| 16 | |
| 17 | //ham tim so lon nhat trong mang 2 chieu |
| 18 | float max(float fa[][MAX]) |
| 19 | { |
| 20 | float fmax; |
| 21 | fmax = fa[0][0]; //cho phan tu dau tien la max |
| 22 | for (int i = 0; i < in; i++) |
| 23 | for (int ij = 0; ij < in; ij++) |
| 24 | if (fmax < fa[i][ij]) //neu so dang xet > max |
| 25 | fmax = fa[i][ij]; //gan so nay cho max |
| 26 | return fmax; //tra ve ket qua so lon nhat |
| 27 | } |
| 28 | |
| 29 | //ham nhap lieu mang 2 chieu |

```

30 void input(float fa[][MAX])
31 {
32     for (int i = 0; i < in; i++)
33         for (int ij = 0; ij < in; ij++)
34             {
35                 printf("Nhap vao ptu[%d][%d]: ", i, ij);
36                 scanf("%f", &fa[i, j]);
37             }
38 }
39
40 //ham in mang 2 chieu ra man hinh
41 void output(float fa[][MAX])
42 {
43     for (int i = 0; i < in; i++)
44         {
45             for (int ij = 0; ij < n; ij++)
46                 printf("%5.2f", fa[i][ij]);
47             printf("\n");
48         }
49 }
50
51 //ham cong 2 mang 2 chieu
52 void add(float fa[][MAX], float fb[][MAX], float fc[][MAX])
53 {
54     for (int i = 0; i < in; i++)
55         for (int ij = 0; ij < in; ij++)
56             fc[i, ij] = fa[i, ij] + fb[i, ij];
57 }
58
59 void main(void)
60 {
61     float fa[MAX][MAX], fb[MAX][MAX], fc[MAX][MAX];
62     printf("Nhap vao cap ma tran: ");
63     scanf("%d", &in);
64     printf("Nhap lieu ma tran a: \n");
65     input(fa);
66     printf("Nhap lieu ma tran b: \n");
67     input(fb);
68     printf("Nhap lieu ma tran c: \n");
69     input(fc);
70     add(fa, fb, fc);
71     printf("Ma tran a: \n");
72     output(fa);
73     printf("Ma tran b: \n");
74     output(fb);
75     printf("Ma tran c: \n");
76     output(fc);
77     printf("So lon nhat cua ma tran c la: %5.2f.\n", max(fc));
78     getch();
79 }

```

F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

 **Kết quả in ra màn hình**

| | |
|--------------------------|-------------------------------------|
| Nhap vao cap ma tran : 2 | Ma tran a: |
| Nhap lieu ma tran a: | 5.20 4.00 |
| Nhap vao ptu[0][0] : 5.2 | 7.10 9.00 |
| Nhap vao ptu[0][1] : 4 | Ma tran b: |
| Nhap vao ptu[1][0] : 7.1 | 12.00 3.40 |
| Nhap vao ptu[1][1] : 9 | 9.60 11.00 |
| Nhap lieu ma tran b: | Ma tran c: |
| Nhap vao ptu[0][0] : 12 | 17.20 7.40 |
| Nhap vao ptu[0][1] : 3.4 | 16.70 20.00 |
| Nhap vao ptu[1][0] : 9.6 | So lon nhat cua ma tran c la: 20.00 |
| Nhap vao ptu[1][1] : 11 | - |

Chạy lại chương trình và thử lại với số liệu khác.

Viết thêm hàm tìm số nhỏ nhất.

Giải thích chương trình

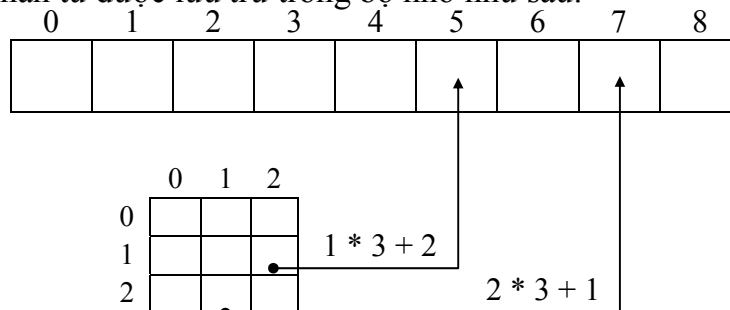
Trong chương trình khai báo biến in toàn cục do biến này sử dụng trong suốt quá trình chạy chương trình. Tham số truyền vào hàm là mảng hai chiều dưới dạng a[][MAX] vì hàm không dành chỗ cho mảng, hàm chỉ cần biết số cột để tham khảo đến các phần tử.

Ví dụ 16 : Mảng 2 chiều được khai báo int ia[3][3]

Truyền tham số vào hàm: ia[][3],
 để tham khảo đến ptu[2][1],
 hàm tính như sau:

$$2 * 3 + 1 = 7 \text{ (chỉ số hàng * số cột + chỉ số cột)}$$

ia[3][3] gồm 9 phần tử được lưu trữ trong bộ nhớ như sau:



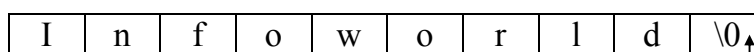
Giống như mảng 1 chiều khi truyền mảng 2 chiều sang hàm cũng không tạo bản sao mới.

8.2.2 Chuỗi

Chuỗi được xem như là một mảng 1 chiều gồm các phần tử có kiểu char như mẫu tự, con số và bất cứ ký tự đặc biệt như +, -, *, /, \$, #...

Theo quy ước, một chuỗi sẽ được kết thúc bởi ký tự **null** ('\0' : kí tự rỗng).

Ví dụ: chuỗi "Infoworld" được lưu trữ như sau:



Kí tự kết thúc chuỗi


8.2.2.1 Cách khai báo chuỗi

Ví dụ 17 : char cname[30];


Ý nghĩa: **Khai báo chuỗi cname có chiều dài 30 kí tự.** Do chuỗi kết thúc bằng kí tự null, nên khi bạn khai báo chuỗi có chiều dài 30 kí tự chỉ có thể chứa 29 kí tự.

Ví dụ 18 : Nhập vào in ra tên

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | /* Chuong trinh nhap va in ra ten*/ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | char cname[30]; |
| 9 | printf("Cho biet ten cua ban: "); |
| 10 | scanf("%s", cname); |
| 11 | printf("Chao ban %s\n", cname); |
| 12 | getch(); |
| 13 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

 **Kết quả in ra màn hình**

| | |
|---|--|
| Cho biet ten cua ban: Minh Chao ban Minh | Chạy lại chương trình và thử nhập tên: Mai Lan, Thanh Nhi Quan sát kết quả. |
|---|--|

 **Lưu ý: không cần sử dụng toán tử địa chỉ & trong cname trong lệnh scanf("%s", fname), vì bản thân fname đã là địa chỉ.**

Dùng hàm scanf để nhập chuỗi có hạn chế như sau: Khi bạn thử lại chương trình trên với dữ liệu nhập vào là Mai Lan, nhưng khi in ra bạn chỉ nhận được Mai. Vì hàm scanf nhận vào dữ liệu đến khi gặp khoảng trắng thì kết thúc.

8.2.2.2 Hàm nhập (gets), xuất (puts) chuỗi


Sử dụng hàm gets, puts phải khai báo #include <stdio.h>

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Chuong trinh nhap va in ra ten*/ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | char cname[30]; |
| 9 | puts("Cho biet ten cua ban: "); |
| 10 | gets(cname); |
| 11 | puts("Chao ban "); |
| 12 | puts(cname); |
| 13 | getch(); |
| 14 | } |

| | | | | | |
|---------|-----------------|-----------------|------------------|---------|----------|
| F1 Help | Alt-F8 Next Msg | Alt-F7 Prev Msg | Alt - F9 Compile | F9 Make | F10 Menu |
|---------|-----------------|-----------------|------------------|---------|----------|

 **Kết quả in ra màn hình**

| | |
|--|--|
| Cho biet ten cua ban: Mai Lan Chao ban Mai Lan _ | Sửa dòng 9 thành <code>printf("Cho biet ten cua ban: ");</code> và từ dòng 11 đến 12 thành <code>printf("Chao ban %s.\n", cname);</code> Chạy lại chương trình vào thử nhập tên: Tuan Anh, Thanh Lan Quan sát kết quả. |
|--|--|


 **Đối với hàm puts kí tự kết thúc chuỗi null (0) được thay thế bằng kí tự newline (\n). Hàm gets và puts chỉ có 1 đối số và không sử dụng dạng thức trong nhập liệu cũng như xuất ra màn hình.**

8.2.2.3 Khởi tạo chuỗi

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | <code>/* Chuong trinh nhap va in ra ten*/</code> |
| 2 | |
| 3 | <code>#include <stdio.h></code> |
| 4 | <code>#include <conio.h></code> |
| 5 | |
| 6 | <code>void main(void)</code> |
| 7 | <code>{</code> |
| 8 | <code>char cname[30];</code> |
| 9 | <code>char chao[] = "Chao ban";</code> |
| 10 | <code>printf("Cho biet ten cua ban: ");</code> |
| 11 | <code>gets(cname);</code> |
| 12 | <code>printf("%s %s.\n", chao, cname);</code> |
| 13 | <code>getch();</code> |
| 14 | <code>}</code> |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

 **Kết quả in ra màn hình**

| | |
|--|---|
| Cho biet ten cua ban: Mai Lan Chao ban Mai Lan _ | Chạy lại chương trình vào thử nhập tên: Doan Trang Quan sát kết quả. |
|--|---|

 **Chiều dài tối đa của chuỗi khởi tạo bằng số kí tự + 1 (kí tự null). Với chuỗi chao có chiều dài là 9.**

8.2.2.4 Mảng chuỗi

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | <code>/* Chuong trinh nhap thang (so) và in ra thang (chu) tuong ung*/</code> |
| 2 | |
| 3 | <code>#include <stdio.h></code> |
| 4 | <code>#include <conio.h></code> |
| 5 | |
| 6 | <code>void main(void)</code> |
| 7 | <code>{</code> |
| 8 | <code>char cthang[12][15] = {"January", "February", "March", "April",</code> |
| 9 | <code>"May", "June", "July", "August", "September",</code> |

| | |
|---|-------------------------------------|
| 10 | "October", "November", "December"}; |
| 11 | int ithang; |
| 12 | printf("Nhap vao thang (1-12): "); |
| 13 | scanf("%d", &ithang); |
| 14 | printf("%s.\n", cthang[ithang-1]); |
| 15 | getch(); |
| 16 | } |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | |

 **Kết quả in ra màn hình**

| | |
|--------------------------------------|--|
| Nhap vao thang (1-12): 2 February | Chạy lại chương trình vào thử nhập vào các tháng khác Quan sát kết quả. |
|--------------------------------------|--|

8.3 Bài tập

- Viết hàm tìm số lớn nhất, nhỏ nhất trong một mảng n số nguyên.
- Viết hàm sắp xếp tăng dần, giảm dần của một dãy số cho trước.
- Viết hàm tách tên và họ lót từ một chuỗi cho trước.
- Viết hàm cắt bỏ khoảng trắng thừa ở giữa, hai đầu.
- Viết hàm chuyển đổi 1 chuỗi sang chữ thường và 1 hàm chuyển đổi sang chữ HOA.
- Viết hàm chuyển đổi 1 chuỗi sang dạng Title Case (kí tự đầu của mỗi từ là chữ HOA, các kí tự còn lại chữ thường)
- Viết chương trình nhập vào 1 chuỗi và in ra chuỗi đảo ngược.
Ví dụ: Nhập vào chuỗi "Lap trinh C can ban"
In ra "nab nac C hnirt paL"
- Viết chương trình nhập vào một chuỗi ký tự rồi đếm xem trong chuỗi đó có bao nhiêu chữ 'th'.
- Biết rằng năm 0 là năm Canh thân (năm kỵ nhau có chu kì là 3, năm hợp nhau có chu kì là 4). Hãy viết chương trình cho phép gõ vào năm dương lịch (ví dụ 1997), xuất ra năm âm lịch (Đinh sừu) và các năm kỵ và hợp.
Có 12 chi: Tý, Sửu, Dần, Mão, Thìn, Tỵ, Ngọ, Mùi, Thân, Dậu, Tuất, Hợi.
Có 10 can: Giáp, Ất, Bính, Đinh, Mậu, Kỷ, Canh, Tân, Nhâm, Quý.
- Viết chương trình nhập vào 3 chữ số (305, 6, 28). Cho biết dòng chữ mô tả giá trị con số đó. Ví dụ 305 -> ba trăm lẻ năm.
- Viết chương trình nhập vào một chuỗi sau đó in ra màn hình mỗi dòng là một từ. Ví dụ chuỗi "Lap trinh C". Kết quả in ra
Lap
trinh
C
- Viết chương trình nhập vào một chuỗi các ký tự, ký số, khoảng trắng và dấu chấm câu. Cho biết chuỗi trên gồm bao nhiêu từ.
- Viết chương trình nhập vào một chuỗi ký tự. Kiểm tra xem chuỗi đó có đối xứng không?

14. Viết chương trình nhập vào một chuỗi gồm các chữ cái (a -> z, A -> Z). Hãy đếm xem có bao nhiêu nguyên âm a, i, e, o, u.

15. Giả sử số phòng trong một khách sạn được cho bởi hằng số NUM_ROOM. Viết:

- Một khai báo dãy thích hợp để theo dõi phòng nào còn trống.
- Một hàm tìm phòng nào còn trống.
- Viết chương trình đơn giản để quản lý phòng khách sạn theo dạng một trình đơn chọn công việc gồm có 4 mục như sau:
 - Tìm phòng trống.
 - Trả phòng.
 - Liệt kê những phòng còn trống.
 - Liệt kê những phòng đã thuê.
 - Kết thúc.

16. Viết chương trình mô tả văn bản của một bức điện tín. Nhập liệu bao gồm 1 hay nhiều dòng chứa một số từ, mỗi từ cách nhau khoảng trắng. In ra hóa đơn tính tiền với mỗi từ giá 100 đồng, phí trả thêm 50 đồng cho từ dài quá 8 kí tự. Hóa đơn có dạng sau:

| | | |
|---------------------------------|-------------|-----------|
| So tu | : 10 | |
| So tu co kích thước bình thường | : 8 x 100 = | 800 đồng |
| So tu có kích thước > 8 kí tự | : 2 x 150 = | 300 đồng |
| Tổng cộng | : | 1100 đồng |

17. Viết chương trình thống kê xem có bao nhiêu người họ "Ly", "Tran"... trong 1 danh sách cho trước. Nếu không có thông báo "Không có người nào thuộc họ".

18. Viết chương trình nhập vào 1 chuỗi, sau đó chép sang chuỗi khác một chuỗi con từ chuỗi ban đầu có số kí tự chỉ định.

Ví dụ: Chuỗi ban đầu "Le Thuy Doan Trang". Nếu số kí tự chỉ định là 2 thì chuỗi đích sẽ là "Le"

19. Viết chương trình nhập vào 1 chuỗi, sau đó chép sang chuỗi khác một chuỗi con từ chuỗi ban đầu với vị trí bắt đầu và số kí tự chỉ định.

Ví dụ: Chuỗi ban đầu "Le Thuy Doan Trang". Nếu vị trí ban đầu là 14 và số kí tự chỉ định là 5 thì chuỗi đích sẽ là "Trang"

20. Viết chương trình nhập vào 1 chuỗi nguồn, ví dụ "Nguyen Minh Long", sau đó nhập vào 1 chuỗi con, ví dụ "Minh", chương trình sẽ xác định vị trí bắt đầu của chuỗi con ở vị trí nào trong chuỗi nguồn. Kết quả in ra màn hình như sau:

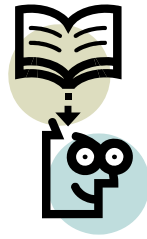
- Chuoi nguồn là : Nguyen Minh Long
- Chuoi con là : Minh
- Vị trí bắt đầu của chuỗi con là : 8

21. Viết chương trình thực hiện các yêu cầu sau:

- Nhập vào 1 chuỗi bất kỳ, ví dụ : "Nguyen Minh Long
- Muốn xóa từ vị trí nào, ví dụ : 8
- Muốn xóa bao nhiêu kí tự, ví dụ : 5

Kết quả in ra màn hình:

- Chuoi nguồn là : Nguyen Minh Long
- Chuoi sau khi xóa : Nguyen Long



Bài 9 : CON TRỎ

9.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Ý nghĩa, cách khai báo con trỏ
- Sử dụng con trỏ trong mảng, chuỗi
- Truyền mảng và chuỗi giữa các hàm qua con trỏ
- Xử lý mảng dễ dàng qua con trỏ

9.2 Nội dung

9.2.1 Con trỏ?

Con trỏ dùng để truy cập biến thông qua địa chỉ biến và chương trình tham khảo biến gián tiếp qua địa chỉ này.

9.2.2 Khái báo biến con trỏ

Ví dụ 1:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Cong hang so */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | int ix = 6, iy = 7; |
| 9 | int *px, *py; |
| 10 | printf("x = %d, y = %d\n", ix, iy); |
| 11 | px = &ix; |
| 12 | py = &iy; |
| 13 | *px += 10; |
| 14 | *py += 10; |
| 15 | printf("x = %d, y = %d\n", ix, iy); |
| 16 | getch(); |
| 17 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

| | |
|-------------------------------------|--|
| x = 6, y = 7 x = 16, y = 17 — | Chạy chương trình và quan sát kết quả. |
|-------------------------------------|--|

Giải thích chương trình

Khai báo ở dòng 9 cấp phát 2 bytes để lưu giữ địa chỉ của biến nguyên và vùng nhớ đó có tên là px, tương tự cho py. Dấu * cho biết biến này chứa địa chỉ chứ không phải giá trị, int cho biết địa chỉ đó sẽ trỏ đến các biến nguyên.

☞ Ví dụ trên cho thấy *px và *py là 2 biến con trỏ trỏ đến địa chỉ của 2 biến ix và iy (dòng 11 và 12), vì vậy khi nội dung của biến con trỏ *px và *py thay đổi thì nội dung của ix, iy cũng thay đổi theo.

| | | |
|------------------|---|----|
| Địa chỉ vùng nhớ | | |
| 1203 | 6 | ix |
| 1204 | | |
| 1205 | | |
| 1206 | | |
| 1207 | 7 | iy |
| 1208 | | |
| 1209 | | |
| 1210 | | |

Sau phép gán px = &ix và py = &iy thì giá trị của px = 1203 và py = 1207

| | | |
|------------------|------|----|
| Địa chỉ vùng nhớ | | |
| 2015 | | |
| 2016 | | |
| 2017 | 1203 | px |
| 2018 | | |
| 2019 | 1207 | py |
| 2020 | | |
| 2021 | | |
| 2022 | | |

*px là nội dung của px và *py là nội dung của py. Nên khi thực hiện 2 phép cộng gán *px += 10 và *py += 10 thì giá trị của ix sẽ là 16 và giá trị iy sẽ là 17.


9.2.3 Truyền địa chỉ sang hàm

Ví dụ 2:

| Dòng | File | Edit | Search | Run | Compile | Debug | Project | Option | Window | Help |
|--|--|------|--------|-----|---------|-------|---------|--------|--------|------|
| 1 | /* Khoi tao 2 so */ | | | | | | | | | |
| 2 | | | | | | | | | | |
| 3 | #include <stdio.h> | | | | | | | | | |
| 4 | #include <conio.h> | | | | | | | | | |
| 5 | | | | | | | | | | |
| 6 | void init (int, int); | | | | | | | | | |
| 7 | | | | | | | | | | |
| 8 | void main(void) | | | | | | | | | |
| 9 | { | | | | | | | | | |
| 10 | int ix, iy; | | | | | | | | | |
| 11 | init(&ix, &iy); | | | | | | | | | |
| 12 | printf("x = %d, y = %d\n", ix, iy); | | | | | | | | | |
| 13 | getch(); | | | | | | | | | |
| 14 | } | | | | | | | | | |
| 15 | | | | | | | | | | |
| 16 | void init(int *px, int *py) | | | | | | | | | |
| 17 | { | | | | | | | | | |
| 18 | *px = 3; //gan 3 cho noi dung cua px | | | | | | | | | |
| 19 | *py = 5; //gan 5 cho noi dung cua py | | | | | | | | | |
| 20 | } | | | | | | | | | |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | | | | | | | | | | |

 **Kết quả in ra màn hình**

| | |
|-------------------|--|
| x = 3, y = 5 — | Chạy chương trình và quan sát kết quả. |
|-------------------|--|

 **Giải thích chương trình**

Ở dòng 9, gọi hàm init truyền 2 tham số là địa chỉ của biến ix và iy, nên khi nội dung của 2 biến con trỏ *px và *py thay đổi thì ix và iy của chương trình chính cũng thay đổi theo. Hàm main(void) đã sử dụng cách truy cập biến khác với hàm init, hàm main(void) gọi chúng là ix, iy còn hàm init gọi chúng là *px, *py. Hàm init đọc giá trị của biến con trỏ *px, *py từ vùng địa chỉ của chương trình gọi, sau khi thực hiện và trả kết quả về chương trình gọi.

9.2.4 Con trỏ và mảng

Ví dụ 3: Khai báo mảng sau int num[] = {23, 54, 16, 72, 16, 84};

Như đã nghiên cứu cách tham chiếu đến phần tử mảng thứ 5 là num[4], còn với kiểu con trỏ là *(num + 4). Nghĩa là num[4] tương đương với *(num + 4) và cách truy cập nội dung như nhau. Tương tự như vậy, cách tham khảo địa chỉ của phần tử mảng là &num[4] tương đương với num + 4.

9.2.5 Con trỏ trỏ đến mảng trong hàm

Ví dụ 4:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Cong hang so vao mang */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | #define SIZE 4 |
| 7 | |
| 8 | add(int *, int, int); |
| 9 | |
| 10 | void main(void) |
| 11 | { |
| 12 | int iarray[] = {2, 5, 6, 9}; |
| 13 | int i, ix = 10; |
| 14 | add(iarray, SIZE, ix); |
| 15 | for (i = 0; i < SIZE; i++); |
| 16 | printf("%d ", *(iarray + i)); |
| 17 | getch(); |
| 18 | } |
| 19 | |
| 20 | void add(int *ptr, int inum, int ia) |
| 21 | { |
| 22 | int ij; |
| 23 | for (ij = 0; ij < inum; ij++) |
| 24 | *(ptr) = *(ptr++) + ia; |
| 25 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

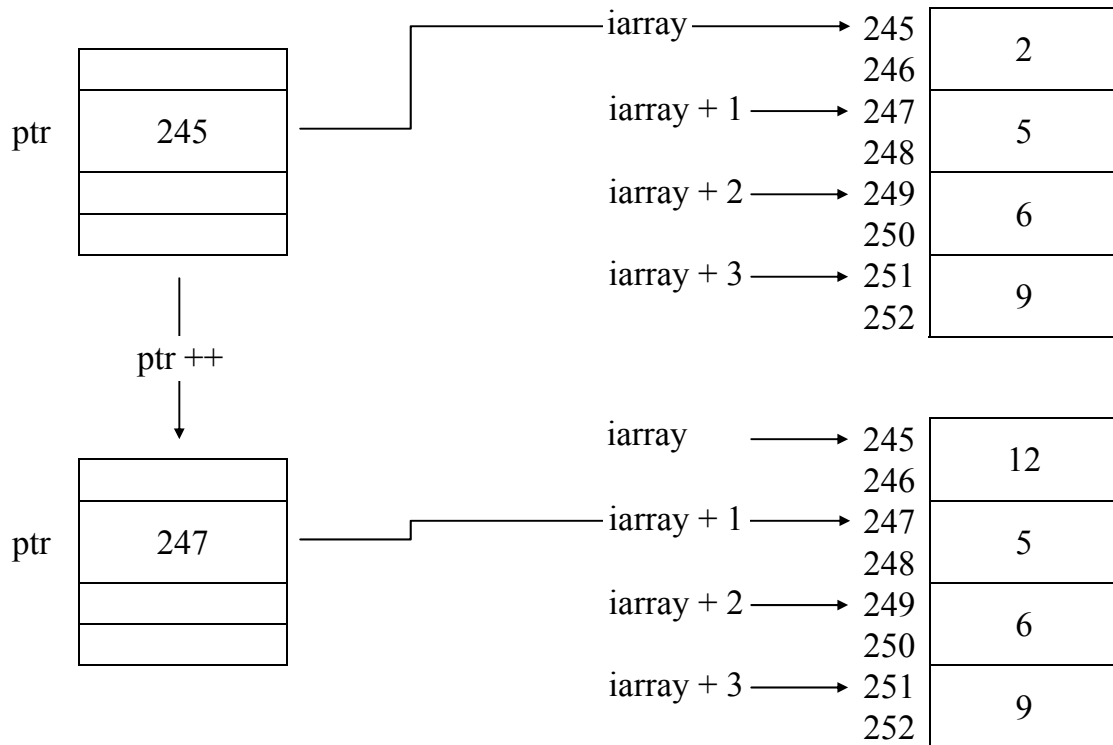
 **Kết quả in ra màn hình**

| | |
|------------------|--|
| 12 15 16 19 — | Chạy chương trình và quan sát kết quả. |
|------------------|--|

Giải thích chương trình

Hàm gán địa chỉ của mảng vào biến con trỏ ptr, kích thước vào biến inum và hằng số vào biến ia. Sau đó dùng vòng lặp để cộng hằng vào từng phần tử của mảng.

Giả sử địa chỉ của ia là 245 khi truyền vào hàm add qua ptr, ptr sẽ có giá trị = 245



9.2.6 Con trỏ và chuỗi

Ví dụ 5:

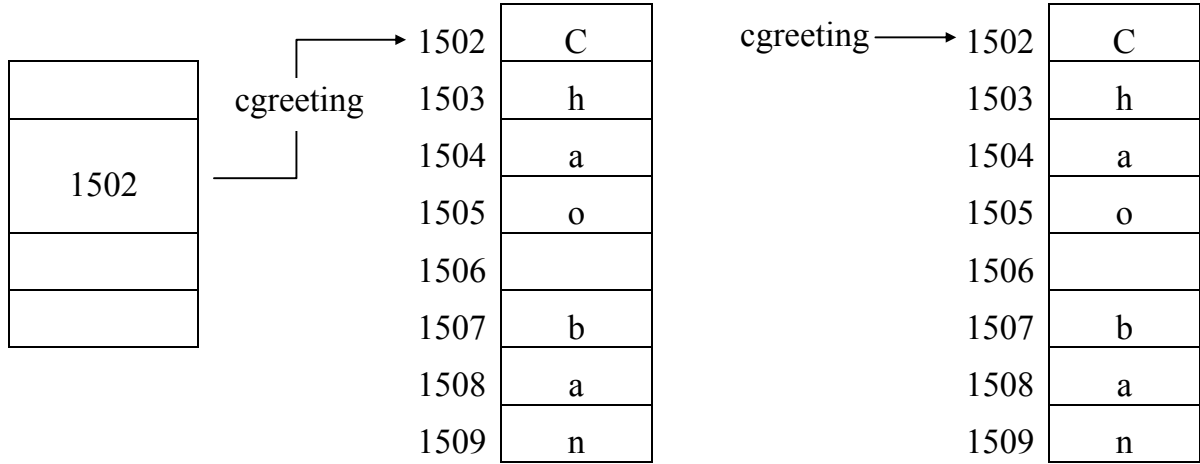
| | |
|------|---|
| Dòng | <pre> File Edit Search Run Compile Debug Project Option Window Help 1 /* Chuong trinh nhap va in ra ten*/ 2 3 #include <stdio.h> 4 #include <conio.h> 5 6 void main(void) 7 { 8 char *cgreeting = "Chao ban"; 9 char cname[30]; 10 puts("Cho biet ten cua ban: "); 11 gets(cname); 12 puts(cgreeting); 13 puts(cname); 14 getch(); 15 } </pre> |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

| | |
|---|--|
| Cho biet ten cua ban: Minh Chao ban Minh | Chạy chương trình và quan sát kết quả. |
|---|--|

Giải thích chương trình

cgreeting là biến con trỏ được khởi tạo bằng phát biểu char *cgreeting = "Chao ban" thay vì char cgreeting[] = "Chao ban". Cả hai cách đều cho cùng kết quả và đều dành số byte cho chuỗi kèm theo kí tự null. Đối với mảng địa chỉ của kí tự đầu tiên của mảng sẽ là tên mảng, còn con trỏ sẽ có thêm biến con trỏ trỏ đến tên cgreeting.



`char *cgreeting = "Chao ban"`
(biến con trỏ)

`char cgreeting[] = "Chao ban"`
(hằng con trỏ)

9.2.7 Khởi tạo mảng con trỏ trỏ đến chuỗi

Ví dụ 6:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Chuong trinh nhap thang (so) và in ra thang (chu) tuong ung*/ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | char *cthang[12]= {"January", "February", "March", "April", |
| 9 | "May", "June", "July", "August", "September", |
| 10 | "October", "November", "December"}; |
| 11 | int ithang; |
| 12 | printf("Nhap vao thang (1-12): "); |
| 13 | scanf("%d", &ithang); |
| 14 | printf("%s.\n", cthang[ithang-1]); |
| 15 | getch(); |
| 16 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

| | |
|--------------------------------------|--|
| Nhap vao thang (1-12): 2 February | Chạy lại chương trình vào thử nhập vào các tháng khác Quan sát kết quả. |
|--------------------------------------|--|

Giải thích chương trình

Khai báo char *cthang[12] có ý nghĩa như sau: cthang là tên gọi, dấu * là kiểu con trỏ trỏ đến kí tự (char).

| | Địa chỉ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|------------|---------|---|---|---|----|----|----|----|----|----|----|
| cthang[0] | → 1010 | J | a | n | u | a | r | y | \0 | | |
| cthang[1] | → 1020 | F | e | b | r | u | a | r | y | \0 | |
| cthang[2] | → 1030 | M | a | r | c | h | \0 | | | | |
| cthang[3] | → 1040 | A | p | r | i | l | \0 | | | | |
| cthang[4] | → 1050 | M | a | y | \0 | | | | | | |
| cthang[5] | → 1060 | J | u | n | e | \0 | | | | | |
| cthang[6] | → 1070 | J | u | l | y | \0 | | | | | |
| cthang[7] | → 1080 | A | u | g | u | s | t | \0 | | | |
| cthang[8] | → 1090 | S | e | p | t | e | m | b | e | r | \0 |
| cthang[9] | → 1100 | O | c | t | o | b | e | r | \0 | | |
| cthang[10] | → 1110 | N | o | v | e | m | b | e | r | \0 | |
| cthang[11] | → 1120 | D | e | c | e | m | b | e | r | \0 | |

Mảng các chuỗi char cthang[12][10]

| | | | | | | | | | | | | |
|------------|------|--------|---|---|---|----|----|----|----|----|----|----|
| cthang[0] | 1010 | → 1010 | J | a | n | u | a | r | y | \0 | | |
| cthang[1] | 1018 | → 1018 | F | e | b | r | u | a | r | y | \0 | |
| cthang[2] | 1027 | → 1027 | M | a | r | c | h | \0 | | | | |
| cthang[3] | 1033 | → 1033 | A | p | r | i | l | \0 | | | | |
| cthang[4] | 1039 | → 1039 | M | a | y | \0 | | | | | | |
| cthang[5] | 1043 | → 1043 | J | u | n | e | \0 | | | | | |
| cthang[6] | 1048 | → 1048 | J | u | l | y | \0 | | | | | |
| cthang[7] | 1053 | → 1053 | A | u | g | u | s | t | \0 | | | |
| cthang[8] | 1060 | → 1060 | S | e | p | t | e | m | b | e | r | \0 |
| cthang[9] | 1070 | → 1070 | O | c | t | o | b | e | r | \0 | | |
| cthang[10] | 1078 | → 1078 | N | o | v | e | m | b | e | r | \0 | |
| cthang[11] | 1087 | → 1087 | D | e | c | e | m | b | e | r | \0 | |

Mảng các con trỏ trỏ đến các chuỗi char *cthang[12]

Khởi tạo mảng các con trỏ trỏ đến các chuỗi chiếm ít bộ nhớ hơn khởi tạo mảng chuỗi.

9.2.8 Xử lý con trỏ trỏ đến chuỗi

Ví dụ 7:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Nhập danh sách tên và sắp xếp theo thu tu tăng dần*/ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #include <string.h> |
| 6 | |
| 7 | #define MAXNUM 5 |

```

8 #define MAXLEN 10
9
10 void main(void)
11 {
12     char cname[MAXNUM][MAXLEN]; //mang chuoai
13     char *cptr[MAXNUM]; //mang con tro tro den chuoai
14     char *ctemp;
15     int i, ij, icount = 0;
16
17     //nhap danh sach ten
18     while (icount < MAXNUM)
19     {
20         printf("Nhap vao ten nguoi thu %d: ", icount + 1);
21         gets(cname[icount]);
22         cptr[icount++] = cname[icount]; //con tro den ten
23     }
24
25     //sap xep danh sach theo thu tu tang dan
26     for (i = 0; i < icount - 1; i++)
27         for (ij = i + 1; ij < icount; ij++)
28             if (strcmp(cptr[i], cptr[ij]) > 0)
29             {
30                 ctemp = cptr[i];
31                 cptr[i] = cptr[ij];
32                 cptr[ij] = ctemp;
33             }
34
35     //In danh sach da sap xep
36     printf("Danh sach sau khi sap xep:\n");
37     for (i = 0; i < icount; i++)
38         printf("Ten nguoi thu %d : %s\n", i + 1, cptr[i]);
39     getch();
40 }

```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

Kết quả in ra màn hình

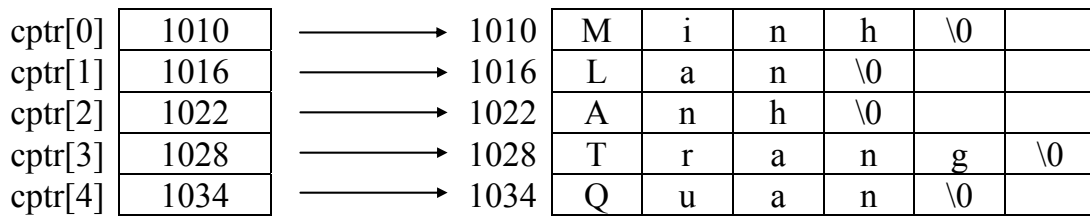
Nhap vao ten nguoi thu 1: Minh
 Nhap vao ten nguoi thu 2: Lan
 Nhap vao ten nguoi thu 3: Anh
 Nhap vao ten nguoi thu 4: Trang
 Nhap vao ten nguoi thu 5: Quan
 Danh sach sau khi sap xep:
 Ten nguoi thu 1: Anh
 Ten nguoi thu 2: Lan
 Ten nguoi thu 3: Minh
 Ten nguoi thu 4: Quan
 Ten nguoi thu 5: Trang

Chạy lại chương trình và thử nhập với dữ liệu khác.

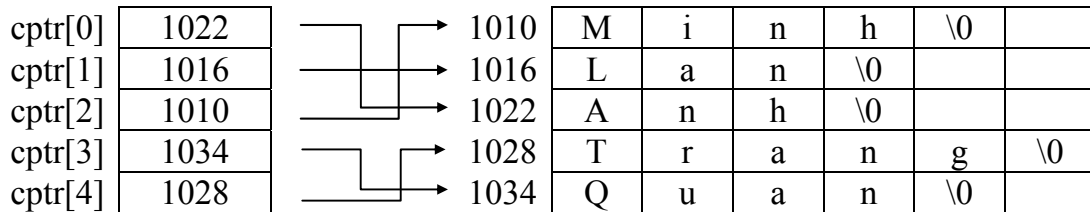
Quan sát kết quả.

Giải thích chương trình

Trong chương trình dùng cả mảng chuỗi char `cname[MAXNUM][MAXLEN]` và mảng con trỏ trỏ đến chuỗi char `*cptr[MAXNUM]`;



↑ Mảng các con trỏ trỏ đến chuỗi trước khi sắp xếp




↑ Mảng các con trỏ trỏ đến chuỗi sau khi sắp xếp

9.2.9 Con trỏ trỏ đến con trỏ


Ví dụ 8:

| Dòng | File | Edit | Search | Run | Compile | Debug | Project | Option | Window | Help |
|------|---|------|--------|-----|---------|-------|---------|--------|--------|------|
| 1 | /* In ma trận*/ | | | | | | | | | |
| 2 | | | | | | | | | | |
| 3 | #include <stdio.h> | | | | | | | | | |
| 4 | #include <conio.h> | | | | | | | | | |
| 5 | | | | | | | | | | |
| 6 | #define ROWS 4 | | | | | | | | | |
| 7 | #define COLS 5 | | | | | | | | | |
| 8 | | | | | | | | | | |
| 9 | void main(void) | | | | | | | | | |
| 10 | { | | | | | | | | | |
| 11 | int itable[ROWS][COLS] = {{10, 12, 14, 16, 18}, | | | | | | | | | |
| 12 | {11, 13, 15, 17, 19}, | | | | | | | | | |
| 13 | {20, 22, 24, 26, 28}, | | | | | | | | | |
| 14 | {21, 23, 25, 27, 29}}; | | | | | | | | | |
| 15 | int i, ij, ix = 10; | | | | | | | | | |
| 16 | | | | | | | | | | |
| 17 | for (i = 0; i < ROWS; i ++) | | | | | | | | | |
| 18 | for (ij = 0; ij < COLS; ij ++) | | | | | | | | | |
| 19 | *(*(table + i) + ij) += ix; | | | | | | | | | |
| 20 | | | | | | | | | | |
| 21 | for (i = 0; i < ROWS; i ++) | | | | | | | | | |
| 22 | { | | | | | | | | | |
| 23 | for (ij = 0; ij < COLS; ij ++) | | | | | | | | | |
| 24 | printf("%4d", *(*(table + i) + ij)); | | | | | | | | | |
| 25 | printf("\n"); | | | | | | | | | |
| 26 | } | | | | | | | | | |
| 27 | getch(); | | | | | | | | | |
| 28 | } | | | | | | | | | |

| | | | | | |
|---------|-----------------|-----------------|------------------|---------|----------|
| F1 Help | Alt-F8 Next Msg | Alt-F7 Prev Msg | Alt - F9 Compile | F9 Make | F10 Menu |
|---------|-----------------|-----------------|------------------|---------|----------|

 **Kết quả in ra màn hình**

| | |
|--|--|
| 20 22 24 26 28 21 23 25 27 29 30 32 34 36 38 31 33 35 37 39 | Chạy chương trình và quan sát kết quả. |
|--|--|

 **Giải thích chương trình**

Trong chương trình dùng cả mảng chuỗi char cname[MAXNUM][MAXLEN] và mảng con trỏ trỏ đến chuỗi char *cptr[MAXNUM];.

9.3 Bài tập

Làm lại các bài tập ở bài Mảng và chuỗi sử dụng biến con trỏ.



Bài 10 :

CÁC KIỂU DỮ LIỆU TỰ TẠO

10.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Ý nghĩa, cách khai structure, enum
- Nhập, xuất structure.
- Khởi tạo structure, enum
- Một số kỹ thuật thao tác trên structure, enum
- Dùng struct tham số cho hàm.

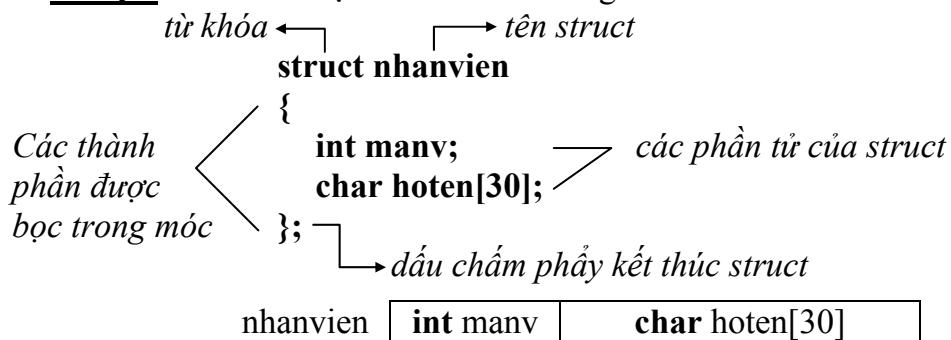
10.2 Nội dung

10.2.1 Structure

Đối với mảng, chỉ có thể lưu nhiều thông tin có cùng kiểu dữ liệu. Nhưng với structure ta có thể lưu thông tin như một mảng có nhiều kiểu dữ liệu khác nhau.

10.2.1.1 Khai báo kiểu structure

Ví dụ 1: khai báo một structure về thông tin nhân viên



Ví dụ trên định nghĩa kiểu dữ liệu mới có tên là **struct nhanvien**. Mỗi biến kiểu này gồm 2 phân tử: biến nguyên có tên là **manv** và biến chuỗi có tên **hoten**.

☞ **struct** phải viết bằng chữ thường

10.2.1.2 Cách khai báo biến có kiểu structure

Ví dụ 2: `struct nhanvien nv;` hoặc `nhanvien nv;`

Khai báo biến `nv` có kiểu `struct nhanvien`

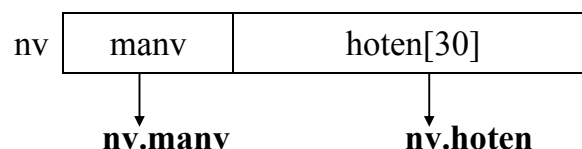
☞ vừa tạo structure `nhanvien` vừa khai báo biến `nv`

```

struct nhanvien
{
    int manv;
    char hoten[30];
} nv;

```

10.2.1.3 Tham chiếu các phân tử trong structure



Để tham chiếu đến `manv` trong `nv` ta viết như sau: **nv.manv** (là biến có kiểu `int`)

☞ Đối với biến khai báo kiểu con trỏ **nhanvien *nv** thì tham chiếu đến phần tử manv: **nv -> manv.**

Ví dụ 3: Nhập và in danh sách nhân viên.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|--|
| 1 | /* Danh sach nhan vien */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #include <stdlib.h> |
| 6 | |
| 7 | #define MAX 50 |
| 8 | |
| 9 | void main(void) |
| 10 | { |
| 11 | struct nhanvien |
| 12 | { |
| 13 | int manv; |
| 14 | char hoten[30]; |
| 15 | }; |
| 16 | nhanvien snv[MAX]; |
| 17 | char ctam[10]; |
| 18 | int i, in; |
| 19 | printf("Nhap vao so nhan vien: "); |
| 20 | gets(ctam); |
| 21 | in = atoi(ctam); |
| 22 | |
| 23 | //Nhap danh sach nhan vien |
| 24 | for(i = 0; i < in; i++) |
| 25 | { |
| 26 | printf("Nhap vao ma nhan vien thu %d: ", i + 1); |
| 27 | gets(ctam); |
| 28 | snv[i].manv = atoi(ctam); |
| 29 | printf("Nhap vao ho ten: "); |
| 30 | gets(snv[i].hoten); |
| 31 | } |
| 32 | |
| 33 | //in danh sach nhan vien |
| 34 | for(i = 0; i < in; i++) |
| 35 | printf("%5d %s\n", snv[i].manv, snv[i].hoten); |
| 36 | getch(); |
| 37 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

☞ **Kết quả in ra màn hình**

| | |
|--|---|
| Nhap vao so nhan vien: 2 Nhap vao ma nhan vien thu 1: 123 Nhap vao ho ten: Le Thuy Doan Trang Nhap vao ma nhan vien thu 2: 35 | Chạy và thử lại chương trình với dữ liệu khác. Quan sát kết quả. |
|--|---|

| | |
|---|--|
| Nhap vao ho ten: Le Nguyen Tuan Anh 123 Le Thuy Doan Trang 35 Le Nguyen Tuan Anh – | |
|---|--|

☞ Trong chương trình trên dùng tổ hợp 2 dòng 20 và 21 gồm 2 lệnh gets, atoi để nhập một số nguyên tránh lỗi do scanf và vùng đệm bàn phím gây ra.

10.2.1.4 Khởi tạo structure

Ví dụ 4: Nhập vào bảng số xe, cho biết xe đó đăng kí ở tỉnh nào.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Xac dinh bien so xe */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #include <stdlib.h> |
| 6 | |
| 7 | #define MAX 6 |
| 8 | |
| 9 | void main(void) |
| 10 | { |
| 11 | struct tinh |
| 12 | { |
| 13 | int ma; |
| 14 | char *ten; |
| 15 | }; |
| 16 | tinh sds[MAX] = {{60, "Dong Nai"}, {61, "Binh Duong"}, {62, "Long An"}, |
| 17 | {63, "Tien Giang"}, {64, "Vinh Long"}, {65, "Can Tho"}}; |
| 18 | char ctam[10]; |
| 19 | int i, in; |
| 20 | printf("Nhap vao bien so xe: "); |
| 21 | gets(ctam); |
| 22 | in = atoi(ctam); |
| 23 | |
| 24 | for(i = 0; i < MAX; i++) |
| 25 | if (sds[i].ma == in) |
| 26 | printf("Xe dang ki o tinh %s.\n", sds[i].ten); |
| 27 | getch(); |
| 28 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

☞ Kết quả in ra màn hình

| | |
|--|--|
| Nhap vao bien so xe: 62F5-1152 Xe dang ki o tinh Long An – | Chạy và thử lại chương trình với 65H5-1246, 60F4-7712, 64F1-4542 Quan sát kết quả. |
|--|--|

☞ Dòng 22 đổi chuỗi sang số nguyên, ở ví dụ trên sau khi dòng này thực hiện giá trị của in = 62.


10.2.1.5 Structure lồng nhau**Ví dụ 5:** Nhập và in danh sách nhân viên.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Danh sach nhan vien */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #include <stdlib.h> |
| 6 | |
| 7 | #define MAX 50 |
| 8 | |
| 9 | void main(void) |
| 10 | { |
| 11 | struct giacanh |
| 12 | { |
| 13 | char vo_chong[30]; |
| 14 | char con; |
| 15 | }; |
| 16 | |
| 17 | struct nhanvien |
| 18 | { |
| 19 | int manv; |
| 20 | char hoten[30]; |
| 21 | giacanh canhan; |
| 22 | }; |
| 23 | nhanvien snv[MAX]; |
| 24 | char ctam[10]; |
| 25 | int i, in; |
| 26 | printf("Nhap vao so nhan vien: "); |
| 27 | gets(ctam); |
| 28 | in = atoi(ctam); |
| 29 | |
| 30 | //Nhap danh sach nhan vien |
| 31 | for(i = 0; i < in; i++) |
| 32 | { |
| 33 | printf("Nhap vao ma nhan vien thu %d: ", i + 1); |
| 34 | gets(ctam); |
| 35 | snv[i].manv = atoi(ctam); |
| 36 | printf("Nhap vao ho ten: "); |
| 37 | gets(snv[i].hoten); |
| 38 | printf("Cho biet ten vo (hoac chong): "); |
| 39 | gets(snv[i].canhan.vo_chong); |
| 40 | printf("So con: "); |
| 41 | gests(ctam); |
| 42 | } |
| 43 | |

| | |
|---|--|
| 44 | //in danh sach nhan vien |
| 45 | for(i = 0; i < in; i++) |
| 46 | { |
| 47 | printf("Ma so: %d\nHo ten: %s\n Ho ten vo (hoac chong): %s\nSo con: %d", |
| 48 | snv[i].manv, snv[i].hoten, snv[i].canhan.vo_chong, snv[i].canhan.con); |
| 49 | getch(); |
| 50 | } |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | |

Kết quả in ra màn hình

| | |
|---|---|
| Nhap vao so nhan vien: 3 Nhap vao ma nhan vien thu 1: 123 Nhap vao ho ten: Le Thuy Doan Trang Nhap vao ma nhan vien thu 2: 35 Nhap vao ho ten: Le Nguyen Tuan Anh 123 Le Thuy Doan Trang 35 Le Nguyen Tuan Anh — | Chạy và thử lại chương trình với dữ liệu khác. Quan sát kết quả. |
|---|---|

 Trong chương trình trên dùng tổ hợp 2 dòng 20 và 21 gồm 2 lệnh gets, atoi để nhập một số nguyên tránh lỗi do scanf và vùng đệm bàn phím gây ra.

10.2.1.6 Truyền structure sang hàm

Giống như mảng, bạn có thể truyền vào hàm qua tham biến.

Ví dụ 6: Sửa lại ví dụ 3, sử dụng hàm cho nhập và in danh sách

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Danh sach nhan vien */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #include <stdlib.h> |
| 6 | |
| 7 | #define MAX 50 |
| 8 | |
| 9 | //Khai bao structure toan cuc |
| 10 | struct nhanvien |
| 11 | { |
| 12 | int manv; |
| 13 | char hoten[30]; |
| 14 | }; |
| 15 | |
| 16 | //Khai bao prototype |
| 17 | void input(nhanvien, int); |
| 18 | void output(nhanvien, int); |
| 19 | |
| 20 | //Ham nhap danh sach |
| 21 | void input(nhanvien snv[], int in) |

```

22 {
23     char ctam[10];
24     for(int i = 0; i < in; i++)
25     {
26         printf("Nhap vao ma nhan vien thu %d: ", i + 1);
27         gets(ctam);
28         snv[i].manv = atoi(ctam);
29         printf("Nhap vao ho ten: ");
30         gets(snv[i].hoten);
31     }
32 }
33
34 //Ham in danh sach ra man hinh
35 void output(nhanvien snv[], int in)
36 {
37     for(i = 0; i < in; i++)
38         printf("%5d %s\n", snv[i].manv, snv[i].hoten);
39 }
40
41 void main(void)
42 {
43     nhanvien snv[MAX];
44     char ctam[10];
45     int i, in;
46     printf("Nhap vao so nhan vien: ");
47     gets(ctam);
48     in = atoi(ctam);
49     input(snv, in);
50     output(snv, in);
51     getch();
52 }

```

F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

Kết quả in ra màn hình

```

Nhap vao so nhan vien: 3
Nhap vao ma nhan vien thu 1: 123
Nhap vao ho ten: Le Thuy Doan Trang
Nhap vao ma nhan vien thu 2: 35
Nhap vao ho ten: Le Nguyen Tuan Anh
123 Le Thuy Doan Trang
35 Le Nguyen Tuan Anh
_

```

Chạy và thử lại chương trình với dữ liệu khác.
Quan sát kết quả.

Giải thích chương trình

Ở chương trình này ta phải khai báo struct nhanvien là biến toàn cục, vì khi định nghĩa hàm input và output có sử dụng kiểu dữ liệu struct nhanvien.

Bạn lưu ý rằng khi truyền struct sang hàm, không tạo bản sao mới. Vì vậy struct truyền sang hàm có dạng tham biến. Nghĩa là giá trị của các phần tử trong struct sẽ bị ảnh hưởng nếu có sự thay đổi trên chúng.

Ví dụ 7: Sửa lại ví dụ 6, từ dòng 20 đến dòng 32 như sau:

```
//Ham nhap tung nhan vien
nhanvien newnv()
{
    nhanvien snv;
    printf("Ma nhan vien: ");
    gets(ctam);
    snv.manv = atoi(ctam);
    printf("Ho ten: ");
    gets(snv.hoten);
    return (snv);
}

//Ham nhap danh sach nhan vien
void input(nhanvien snv[], int in)
{
    for(int i = 0; i < in; i++)
    {
        printf("Nhap vao nhan vien thu %d: ", i + 1);
        snv[i] = newnv();
    }
}
```

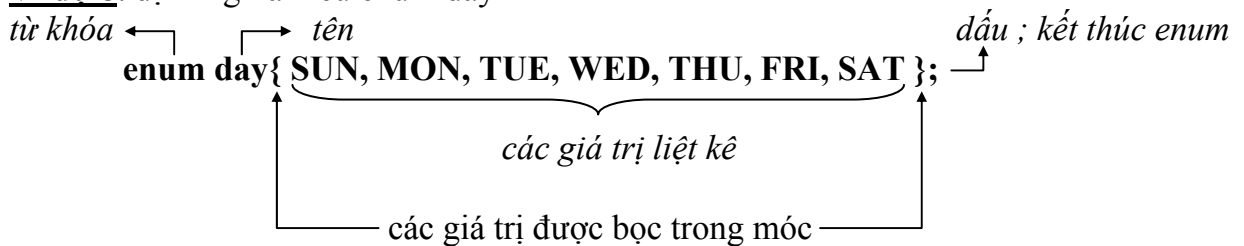
Hàm newnv có kiểu trả về là struct nhanvien

10.2.2 Enum

Một biến là kiểu dữ liệu enum có thể nhận được một giá trị nào đó trong các giá trị được liệt kê.

10.2.2.1 Định nghĩa kiểu enum

Ví dụ 8: định nghĩa kiểu enum day



⇒ Các tên thứ (SUN, MON ... SAT) trong day sẽ được đánh số lần lượt từ 0 đến 6 (SUN là 1, MON là 2... SAT là 6). Nếu bạn muốn bắt đầu bằng giá trị khác thì gán giá trị mong muốn vào và trị kế tiếp sẽ tăng lên 1.

enum phải viết bằng chữ thường

10.2.2.2 Cách khai báo biến có kiểu enum

Ví dụ 9: enum day ngay; hoặc day ngay;

Khai báo biến ngay có kiểu enum day.

☞ vừa tạo enum day vừa khai báo biến ngay

```
enum day { SUN, MON, TUE, WED, THU, FRI, SAT } ngay;
```

10.2.2.3 Sử dụng enum trong chương trình

Ví dụ 10: Tính tiền lương tuần cho nhân viên. Thứ bảy và Chủ nhật được tính phụ trội

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Tinh tien luong tuan cho nhan vien */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | #define PHU_TROI_T7 1.5 |
| 7 | #define PHU_TROI_CN 2.0 |
| 8 | |
| 9 | //đinh nghĩa enum |
| 10 | enum tuan {CHU_NHAT, THU_HAI, THU_BA, THU_TU, THU_NAM, THU_SAU, THU_BAY}; |
| 11 | typedef enum tuan ngay_tuan; //đinh nghĩa ngay_tuan là tuan |
| 12 | |
| 13 | void main(void) |
| 14 | { |
| 15 | int igio; |
| 16 | float fLuongCB, fLuongNgay, fTongLuong; |
| 17 | char cngay[][4] = {"Chu Nhat", "Thu Hai", "Thu Ba", "Thu Tu", "Thu Nam", "Thu |
| 18 | Sau", "Thu Bay"}; |
| 19 | ngay_tuan ngay; |
| 20 | ngay_tuan ngay_mai(ngay_tuan); //khai bao prototype |
| 21 | |
| 22 | printf("Nhap vao luong can ban: "); |
| 23 | scanf("%f", &fLuongCB); |
| 24 | |
| 25 | luong = 0.0; |
| 26 | printf("Nhap vao so gio lam viec tu Thu hai den Chu nhat:\n"); |
| 27 | engay = CHU_NHAT; |
| 28 | do |
| 29 | { |
| 30 | engay = ngay_mai(engay); |
| 31 | printf("Nhap vao gio lam viec ngay %s :", cngay[engay]); |
| 32 | scanf("%d", &igio); |
| 33 | swith(engay) |
| 34 | { |
| 35 | case THU_HAI: case THU_BA: case THU_TU: case THU_NAM: case THU_SAU: |
| 36 | fLuongNgay = fLuongCB; |
| 37 | break; |
| 38 | case THU_BAY: |
| 39 | fLuongNgay = fLuongCB * PHU_TROI_T7; |

| | |
|---|--|
| <pre> 40 break; 41 case CHU_NHAT: 42 fLuongNgay = fLuongCB * PHU_TROI_CN; 43 break; 44 } 45 fTongLuong += fLuongNgay * igio; 46 } while (ngay != CHU_NHAT); 47 printf("Tong luong tuan = %8.2f dong.\n", fTongLuong); 48 getch(); 49 } 50 51 //ham chon ngay ke tiep 52 ngay_tuan ngay_mai(ngay_tuan en) 53 { 54 ngay_tuan engay_ke; 55 switch(en) 56 { 57 case CHU_NHAT : engay_ke = THU_HAI; break; 58 case THU_HAI : engay_ke = THU_BA; break; 59 case THU_BA : engay_ke = THU_TU; break; 60 case THU_TU : engay_ke = THU_NAM; break; 61 case THU_NAM : engay_ke = THU_SAU; break; 62 case THU_SAU : engay_ke = THU_BAY; break; 63 case THU_BAY : engay_ke = CHU_NHAT; break; 64 } 65 return (engay_ke); 66 } </pre> | <p>F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu</p> |
|---|--|

Kết quả in ra màn hình

| | |
|--|---|
| <pre> Nhap vao luong can ban: 250 Nhap vao so gio lam viec tu Thu hai den Chu nhat: Nhap vao gio lam viec ngay Thu Hai: 7 Nhap vao gio lam viec ngay Thu Ba: 8 Nhap vao gio lam viec ngay Thu Tu: 6 Nhap vao gio lam viec ngay Thu Nam: 7 Nhap vao gio lam viec ngay Thu Sau: 8 Nhap vao gio lam viec ngay Thu Bay: 7 Nhap vao gio lam viec ngay Chu Nhat: 6 Tong luong tuan = 14625.00 dong. _ </pre> | <p>Hàm chọn ngày kế tiếp trên khá dài, bạn thay từ dòng 54 đến 65 bằng câu lệnh</p> <p>return (++en > 6 ? 0 : en); hoặc</p> <p>return (++en % 7);</p> <p>Chạy lại chương trình, quan sát, nhận xét và đánh giá kết quả với dữ liệu khác.</p> |
|--|---|

Giải thích chương trình

Ở chương trình này ta phải khai báo struct nhanvien là biến toàn cục, vì khi định nghĩa hàm input và output có sử dụng kiểu dữ liệu struct nhanvien.

10.3 Bài tập

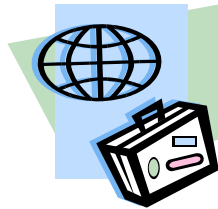
1. Định nghĩa 1 dãy cấu trúc có thể được dùng làm danh bạ điện thoại, gồm có tên, địa chỉ, số điện thoại, với số mẫu tin tối đa là 40. Viết chương trình với các chức năng sau: nhập thông mới, tìm kiếm số điện thoại, in danh sách theo quận.

2. Viết chương trình đọc vào tên, địa chỉ, sắp xếp tên và địa chỉ theo thứ tự alphabet, sau đó hiển thị danh sách đã được sắp xếp.

3. Viết chương trình nhận vào các thông tin sau: Tên đội bóng, số trận thắng, số trận hòa, số trận thua. In ra đội bóng có số điểm cao nhất (với 1 trận thắng = 3 điểm, 1 trận hòa = 1 điểm và 1 trận thua = 0 điểm).

4. Xây dựng cấu trúc gồm: Họ tên, ngày sinh, trường, số báo danh, điểm thi. Trong đó, điểm thi là cấu trúc gồm 3 môn: Toán, Lý, Hóa. Nhập liệu vào khoảng 10 thí sinh, tìm và in ra các thí sinh có tổng điểm 3 môn ≥ 15 .

5. Viết chương trình tạo lập và tìm kiếm dữ liệu. Nội dung yêu cầu gồm: Nhập họ và tên, địa chỉ (gồm: Quận, phường, tổ), tuổi, lương. Tìm kiếm những người ở Quận 3 có tuổi dưới 30 thu nhập từ 500.000đ trở lên và in ra màn hình.



Bài 11 :**TẬP TIN****11.1 Mục tiêu**

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Ý nghĩa của việc sử dụng tập tin (file)
- Mở, đóng file
- Ghi, đọc file số nguyên, mảng, chuỗi.
- Một số hàm xử lý tập tin.


11.2 Nội dung**11.2.1 Ví dụ ghi, đọc số nguyên**

| Dòng | File | Edit | Search | Run | Compile | Debug | Project | Option | Window | Help |
|------|--|------|--------|-----|---------|-------|---------|--------|--------|------|
| 1 | /* Ghi n so nguyen vao file va doc ra tu file*/ | | | | | | | | | |
| 2 | | | | | | | | | | |
| 3 | #include <stdio.h> | | | | | | | | | |
| 4 | #include <conio.h> | | | | | | | | | |
| 5 | #include <stdlib.h> | | | | | | | | | |
| 6 | | | | | | | | | | |
| 7 | void main(void) | | | | | | | | | |
| 8 | { | | | | | | | | | |
| 9 | FILE *f; | | | | | | | | | |
| 10 | int in, i; | | | | | | | | | |
| 11 | printf("Nhap vao so n: "); | | | | | | | | | |
| 12 | scanf("%d", &in); | | | | | | | | | |
| 13 | | | | | | | | | | |
| 14 | //Ghi file | | | | | | | | | |
| 15 | if((f = fopen("int_data.dat", "wb")) == NULL) //mo file | | | | | | | | | |
| 16 | { | | | | | | | | | |
| 17 | printf("Khong the mo file!\n"); | | | | | | | | | |
| 18 | exit(0); | | | | | | | | | |
| 19 | } | | | | | | | | | |
| 20 | else | | | | | | | | | |
| 21 | for(i = 1; i <= in; i++) | | | | | | | | | |
| 22 | fwrite(&i, sizeof(int), 1, f); //ghi file | | | | | | | | | |
| 23 | fclose(f); //dong file | | | | | | | | | |
| 24 | | | | | | | | | | |
| 25 | //Doc file | | | | | | | | | |
| 26 | f = fopen("int_data.dat", "rb"); | | | | | | | | | |
| 27 | while(fread(&i, sizeof(int), 1, f) == 1) | | | | | | | | | |
| 28 | printf("%d ", i); | | | | | | | | | |
| 29 | fclose(f); | | | | | | | | | |
| 30 | getch(); | | | | | | | | | |
| 31 | } | | | | | | | | | |

| | | | | | |
|---------|-----------------|-----------------|------------------|---------|----------|
| F1 Help | Alt-F8 Next Msg | Alt-F7 Prev Msg | Alt - F9 Compile | F9 Make | F10 Menu |
|---------|-----------------|-----------------|------------------|---------|----------|

 **Kết quả in ra màn hình**

| | |
|--|---|
| Nhập vào số n: 10 1 2 3 4 5 6 7 8 9 10 _ | Chạy và thử lại chương trình với dữ liệu khác. Quan sát kết quả. |
|--|---|

 **Giải thích chương trình**

- Dòng 9 : `FILE *f;` : khai báo biến con trỏ f có kiểu cấu trúc FILE.
- Dòng 15 : `if(f = fopen("int_data.dat", "wb") == NULL)` : là câu lệnh mở tập tin có tên int_data.dat ở mode "w" (ghi) dạng "b" (nhị phân), sau khi lệnh này thực hiện xong trả về dạng con trỏ FILE và gán cho f, nếu kết quả trả về = NULL thì không thể mở được tập tin, tập tin mở ở mode "w" nếu trên đĩa đã có sẵn tập tin này thì nội dung của nó sẽ bị ghi đè, nếu chưa có thì tập tin sẽ được tạo mới.
- Dòng 22 : `fwrite(&i, sizeof(int), 1, f);` : ghi thông tin vào tập tin, thông tin được ghi vào mỗi lần là một số nguyên i. Hàm này có 4 đối số: địa chỉ để ghi cấu trúc, kích thước của cấu trúc và số cấu trúc sẽ ghi, sau cùng là con trỏ để trỏ tới tập tin.
- Dòng 23 : `fclose(f);` : đóng tập tin
- Dòng 26 : `f = fopen("int_data.dat", "rb");` : mở tập tin có tên int_data.dat ở mode "r" (đọc) dạng "b" (nhị phân). Tập tin phải có sẵn trên đĩa.
- Dòng 27 : `while(fread(&i, sizeof(int), 1, f) == 1)` : đọc thông tin từ tập tin, mỗi lần đọc một số nguyên và lưu vào biến i. Mỗi lần đọc thành công giá trị trả về sẽ là số cấu trúc thực sự được đọc, nếu giá trị trả về = 0 báo hiệu kết thúc file.

 **Từ khóa FILE phải viết bằng chữ in hoa. Sử dụng fopen, fwrite, fread, fclose phải khai báo #include <stdio.h>, NULL phải viết hoa.**

11.2.2 Ghi, đọc mảng

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Ghi n số nguyên vào file và đọc ra từ file*/ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | #include <stdlib.h> |
| 6 | |
| 7 | #define MAX 5 |
| 8 | |
| 9 | void main(void) |
| 10 | { |
| 11 | FILE *f; |
| 12 | int i, ia[MAX], ib[MAX]; |
| 13 | for (i = 0; i < 10; i++) |
| 14 | { |
| 15 | printf("Nhập vào một số: "); |
| 16 | scanf("%d", &ia[i]); |
| 17 | } |
| 18 | |
| 19 | if((f = fopen("array.dat", "wb")) == NULL) |

| | | |
|---|---------------------------------|---------------------|
| 20 | { | |
| 21 | printf("Khong the mo file!\n"); | |
| 22 | exit(0); | |
| 23 | } | |
| 24 | fwrite(ia, sizeof(ia), 1, f); | //ghi mang vao file |
| 25 | fclose(f); | |
| 26 | | |
| 27 | f = fopen("array.dat", "rb"); | |
| 28 | fread(ib, sizeof(ib), 1, f); | //doc mang tu file |
| 29 | for (i = 0; i < 10; i++) | |
| 30 | printf("%d ", ib[i]); | |
| 31 | fclose(f); | |
| 32 | getch(); | |
| 33 | } | |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu | | |

Kết quả in ra màn hình

| | |
|--|---|
| Nhap vao mot so: 3 Nhap vao mot so: 6 Nhap vao mot so: 8 Nhap vao mot so: 1 Nhap vao mot so: 9 3 6 8 1 9 — | Chạy và thử lại chương trình với dữ liệu khác. Quan sát kết quả. |
|--|---|

11.2.3 Ghi, đọc structure

| Dòng | File | Edit | Search | Run | Compile | Debug | Project | Option | Window | Help |
|------|------------------------------------|------|--------|-----|---------|-------|---------|--------|--------|------|
| 1 | /* Danh sach nhan vien */ | | | | | | | | | |
| 2 | | | | | | | | | | |
| 3 | #include <stdio.h> | | | | | | | | | |
| 4 | #include <conio.h> | | | | | | | | | |
| 5 | #include <stdlib.h> | | | | | | | | | |
| 6 | | | | | | | | | | |
| 7 | #define MAX 50 | | | | | | | | | |
| 8 | | | | | | | | | | |
| 9 | void main(void) | | | | | | | | | |
| 10 | { | | | | | | | | | |
| 11 | FILE *f; | | | | | | | | | |
| 12 | struct nhanvien | | | | | | | | | |
| 13 | { | | | | | | | | | |
| 14 | int manv; | | | | | | | | | |
| 15 | char hoten[30]; | | | | | | | | | |
| 16 | }; | | | | | | | | | |
| 17 | nhanvien snv[MAX], snv1[MAX]; | | | | | | | | | |
| 18 | char ctam[10]; | | | | | | | | | |
| 19 | int i, in; | | | | | | | | | |
| 20 | printf("Nhap vao so nhan vien: "); | | | | | | | | | |
| 21 | gets(ctam); | | | | | | | | | |
| 22 | in = atoi(ctam); | | | | | | | | | |
| 23 | | | | | | | | | | |

| | |
|---|---|
| <pre> 24 //Nhap danh sach nhan vien va ghi vao file 25 if((f = fopen("struct.dat", "wb")) == NULL) 26 { 27 printf("Khong the mo file!\n"); 28 exit(0); 29 } 30 fwrite(&in, sizeof(int), 1, f); //ghi so nhan vien vao file 31 for(i = 0; i < in; i++) 32 { 33 printf("Nhap vao ma nhan vien thu %d: ", i + 1); 34 gets(ctam); 35 snv[i].manv = atoi(ctam); 36 printf("Nhap vao ho ten: "); 37 gets(snv[i].hoten); 38 fwrite(&snv[i], sizeof(nhanvien), 1, f); //ghi tung nhan vien vao file 39 } 40 fclose(f); 41 42 //doc danh sach nhan vien tu file va in ra 43 f = fopen("struct.dat", "rb"); 44 fread(&in, sizeof(int), 1, f); //doc so nhan vien 45 for(i = 0; i < in; i++) 46 { 47 fread(&snv1[i], sizeof(nhanvien), 1, f); //doc tung nhan vien in ra man hinh 48 printf("%5d %s\n", snv[i].manv, snv[i].hoten); 49 } 50 getch(); 51 } </pre> | <p>F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu</p> |
|---|---|

Kết quả in ra màn hình

| | |
|---|---|
| <pre> Nhap vao so nhan vien: 2 Nhap vao ma nhan vien thu 1: 123 Nhap vao ho ten: Le Thuy Doan Trang Nhap vao ma nhan vien thu 2: 35 Nhap vao ho ten: Le Nguyen Tuan Anh 123 Le Thuy Doan Trang 35 Le Nguyen Tuan Anh _ </pre> | <p>Chạy và thử lại chương trình với dữ liệu khác. Quan sát kết quả.</p> |
|---|---|

11.2.4 Các mode khác để mở tập tin

Ở 3 ví dụ trên chỉ sử dụng 2 mode "w" (ghi) và "r" (đọc), sau đây là một số mode khác:

- "a": mở để nối thêm, thông tin sẽ được ghi thêm vào cuối của tập tin đã có hoặc tạo tập tin mới nếu chưa có trên đĩa.
- "r+": mở để vừa đọc vừa ghi, tập tin phải có sẵn trên đĩa.
- "w+": mở để vừa đọc vừa ghi, nội dung của tập tin đã có trên đĩa sẽ bị ghi đè lên.
- "a+": mở để đọc và nối thêm, nếu trên đĩa chưa có tập tin nó sẽ được tạo mới.

11.2.5 Một số hàm thao tác trên file khác

Xem bài **Các hàm chuẩn**

11.3 Bài tập

Thêm chức năng ghi, đọc file ở các bài tập của bài Mạng và chuỗi, Các dữ liệu tự tạo.

Bài 12 :

ĐỆ QUY

12.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kỹ năng cơ bản sau:

- Ý nghĩa, phương pháp hoạt động của đệ quy.
- Có thể thay vòng lặp bằng đệ quy.

12.2 Nội dung

Bất cứ một hàm nào đó có thể triệu gọi hàm khác, nhưng ở đây một hàm nào đó có thể triệu gọi chính mình. Kiểu hàm như thế được gọi là **hàm đệ quy**.

Phương pháp đệ quy thường dùng phổ biến trong những ứng dụng mà cách giải quyết có thể được thể hiện bằng việc áp dụng liên tiếp cùng giải pháp cho những tập hợp con của bài toán.

Ví dụ 1: tính $n!$

$n! = 1*2*3*...*(n-2)*(n-1)*n$ với $n \geq 1$ và $0! = 1$.

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Ham tinh giai thua */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | int in; |
| 9 | long giai thua(int); |
| 10 | printf("Nhap vao so n: "); |
| 11 | scanf("%d", &in); |
| 12 | printf("%d! = %ld.\n", in, giai thua(in)); |
| 13 | getch(); |
| 14 | } |
| 15 | |
| 16 | long giai thua(int in) |
| 17 | { |
| 18 | int i; |
| 19 | long ltich = 1; |
| 20 | if (in == 0) |
| 21 | return (1L); |
| 22 | else |
| 23 | { |
| 24 | for (i = 1; i <= in; i++) |
| 25 | ltich *= i; |
| 26 | return (ltich); |
| 27 | } |
| 28 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

👉 Kết quả in ra màn hình

| | |
|------------------------------------|--|
| Nhập vào số n: 5 5! = 120. — | Thử lại chương trình với số liệu khác. |
|------------------------------------|--|

Với $n! = 1 * 2 * 3 * \dots * (n-2) * (n-1) * n$,
 ta viết lại như sau: $(1 * 2 * 3 * \dots * (n-2) * (n-1)) * n = n * (n-1)! \dots = n * (n-1) * (n-2)! \dots$

👉 Ta viết lại hàm giai thừa bằng đệ quy như sau:

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Ham tinh giai thua */ |
| 2 | |
| 3 | long giai thua(int in) |
| 4 | { |
| 5 | int i; |
| 6 | if (in == 0) |
| 7 | return (1L); |
| 8 | else |
| 9 | return (in * giai thua(in - 1)); |
| 10 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

👉 Chạy lại chương trình, quan sát, nhận xét và đánh giá kết quả

👉 Giải thích hoạt động của hàm đệ quy giai thừa

Ví dụ giá trị truyền vào hàm giai thừa qua biến in = 5.

- Thứ tự gọi thực hiện hàm giai thừa

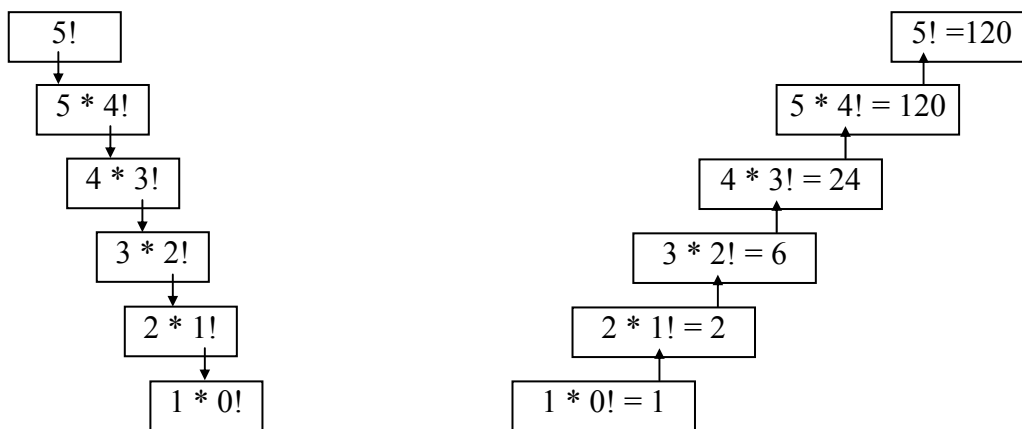
| giai thua(in) | return(in * giai thua(in - 1)) |
|---------------|--------------------------------|
| 5 | 5 * giai thua(4) = 5 * ? |
| 4 | 4 * giai thua(3) = 4 * ? |
| 3 | 3 * giai thua(2) = 3 * ? |
| 2 | 2 * giai thua(1) = 2 * ? |
| 1 | 1 * giai thua(0) = 1 * ? |

Khi tham số in = 0 thì return về giá trị 1L (giá trị 1 kiểu long). Lúc này các giá trị ? bắt đầu định trị theo thứ tự ngược lại.

- Định trị theo thứ tự ngược lại

| giai thua(in) | return(in * giai thua(in - 1)) |
|---------------|---------------------------------|
| 1 | 1 * giai thua(0) = 1 * 1 = 1 |
| 2 | 2 * giai thua(1) = 2 * 1 = 2 |
| 3 | 3 * giai thua(2) = 3 * 2 = 6 |
| 4 | 4 * giai thua(3) = 4 * 6 = 24 |
| 5 | 5 * giai thua(4) = 5 * 24 = 120 |

Kết quả sau cùng ta có $5! = 120$.



Thứ tự gọi đệ quy

Định trị theo thứ tự ngược lại

Ví dụ 2: Dãy số Fibonacci

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ... Bắt đầu bằng 0 và 1, các số tiếp theo bằng tổng hai số đi trước.

Dãy Fibonacci được khai báo đệ quy như sau:

$$\begin{cases} \text{Fibonacci}(0) = 0 \\ \text{Fibonacci}(1) = 1 \\ \text{Fibonacci}(n) = \text{Fibonacci}(n - 1) + \text{Fibonacci}(n - 2) \end{cases}$$

| Dòng | File Edit Search Run Compile Debug Project Option Window Help |
|------|---|
| 1 | /* Tinh so fibonacci thu n */ |
| 2 | |
| 3 | #include <stdio.h> |
| 4 | #include <conio.h> |
| 5 | |
| 6 | void main(void) |
| 7 | { |
| 8 | long in; |
| 9 | long fibonacci(long); |
| 10 | printf("Nhap vao so n: "); |
| 11 | scanf("%ld", &in); |
| 12 | printf("Fibonacci(%ld) = %ld.\n", in, fibonacci(in)); |
| 13 | getch(); |
| 14 | } |
| 15 | |
| 16 | long fibonacci(long in) |
| 17 | { |
| 18 | if (in == 0 in == 1) |
| 19 | return in; |
| 20 | else |
| 21 | return fibonacci(in - 1) + fibonacci(in - 2); |
| 22 | } |
| | F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Kết quả in ra màn hình

| | |
|--|--|
| Nhap vao so n: 10 Fibonacci(10) = 55. | Thử lại chương trình với số liệu khác. |
|--|--|

| | |
|---|--|
| – | |
|---|--|

? Sử dụng đệ quy hay vòng lặp

Phương pháp đệ quy không phải bao giờ cũng là giải pháp hữu hiệu nhất. Giải pháp vòng lặp có hiệu quả về mặt thời gian và vùng nhớ. Còn với đệ quy mỗi lần gọi đệ quy máy phải dành một số vùng nhớ để trữ các trị, thông số và biến cục bộ. Do đó, đệ quy tốn nhiều vùng nhớ, thời gian truyền đối mục, thiết lập vùng nhớ trung gian và trả về kết quả... Nhưng sử dụng phương pháp đệ quy trông chương trình đẹp mắt hơn vòng lặp và tính thuyết phục của nó. Điều cốt lõi khi thiết đặt chương trình phải làm thế nào hàm đệ quy có thể chấm dứt thông qua điều kiện cơ bản.

12.3 Bài tập

1. Viết hàm đệ quy tính tổng n số nguyên dương đầu tiên:

$$\text{tong}(n) = n + \text{tong}(n - 1)$$



Bài 13 :**TRÌNH SOẠN THẢO CỦA BORLAND C**

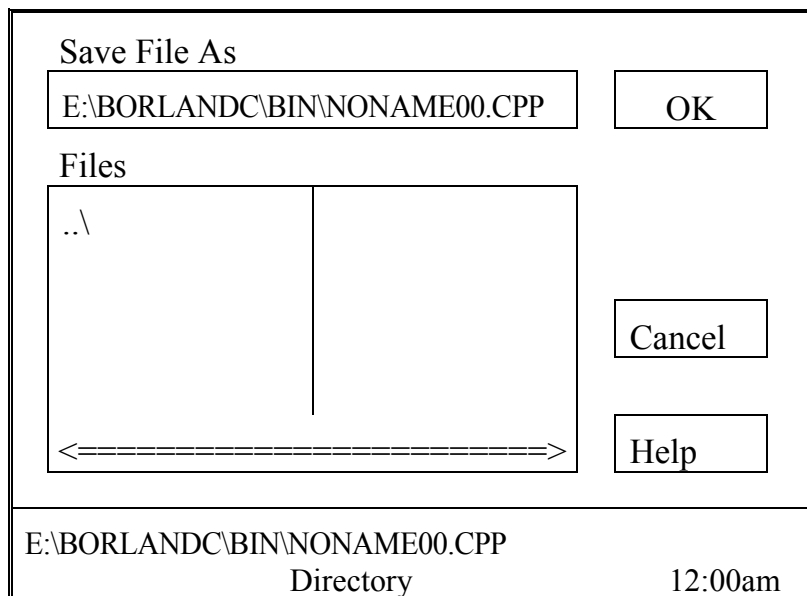
BC có hệ thống menu nhiều cấp. Để chọn một mục trong Menu bạn ấn phím **F10** (kích hoạt Menu), dùng các phím mũi tên di chuyển vệt sáng đến mục muốn chọn ấn **Enter** hoặc ấn phím có kí tự đổi màu (*phím chọn nhanh màu đỏ*). Có thể chọn nhanh mục menu trên thanh menu chính bạn ấn tổ hợp phím **Alt + phím có kí tự màu đỏ**. Ví dụ: ấn tổ hợp phím **Alt + F** kích hoạt menu File.

13.1 Mở tập tin soạn thảo mới

Chọn menu File -> chọn mục New -> tạo file soạn thảo mới có tên mặc định là NONAME00.CPP, NONAME01.CPP... tùy theo số lần mục New được chọn.

13.2 Lưu tập tin**13.2.1 Nếu là tập tin soạn thảo mới chưa lưu**

Ấn phím F2 hoặc chọn menu File -> Save hoặc chọn menu File -> Save As sẽ xuất hiện hộp thoại Save File As



- + Chọn đường dẫn cần lưu tập tin ở hộp Files, chọn ..\ để trở về thư mục cha thư mục hiện tại.
- + Đặt tên tập tin ở hộp Save File As
- + Chọn OK
- + Hoặc có thể gõ [ổ đĩa:][đường dẫn]<tên tập tin>, chọn OK.

☞ Dùng phím TAB để chuyển đổi vệt sáng giữa các mục trong hộp thoại.

Ví dụ: muốn lưu tập tin có tên BT_IF1.CPP vào thư mục D:\BAITAPC

- + Bạn gõ vào hộp Save File As D: -> Enter (OK), danh sách các thư mục, tập tin của D hiển thị ở hộp Files, chọn thư mục BAITAPC ở hộp Files, gõ tên BT_IF1.CPP vào hộp Save File As, chọn OK.
- + Hoặc nếu bạn nhớ rõ đường dẫn, gõ vào hộp Save File As D:\BAITAPC\BT_IF1, chọn OK.

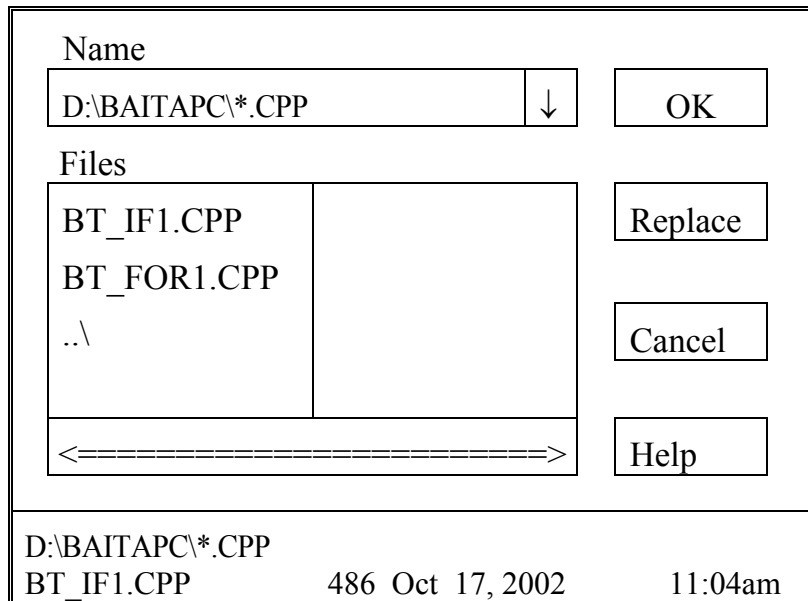
13.2.2 Nếu là tập tin đã lưu ít nhất 1 lần hoặc được mở bằng lệnh Open:

- + Ấn F2 hoặc chọn menu File -> Save, nội dung tập tin hiện hành sẽ được cập nhật nếu có thay đổi.
- + Chọn menu File -> Save As, xuất hiện hộp thoại Save File As, thực hiện các bước như mục 2.1 (nghĩa là bạn muốn lưu nội dung tập tin hiện hành với đường dẫn, tên tập tin khác).

☞ Để biết tập tin đang soạn thảo đã lưu hay chưa, bạn xem ở góc dưới trái cửa sổ, nếu có dấu hoa thị là văn bản của bạn có thay đổi và chưa được lưu.

13.3 Mở tập tin

Mở một tập tin đã có trên đĩa. Ấn phím F3 hoặc chọn menu File -> Open, hộp thoại Open a File xuất hiện:



- + Chọn đường dẫn cần mở tập tin ở hộp Files, chọn ..\ để trở về thư mục cha thư mục hiện tại.
- + Chọn tập tin cần mở ở hộp Files.
- + Chọn OK
- + Hoặc có thể gõ [ô đĩa:][đường dẫn]<tên tập tin>, chọn OK.

Ví dụ: muốn mở tập tin có tên BT_IF1.CPP chứa trong thư mục D:\BAITAPC

- + Bạn gõ vào hộp Name D: -> Enter (OK), danh sách các thư mục, tập tin của D hiển thị ở hộp Files, chọn thư mục BAITAPC ở hộp Files, chọn tập tin BT_IF1.CPP, chọn OK.
- + Hoặc bạn có thể gõ vào hộp Name D:\BAITAPC*.CPP để hiển thị danh sách các tập tin có phần mở rộng CPP ở hộp Files, chọn tập tin BT_IF1.CPP, chọn OK.
- + Hoặc nếu bạn nhớ rõ đường dẫn, gõ vào hộp Name D:\BAITAPC\BT_IF1.CPP, chọn OK.

13.4 Các phím, tổ hợp phím thường dùng

13.4.1 Các phím di chuyển con trỏ

| Phím / Tổ hợp phím | Chức năng |
|--------------------|---------------------------------------|
| ← | Di chuyển con trỏ sang trái một ký tự |
| → | Di chuyển con trỏ sang phải một ký tự |
| ↑ | Di chuyển con trỏ lên trên một dòng |
| ↓ | Di chuyển con trỏ xuống dưới một dòng |
| Home | Di chuyển con trỏ về đầu dòng |
| End | Di chuyển con trỏ về cuối dòng |
| PgUp (Page Up) | Lật lùi lại một trang màn hình |
| PgDn (Page Down) | Lật tới một trang màn hình |
| Ctrl – PgUp | Di chuyển con trỏ về đầu tập tin |
| Ctrl – PgDn | Di chuyển con trỏ về cuối tập tin |

| | |
|---------------|---|
| Backspace (←) | Xóa một ký tự bên trái con trỏ |
| Del (Delete) | Xóa một ký tự tại vị trí con trỏ |
| Ins (Insert) | Chuyển đổi giữa chế độ ghi chèn và ghi đè |
| Enter | Xuống một dòng |

13.4.2 Các phím thao tác trên khối

| Phím / Tổ hợp phím | Chức năng |
|---------------------------|--|
| Shift – → | Đánh dấu chọn một ký tự bên phải |
| Shift – ← | Đánh dấu chọn một ký tự bên trái |
| Shift – ↑ | Đánh dấu chọn một hàng trên vị trí con trỏ |
| Shift – ↓ | Đánh dấu chọn một hàng tại vị trí con trỏ |
| Shift – Home | Đánh dấu chọn từ đầu hàng đến vị trí con trỏ |
| Shift – End | Đánh dấu chọn từ vị trí con trỏ đến cuối hàng |
| Shift – PgUp | Đánh dấu chọn một trang lui màn hình |
| Shift – PgDn | Đánh dấu chọn một trang tới màn hình |
| Ctrl – Shift – ← | Đánh dấu chọn một từ bên trái |
| Ctrl – Shift – → | Đánh dấu chọn một từ bên phải |
| Ctrl – Shift – End | Đánh dấu chọn từ vị trí con trỏ đến cuối tập tin |
| Ctrl – Shift – Home | Đánh dấu chọn từ vị trí con trỏ đến đầu tập tin |

13.4.3 Các thao tác xóa

| Phím | Chức năng |
|---------------|-------------------------------------|
| Backspace (←) | Xóa một ký tự bên trái con trỏ |
| Del (Delete) | Xóa một ký tự tại vị trí con trỏ |
| Ctrl – Y | Xóa dòng tại vị trí con trỏ |
| Ctrl – K – Y | Xóa khối |
| Ctrl – Q – Y | Xóa từ vị trí con trỏ đến cuối dòng |
| Ctrl – T | Xóa một từ tại vị trí con trỏ |
| Insert | Bật / tắt chế độ viết chèn / đè |

13.4.4 Các thao tác copy, di chuyển

| Phím / Tổ hợp phím | Chức năng |
|---------------------------|---|
| Ctrl – Insert | Sao chép khối chọn vào Clipboard |
| Shift – Delete | Cắt khối chọn vào Clipboard |
| Ctrl – Delete | Xóa khối chọn |
| Shift – Insert | Dán thông tin từ Clipboard vào vị trí con trỏ |
| Ctrl – K – R | Đọc thông tin từ tập tin vào cửa sổ soạn thảo |
| Ctrl – K – W | Ghi thông tin từ cửa sổ soạn thảo vào tập tin |

13.4.5 Các thao tác khác

| Phím / Tổ hợp phím | Chức năng |
|---------------------------|--|
| F3 | Tạo tập tin mới hoặc nạp tập tin từ đĩa vào cửa sổ |

| | |
|-------------------------------|--|
| | soạn thảo |
| Alt – F3 | Đóng tập tin tại cửa sổ hiện hành |
| F2 | Lưu tập tin hiện hành |
| F6 | Chuyển đổi qua lại giữa các cửa sổ đang soạn thảo |
| F5 | Chuyển đổi cửa sổ soạn thảo maximize ↔ restore |
| Alt – Backspace | Phục hồi lại thao tác trước đó |
| Ctrl – K – H | Ẩn / hiện dấu khối |
| Ctrl – Q – F | Tìm kiếm |
| Ctrl – L | Lập lại lần tìm kiếm sau cùng |
| Ctrl – Q – A | Tìm kiếm và thay thế |
| Ctrl – Q – [, Ctrl – Q –] | Xác định cặp ngoặc bao 1 khối lệnh |
| F1 | Gọi giúp đỡ |
| Shift – F1 | Hiện cửa sổ giúp đỡ theo mục |
| Ctrl – F1 | Hiện cửa sổ giúp đỡ về hàm, toán tử... tương ứng tại vị trí con trỏ. |

13.5 Ghi một khối ra đĩa

Đánh dấu chọn khối bằng các phím thao tác trên khối. Ấn tổ hợp phím Ctrl - K - W, xuất hiện hộp thoại Write Block to File, thực hiện các bước như lưu tập tin.

13.6 Chèn nội dung file từ đĩa vào vị trí con trỏ

Di chuyển con trỏ đến vị trí cần chèn nội dung, Ấn tổ hợp phím Ctrl - K - R, xuất hiện hộp thoại Read Block from File, thực hiện các bước như mở tập tin.

13.7 Tìm kiếm văn bản trong nội dung soạn thảo

Ấn tổ hợp phím Ctrl - Q - F hoặc chọn menu Search -> Find, hộp thoại Find Text xuất hiện:

- + Gõ nội dung cần tìm vào hộp Text to Find.
- + Nếu cần đánh dấu / bỏ chọn các mục sau:
 - Case-sensitive: phân biệt chữ hoa chữ thường.
 - Whole words only: tìm văn bản đứng riêng một từ.
 - Forward: Tìm xuôi.
 - Backward: Tìm ngược.
- + Chọn OK.

Khi tìm xong, muốn tìm tiếp ấn tổ hợp phím Ctrl - L hoặc chọn menu Search -> Search again.

13.8 Tìm và thay thế văn bản trong nội dung soạn thảo

Ấn tổ hợp phím Ctrl - Q - A hoặc chọn menu Search -> Replace, hộp thoại Find Text xuất hiện:

- + Gõ nội dung cần thay thế vào hộp Text to Find.
- + Gõ nội dung mới vào hộp New Text.
- + Nếu cần đánh dấu /bỏ chọn các mục sau:
 - Case-sensitive: phân biệt chữ hoa chữ thường.
 - Whole words only: tìm văn bản đứng riêng một từ.
 - Forward: Tìm xuôi.
 - Backward: Tìm ngược.
- + Chọn OK để thay thế từng văn bản được tìm thấy, chọn Change All để thay thế tất cả.

13.9 Sửa lỗi cú pháp

Khi biên dịch chương trình, nếu thành công bạn sẽ nhận được thông báo từ cửa sổ Compile (dòng cuối): **Success: Press any key**, ngược lại là thông báo lỗi **Error: Press any key**.

Nếu là thông báo lỗi, khi ấn phím bất kỳ cửa sổ Message xuất hiện chứa danh sách các lỗi. Thông báo lỗi đầu tiên được làm sáng và dòng có lỗi trong chương trình cũng được làm sáng. Kèm theo dấu đồ cho biết trình biên dịch phát hiện vị trí lỗi. Dùng phím mũi tên để di chuyển đến các thông báo lỗi khác, bạn sẽ thấy vệt sáng trong chương trình cũng sẽ chuyển đến dòng chứa lỗi tương ứng. Nếu bạn Enter tại dòng thông báo lỗi nào thì con trỏ sẽ chuyển vào cửa sổ soạn thảo tại dòng chứa lỗi tương ứng.

Ví dụ: In ra "Hello".

| |
|--|
| File Edit Search Run Compile Debug Project Option Window Help |
| <pre>#include <stdio.h> #include <conio.h> void main(void) { printf("Hello"; getch(); }</pre> |
| Message |
| <p>Compiling HELLO.CPP Error HELLO.CPP 5: Function call missing) Error HELLO.CPP 6: Function 'gech' should have a prototype</p> |
| F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu |

Vệt sáng nằm ở thông báo lỗi đầu tiên và dòng chứa lỗi tương ứng trong chương trình cũng được làm sáng: Lỗi ở dòng 5, không đóng ngoặc hàm printf.

13.10 Chạy từng bước

13.10.1 Mỗi lần 1 bước

Ở mỗi bước thực hiện ta phải bấm phím F7. Với lần bấm F7 đầu tiên, dòng đầu tiên trong chương trình (dòng main()) sẽ được làm sáng, dòng được làm sáng này được gọi là dòng chuẩn bị thực hiện, và nó sẽ được thực hiện ở lần bấm phím F7 tiếp theo. Mỗi lần bấm phím F7 dòng đang được làm sáng sẽ được thực hiện, sau đó trở về màu bình thường, và tùy theo nội dung của dòng đó mà một dòng lệnh tiếp theo nào đó sẽ được làm sáng để chuẩn bị thực hiện ở bước tiếp theo.

Ta cũng có thể dùng phím F8 thay cho F7 với những dòng không có lời gọi hàm được khai báo trong chương trình. Sự khác nhau giữa F7 và F8 chỉ xảy ra khi dòng được làm sáng có lời gọi hàm được khai báo trong chương trình.

Như vậy nhờ chạy từng bước, ta có thể dễ dàng nắm được các lỗi logic trong chương trình.

13.10.2 Tái lập lại quá trình gỡ rối

Bấm Ctrl-F2 hoặc vào menu Run chọn Program reset. Khi đó bộ nhớ dùng cho việc gỡ rối sẽ được giải tỏa, không có dòng nào được làm sáng và kết thúc quá trình gỡ rối.

13.10.3 Dừng cửa sổ Watch

Lần từng bước thường được dùng kèm với việc sử dụng cửa sổ Watch để theo dõi giá trị của biến trong mỗi bước thực hiện để dễ tìm ra nguyên nhân chương trình thực hiện sai.

Để làm điều đó ta phải nhập vào các biến cần theo dõi, bằng cách chọn mục Add watch của menu Break/Watch hoặc có thể bấm Ctrl-F7, sau đó nhập tên biến vào tại vị trí con trỏ trong cửa sổ Add watch và bấm Enter. Để nhập thêm tên biến vào cửa sổ này phải lập lại chức năng này hoặc bấm phím Insert.

Trong soạn thảo nếu chưa nhìn thấy cửa sổ Watch, ta bấm phím F5, khi đó trên màn hình sẽ đồng thời hiện ra cả 2 cửa sổ, để chuyển đổi giữa 2 cửa sổ này bấm phím F6. Mỗi biến trên cửa sổ Watch thực hiện trên 1 dòng. Khi cửa sổ Watch được chọn sẽ có 1 dòng được làm sáng để chỉ rằng biến đó đang được chọn. Giá trị trong cửa sổ Watch sẽ thay đổi theo kết quả của từng bước thực hiện.

13.11 Sử dụng Help (Giúp đỡ)

- Ấn phím F1 để kích hoạt màn hình Help chính.
- Muốn xem Help của hàm trong soạn thảo, di chuyển con trỏ đến vị trí hàm đó ấn tổ hợp phím Ctrl - F1
- Ấn tổ hợp phím Shift - F1 để xem danh sách các mục Help
- Ấn tổ hợp phím Alt - F1 để quay về màn hình Help trước đó.



Bài 14 :

CÁC HỆ ĐẾM

14.1 Khái niệm

Các chữ số cơ bản của một hệ đếm là các chữ số dùng để biểu diễn mọi số trong hệ đếm ấy. Hệ đếm thường gặp nhất là hệ thập phân (hệ 16). Nhưng do bản chất nhị phân của các thiết bị điện tử cho nên hầu hết dạng biểu diễn dữ liệu và các phép đại số đều thực hiện bằng hệ nhị phân (hệ 2). Hệ bát phân (hệ 8) rất ít dùng và hệ thập phân (hệ 10) là hệ chúng ta đang sử dụng để biểu diễn một con số nào đó trong cuộc sống hằng ngày.

- Ví dụ 1:
- Hệ nhị phân gồm 2 chữ số : 0, 1
 - Hệ bát phân gồm 8 chữ số : 0, 1, 2, 3, 4, 5, 6, 7
 - Hệ thập phân gồm 10 chữ số : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
 - Hệ thập lục phân gồm 16 chữ số : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

■ Các chữ số trong một hệ đếm được sắp xếp theo quy tắc: Bất kỳ cơ số N nguyên dương nào, có N ký hiệu khác nhau để biểu diễn các số trong hệ thống. Giá trị của N ký hiệu này được sắp xếp từ 0 đến N-1.

- Ví dụ 2:
- Hệ nhị phân có cơ số N = 2 : các chữ số được đánh từ 0..1
 - Hệ bát phân có cơ số N = 8 : các chữ số được đánh từ 0..7
 - Hệ thập phân có cơ số N = 10 : các chữ số được đánh từ 0..9
 - Hệ thập lục phân có cơ số N = 16 : các chữ số được đánh từ 0..9, A..F

■ Do hệ thập lục phân có 16 chữ số, mà trong hệ thống chữ viết chỉ biểu diễn được 9 chữ số, vì vậy người ta chọn các ký tự A..F để biểu diễn 10..15 và nó cũng được xem như 1 chữ số (A, B...F) thay vì phải viết 10, 11...15 (2 chữ số)

14.2 Quy tắc

Để biểu diễn một số của một hệ đếm, ta dùng chỉ số đặt ở góc dưới phải số đó.

- 01101₂ : biểu thị số nhị phân.
- 082₈ : biểu thị số bát phân.
- 23₁₆ : biểu thị số thập lục phân.

Đối với hệ thập phân ta có thể ghi chỉ số hoặc không ghi (nhằm hiểu), vì số thập phân là số mà ta sử dụng quen thuộc hằng ngày. Do đó, ta sử dụng công thức sau để chuyển đổi từ các hệ đếm sang hệ thập phân (cơ số 10) :

$$X = a_n a_{n-1} \dots a_1 a_0 = a_n b^n + a_{n-1} b^{n-1} + \dots + a_1 b^1 + a_0 b^0 \quad (*)$$

trong đó,

- **b** : là cơ số hệ đếm.
- **a₀...a_n** : là các chữ số trong một hệ đếm.
- **X** : là số thuộc một hệ đếm cơ số b.

■ Bảng các giá trị tương đương ở hệ thập phân, nhị phân, bát phân, thập lục phân. (**)

| Thập phân | Nhị phân | Bát phân | Thập lục phân |
|-----------|----------|----------|---------------|
| 0 | 0000 | 0 | 0 |
| 1 | 0001 | 1 | 1 |
| 2 | 0010 | 2 | 2 |
| 3 | 0011 | 3 | 3 |
| 4 | 0100 | 4 | 4 |
| 5 | 0101 | 5 | 5 |

| | | | |
|----|------|----|---|
| 6 | 0110 | 6 | 6 |
| 7 | 0111 | 7 | 7 |
| 8 | 1000 | 10 | 8 |
| 9 | 1001 | 11 | 9 |
| 10 | 1010 | 12 | A |
| 11 | 1011 | 13 | B |
| 12 | 1100 | 14 | C |
| 13 | 1101 | 15 | D |
| 14 | 1110 | 16 | E |
| 15 | 1111 | 17 | F |

14.3 Chuyển đổi giữa các hệ

14.3.1 Chuyển đổi giữa hệ 2 và hệ 10

- Chuyển đổi từ hệ 2 sang hệ 10

Ví dụ 3: $X = 01011_2$, để chuyển sang hệ 10 ta dùng công thức (*)

$$X = 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$$

$$= 0 + 8 + 0 + 2 + 1$$

$$= 11$$

Ví dụ 4: $X = 1011010_2$, để chuyển sang hệ 10 ta dùng công thức (*)

$$X = 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0$$

$$= 64 + 0 + 16 + 8 + 0 + 2 + 0$$

$$= 90$$

- Chuyển đổi từ hệ 10 sang hệ 2

Ví dụ 5: $X = 11$

| | | | | | | | | | | |
|----|---|---|----|------|------|---|---|-----|-----|---|
| 11 | 2 | | 11 | chia | 2 | = | 5 | dur | 1 | |
| 1 | 5 | 2 | 5 | chia | 2 | = | 2 | dur | 1 | |
| | 1 | 2 | 2 | 2 | chia | 2 | = | 1 | dur | 0 |
| | | 0 | 1 | 1 | chia | 2 | = | 0 | dur | 1 |
| | | | 1 | 0 | | | | | | |

→ **1011**₂ →
 kết quả hệ nhị phân
 ← **1011**₂ ←

Ví dụ 6: $X = 90$

| | | | | | | | | | | |
|----|----|----|----|------|------|---|---|-----|-----|---|
| 90 | 2 | | 90 | chia | 2 | = | 2 | dur | 0 | |
| 0 | 45 | 2 | 45 | chia | 2 | = | 1 | dur | 1 | |
| | 1 | 22 | 2 | 22 | chia | 2 | = | 0 | dur | 0 |
| | | 0 | 11 | 11 | chia | 2 | = | 5 | dur | 1 |
| | | | 1 | 5 | chia | 2 | = | 2 | dur | 1 |
| | | | 1 | 2 | chia | 2 | = | 1 | dur | 0 |
| | | | 0 | 1 | chia | 2 | = | 1 | dur | 0 |
| | | | 1 | 0 | | | | | | |

→ **1011010**₂ →
 kết quả hệ nhị phân
 ← **1011010**₂ ←

14.3.2 Chuyển đổi giữa hệ 8 và hệ 10

■ Chuyển đổi từ hệ 8 sang hệ 10

Ví dụ 7: $X = 2106_8$, để chuyển sang hệ 10 ta dùng công thức (*)

$$\begin{aligned}
 X &= 2 \cdot 8^3 + 1 \cdot 8^2 + 0 \cdot 8^1 + 6 \cdot 8^0 \\
 &= 1024 + 64 + 0 + 6 \\
 &= 1094
 \end{aligned}$$

Ví dụ 8: $X = 130_8$, để chuyển sang hệ 10 ta dùng công thức (*)

$$\begin{aligned}
 X &= 1 \cdot 8^2 + 3 \cdot 8^1 + 0 \cdot 8^0 \\
 &= 64 + 24 + 0 \\
 &= 88
 \end{aligned}$$

■ Chuyển đổi từ hệ 10 sang hệ 8

Ví dụ 9: $X = 1094$

| | |
|--|---|
| $ \begin{array}{r l} 1094 & 8 \\ \hline 6 & 136 \\ \hline & 0 \quad \quad 8 \\ & \quad \quad 17 \\ & \quad \quad 1 \quad \quad 8 \\ & \quad \quad 2 \quad \quad 8 \\ & \quad \quad 2 \quad \quad 0 \end{array} $ | $ \begin{array}{l} 1094 \text{ chia } 8 = 136 \text{ dư } 6 \\ \hline 136 \text{ chia } 8 = 17 \text{ dư } 0 \\ \hline 17 \text{ chia } 8 = 2 \text{ dư } 1 \\ \hline 2 \text{ chia } 8 = 0 \text{ dư } 2 \end{array} $ |
| 2106_8 | 2106_8 |

kết quả hệ bát phân

Ví dụ 10: $X = 88$

| | |
|--|---|
| $ \begin{array}{r l} 88 & 8 \\ \hline 0 & 11 \\ \hline & 3 \quad \quad 8 \\ & \quad \quad 1 \\ & \quad \quad 1 \quad \quad 8 \\ & \quad \quad 1 \quad \quad 0 \end{array} $ | $ \begin{array}{l} 88 \text{ chia } 8 = 11 \text{ dư } 0 \\ \hline 11 \text{ chia } 8 = 1 \text{ dư } 3 \\ \hline 1 \text{ chia } 8 = 0 \text{ dư } 1 \end{array} $ |
| 130_8 | 130_8 |

kết quả hệ bát phân

14.3.3 Chuyển đổi giữa hệ 16 và hệ 10

■ Chuyển đổi từ hệ 16 sang hệ 10

Ví dụ 11: $X = F40_{16}$, để chuyển sang hệ 10 ta dùng công thức (*)

$$\begin{aligned}
 X &= 15 \cdot 16^2 + 4 \cdot 16^1 + 0 \cdot 16^0 \\
 &= 3840 + 64 + 0 \\
 &= 3904
 \end{aligned}$$

Ví dụ 12: $X = 1D_{16}$, để chuyển sang hệ 10 ta dùng công thức (*)

$$X = 1 \cdot 16^1 + 13 \cdot 16^0$$

$$= 16 + 13$$

$$= 29$$

■ Chuyển đổi từ hệ 10 sang hệ 16

Ví dụ 13: X = 3904

| | | |
|------|-----|----|
| 3904 | 16 | |
| 0 | 244 | 16 |
| | 4 | 15 |
| | | 0 |

| | | | | | |
|------|------|------|-----|-----|----|
| 3904 | chia | 16 = | 244 | dur | 0 |
| 244 | chia | 16 = | 15 | dur | 4 |
| 15 | chia | 16 = | 0 | dur | 15 |

Số 15 tương ứng trong hệ 16 là F (xem bảng (**))

\rightarrow **F40**₁₆ \rightarrow kết quả hệ thập lục phân \leftarrow **F40**₁₆

Ví dụ 14: X = 29

| | | |
|----|----|----|
| 29 | 16 | |
| 13 | 1 | 16 |
| | 1 | 0 |

| | | | | | |
|----|------|------|---|-----|----|
| 29 | chia | 16 = | 1 | dur | 13 |
| 1 | chia | 16 = | 0 | dur | 1 |

Số 13 tương ứng trong hệ 16 là D (xem bảng (**))

\rightarrow **1D**₁₆ \rightarrow kết quả hệ thập lục phân \leftarrow **1D**₁₆

14.3.4 Chuyển đổi giữa hệ 2 và hệ 16

■ Chuyển đổi từ hệ 2 sang hệ 16

Ví dụ 15: X = 01011₂, để chuyển sang hệ 16 ta tra trong bảng (**)

\rightarrow X = B₁₆

Diễn giải : $\begin{matrix} 0 & 1011_2 \\ 0 & B_{16} \end{matrix} = B_{16}$

Ví dụ 16: X = 1011010₂, để chuyển sang hệ 16 ta tra trong bảng (**)

\rightarrow X = 5A₁₆

Diễn giải : $\begin{matrix} 101 & 1010_2 \\ 5 & A_{16} \end{matrix} = 5A_{16}$

■ Chuyển đổi từ hệ 16 sang hệ 2

Ví dụ 17: X = B₁₆, để chuyển sang hệ 2 ta tra trong bảng (**)

\rightarrow X = 1011₂

Diễn giải : $\begin{matrix} B_{16} \\ 1011_2 \end{matrix} = 1011_2$

Ví dụ 18: X = 5A₁₆, để chuyển sang hệ 2 ta tra trong bảng (**)

\rightarrow X = 1011010₂

Diễn giải : $\begin{matrix} 5 & A_{16} \\ 0101 & 1010_2 \end{matrix} = 1011010_2$

Bài 15 :

BIỂU THỨC VÀ PHÉP TOÁN

15.1 Biểu thức

Là sự phối hợp của những toán tử và toán hạng.

Ví dụ 1:

```
a + b
b = 1 + 5 * 2/i
a = 6 % (7 + 1)
x++ * 2/4 + 5 - power(i, 2)
```

Toán hạng sử dụng trong biểu thức có thể là hằng số, biến, hàm.

15.2 Phép toán

Trong C có 4 nhóm toán tử chính yếu sau đây:

15.2.1 Phép toán số học

- | | | |
|--|---|--|
| <ul style="list-style-type: none"> + : cộng - : trừ * : nhân / : chia % : lấy phần dư | } | áp dụng trên tất cả các toán hạng có kiểu dữ liệu char, int, float, double (kể cả long, short, unsigned) |
| |] | áp dụng trên các toán hạng có kiểu dữ liệu char, int, long |

* Thứ tự ưu tiên: Đảo dấu +, - () *, /, % +, -

Ví dụ 2:

```
10%4 = 2 (10 chia 4 dư 2);   9%3 = 0 (9 chia 3 dư 0)
3 * 5 + 4 = 19
6 + 2 / 2 - 3 = 4
-7 + 2 * ((4 + 3) * 4 + 8) = 65
```

☞ chỉ sử dụng cặp ngoặc () trong biểu thức, cặp ngoặc đơn được thực hiện theo thứ tự ưu tiên từ trong ra ngoài.

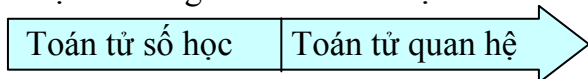
15.2.2 Phép quan hệ

- > : lớn hơn
- >= : lớn hơn hoặc bằng
- < : nhỏ hơn
- <= : nhỏ hơn hoặc bằng
- == : bằng
- != : khác

* Thứ tự ưu tiên: >, >=, <, <= ==, !=

☞ Kết quả của phép toán quan hệ là số nguyên kiểu int, bằng 1 nếu đúng, bằng 0 nếu sai. Phép toán quan hệ ngoài toán hạng được sử dụng là số còn được sử dụng với kiểu dữ liệu char.

* Thứ tự ưu tiên giữa toán tử số học và toán tử quan hệ



Ví dụ 3:

```
4 > 10          → có giá trị 0 (sai)
```


- 4 >= 4 → có giá trị 1 (đúng)
- 3 == 5 → có giá trị 0 (sai)
- 2 <= 1 → có giá trị 0 (sai)
- 6 != 4 → có giá trị 1 (đúng)
- 6 - 3 < 4 → có giá trị 1 (đúng), tương đương (6 - 3) < 4
- 2 * -4 < 3 + 2 → có giá trị 0 (sai), tương đương (-2 * -4) < (3 + 2)

15.2.3 Phép toán luận lý

- ! : NOT (phép phủ định)
- &&: AND (phép và)
- || : OR (phép hoặc)

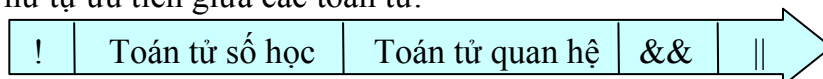
| Toán hạng a | Toán hạng b | !a | a && b | a b |
|-------------|-------------|----------|----------|----------|
| Khác 0 | Khác 0 | 0 (sai) | 1 (đúng) | 1 (đúng) |
| Khác 0 | Bằng 0 | 0 (sai) | 0 (sai) | 1 (đúng) |
| Bằng 0 | Khác 0 | 1 (đúng) | 0 (sai) | 1 (đúng) |
| Bằng 0 | Bằng 0 | 1 (đúng) | 0 (sai) | 0 (sai) |

* Thứ tự ưu tiên:

Ví dụ 4:

- !(2 <= 1) → có giá trị 1 (đúng)
- 5 && 10 → có giá trị 1 (đúng)
- !6 → có giá trị 0 (sai)
- 1 && 0 → có giá trị 0 (sai)
- 1 || 0 → có giá trị 1 (đúng)

* Thứ tự ưu tiên giữa các toán tử:



15.2.4 Phép toán trên bit (bitwise)

- & : và (AND)
- | : hoặc (OR)
- ^ : hoặc loại trừ (XOR)
- >> : dịch phải
- << : dịch trái
- ~ : đảo

| Bit a | Bit b | ~a | a & b | a b | a ^ b |
|-------|-------|----|-------|-------|-------|
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 |

Ví dụ 5:

- a = 13 → đổi ra hệ nhị phân → 1101
 - b = 10 → đổi ra hệ nhị phân → 1010
- | | | |
|--------|--------|----------------------|
| 1101 | 1101 | 1101 |
| & 1010 | 1010 | ^ 1010 |
| = 1000 | = 1111 | = 0111 |
| = 8 | = 15 | = 7 (dạng thập phân) |

$$\begin{array}{rcl}
 a = 1235 & \rightarrow \text{đổi ra hệ nhị phân} \rightarrow & 0100\ 1101\ 0011 \\
 b = 465 & \rightarrow \text{đổi ra hệ nhị phân} \rightarrow & 0001\ 1101\ 0001 \\
 \begin{array}{r}
 0100\ 1101\ 0011 \\
 \underline{\& 0001\ 1101\ 0001} \\
 = 0000\ 1101\ 0001 \\
 = 209
 \end{array} & & \begin{array}{r}
 0100\ 1101\ 0011 \\
 \underline{| 0001\ 1101\ 0001} \\
 = 0101\ 1101\ 0011 \\
 = 1491
 \end{array} & & \begin{array}{r}
 0100\ 1101\ 0011 \\
 \underline{\wedge 0001\ 1101\ 0001} \\
 = 0101\ 0000\ 0010 \\
 = 1282 \quad (\text{dạng thập phân})
 \end{array}
 \end{array}$$

15.2.5 Các phép toán khác

1. Phép toán gán

Phép gán là thay thế giá trị hiện tại của biến bằng một giá trị mới.

Các phép gán: =, +=, -=, *=, /=, %=, <<=, >>=, &=, |=, ^=.

Ví dụ 6: ta có giá trị i = 3

$$\begin{array}{ll}
 i = i + 3 & \rightarrow i = 6 \\
 i += 3 & \rightarrow i = 6 \equiv i = i + 3 \\
 i *= 3 & \rightarrow i = 9 \equiv i = i * 3
 \end{array}$$

2. Phép toán tăng, giảm: ++, --

Toán tử ++ sẽ cộng thêm 1 vào toán hạng của nó, toán tử -- sẽ trừ đi 1.

Ví dụ 7: ta có giá trị n = 6

+ Sau phép tính ++n hoặc n++, ta có n = 7.

+ Sau phép tính --n hoặc n--, ta có n = 5.

* Sự khác nhau giữa ++n và n++, --n và n--

+ Sau phép tính x = ++n + 2, ta có x = 9. (n tăng 1 cộng với 2 rồi gán cho x)

+ Sau phép tính x = n++ + 2, ta có x = 8. (n cộng với 2 gán cho x rồi mới tăng 1)

15.2.6 Độ ưu tiên của các phép toán

| Độ ưu tiên | Các phép toán | Trình tự kết hợp |
|------------|-----------------------------------|------------------|
| 1 | () [] -> | Trái sang phải |
| 2 | ! ~ & * - ++ -- (type) sizeof | Phải sang trái |
| 3 | * / % | Trái sang phải |
| 4 | + - | Trái sang phải |
| 5 | << >> | Trái sang phải |
| 6 | < <= > >= | Trái sang phải |
| 7 | == != | Trái sang phải |
| 8 | & | Trái sang phải |
| 9 | ^ | Trái sang phải |
| 10 | | Trái sang phải |
| 11 | && | Trái sang phải |
| 12 | | Trái sang phải |
| 13 | ? : | Phải sang trái |
| 14 | = += -= *= /= %= <<= >>= &= ^= = | Phải sang trái |
| 15 | , | Trái sang phải |

Lưu ý:

- Phép đảo (–) ở dòng 2, phép trừ (–) ở dòng 4
- Phép lấy địa chỉ (&) ở dòng 2, phép AND bit (&) ở dòng 8
- Phép lấy đối tượng con trỏ (*) ở dòng 2, phép nhân (*) ở dòng 3.

15.3 Bài tập

1. Giả sử a, b, c là biến kiểu int với a = 8, b = 3 và c = 5. Xác định giá trị các biểu thức sau:

| | | | | | |
|-----------|--|-----------|--|-------------|--|
| a + b + c | | a % c * 2 | | a * (a % b) | |
|-----------|--|-----------|--|-------------|--|

| | |
|-------------|--|
| $a / b - c$ | |
| $a + c / a$ | |
| $a \% b$ | |

| | |
|-----------------------|--|
| $2 * b + 3 * (a - c)$ | |
| $c * (b / a)$ | |
| $(a * b) \% c$ | |

| | |
|-------------------------|--|
| $a * (b + (c - 4 * 3))$ | |
| $5 * a - 6 / b$ | |
| $5 \% b \% c$ | |

2. Giả sử x, y, z là biến kiểu float với x = 8.8, y = 3.5 và z = 5.2. Xác định giá trị các biểu thức sau:

| | |
|-----------------------|--|
| $x + y + z$ | |
| $5 * y + 6 * (x - z)$ | |
| x / z | |
| $x \% z$ | |

| | |
|-------------------|--|
| $z / (y + x)$ | |
| $(z / y) + x$ | |
| $2 * y / 3 * z$ | |
| $2 * y / (3 * z)$ | |

| | |
|---------------------------------|--|
| $x / y - z * y$ | |
| $2.5 * x / z - (y + 6)$ | |
| $5 * 6 / ((x + y) / z)$ | |
| $x / y * (6 + ((z - y) + 3.4))$ | |

3. Cho chương trình C với các khai báo và khởi tạo các biến như sau:

```
int i = 8, j = 5;
float x = 0.005, y = -0.01;
char c = 'c', d = 'd';
```

Hãy xác định giá trị trả về của các biểu thức sau:

| | |
|--|--|
| $(3 * i - 2 * j) \% (4 * d - c)$ | |
| $2 * ((i / 4) + (6 * (j - 3)) \% (i + j - 4))$ | |
| $(i - 7 * j) \% (c + 3 * d) / (x - y)$ | |
| $-(i + j) * -1$ | |
| $++i$ | |
| $i++$ | |
| $i+++5$ | |
| $++i+5$ | |
| $j--$ | |
| $--j$ | |
| $j--+i$ | |
| $--j-5$ | |
| $++x$ | |
| $y--$ | |
| $i >= j$ | |

| | |
|--|--|
| $c < d$ | |
| $x >= 0$ | |
| $x < y$ | |
| $j != 6$ | |
| $c == 99$ | |
| $d != 100$ | |
| $5 * (i + j + 1) > 'd'$ | |
| $(3 * x + y) == 0$ | |
| $2 * x + (y == 0)$ | |
| $!(i < j)$ | |
| $!(d == 100)$ | |
| $!(x < 0)$ | |
| $(i > 0) \&\& (j < 6)$ | |
| $(i > 0) !! (j < 5)$ | |
| $(x > y) \&\& (i > 0) \parallel (j < 5)$ | |

4. Cho chương trình có các khai báo biến và khởi tạo như sau:

```
int i = 8, j = 5, k;
float x = 0.005, y = -0.01, z;
char a, b, c = 'c', d = 'd';
```

Xác định giá trị các biểu thức gán sau:

| | |
|---------------------|--|
| $k = (i + j * 4)$ | |
| $x = (x + y * 1.2)$ | |
| $i = j$ | |
| $k = (x + y)$ | |
| $k = c$ | |
| $i = j = 1.1$ | |
| $z = k = x$ | |
| $k = z = x$ | |

| | |
|------------------------|--|
| $z = i / j$ | |
| $a = b = d$ | |
| $y -= x$ | |
| $x *= 2$ | |
| $i /= j$ | |
| $i += 2$ | |
| $z = (x >= 0) ? x : 0$ | |
| $z = (y >= 0) ? y : 0$ | |

| | |
|-----------------------------------|--|
| $i \% = j$ | |
| $i += (j - 3)$ | |
| $k = (j == 5) ? i : j$ | |
| $k = (j > 5) ? i : j$ | |
| $i += j * = i /= 2$ | |
| $a = (c < d) ? c : d$ | |
| $i -= (j > 0) ? j : 0$ | |
| $i = (i * 9 * (3 + (8 * j / 3)))$ | |

Bài 16 :

MỘT SỐ HÀM CHUẨN THƯỜNG DÙNG

16.1 Các hàm chuyển đổi dữ liệu

16.1.1 atof

double atof(const char *s); ☞ Phải khai báo **math.h** hoặc **stdlib.h**

Chuyển đổi 1 chuỗi sang giá trị double.

Ví dụ: float f;
char *str = "12345.67";
f = atof(str);
Kết quả f = 12345.67;

16.1.2 atoi

int atoi(const char *s); ☞ Phải khai báo **stdlib.h**

Chuyển đổi 1 chuỗi sang giá trị int.

Ví dụ: int i;
char *str = "12345.67";
i = atoi(str);
Kết quả i = 12345

16.1.3 itoa

char *itoa(int value, char *string, int radix); ☞ Phải khai báo **stdlib.h**

Chuyển đổi số nguyên value sang chuỗi string theo cơ số radix.

Ví dụ: int number = 12345;
char string[25];
itoa(number, string, 10); //chuyển đổi number sang chuỗi theo cơ số 10
Kết quả string = "12345";
itoa(number, string, 2); //chuyển đổi number sang chuỗi theo cơ số 2
Kết quả string = "11000000111001";

16.1.4 tolower

int tolower(int ch); ☞ Phải khai báo **ctype.h**

Đổi chữ hoa sang chữ thường.

Ví dụ: int len, i;
char *string = "THIS IS A STRING";
len = strlen(string);
for (i = 0; i < len; i++)
string[i] = tolower(string[i]); //đổi từ kí tự trong string thành chữ thường

16.1.5 toupper

int toupper(int ch); ☞ Phải khai báo **ctype.h**

Đổi chữ thường sang chữ hoa.

Ví dụ: int len, i;
char *string = "this is a string";
len = strlen(string);
for (i = 0; i < len; i++)
string[i] = toupper(string[i]); //đổi từ kí tự trong string thành chữ thường

16.2 Các hàm xử lý chuỗi ký tự

16.2.1 strcat

char *strcat(char *dest, const char *src); ☞ Phải khai báo **string.h**

Thêm chuỗi src vào sau chuỗi dest.

16.2.2 strcpy

char *strcpy(char *dest, const char *src); ☞ Phải khai báo **string.h**

Chép chuỗi src vào dest.

Ví dụ: `char destination[25];`
`char *blank = " ", *c = "C++", *borland = "Borland";`
`strcpy(destination, borland); //chép chuỗi borland vào destination`
`strcat(destination, blank); //thêm chuỗi blank vào sau chuỗi destination`
`strcat(destination, c); //thêm chuỗi c vào sau chuỗi destination`

16.2.3 strcmp

int strcmp(const char *s1, const char *s2); ☞ Phải khai báo **string.h**

So sánh chuỗi s1 với chuỗi s2. Kết quả trả về:

- < 0 nếu s1 < s2
- = 0 nếu s1 = s2
- > 0 nếu s1 > s2

Ví dụ: `char *buf1 = "aaa", *buf2 = "bbb", *buf3 = "aaa";`
`strcmp(buf1, buf2); //kết quả trả về - 1`
`strcmp(buf1, buf3); //kết quả trả về 0`
`strcmp(buf2, buf3); //kết quả trả về 1`

16.2.4 strcmpi

int strcmpi(const char *s1, const char *s2); ☞ Phải khai báo **string.h**

So sánh chuỗi s1 với chuỗi s2 không phân biệt chữ hoa, chữ thường. Kết quả trả về:

- < 0 nếu s1 < s2
- = 0 nếu s1 = s2
- > 0 nếu s1 > s2

Ví dụ: `char *buf1 = "aaa", *buf2 = "AAA";`
`strcmpi(buf1, buf2); //kết quả trả về 0`

16.2.5 strlwr

char *strlwr(char *s); ☞ Phải khai báo **string.h**

Chuyển chuỗi s sang chữ thường

Ví dụ: `char *s = "Borland C";`
`s = strlwr(s); //kết quả s = "borland c"`

16.2.6strupr

char *strupr(char *s); ☞ Phải khai báo **string.h**

Chuyển chuỗi s sang chữ hoa

Ví dụ: `char *s = "Borland C";`
`s = strupr(s); //kết quả s = "BORLAND C"`

16.2.7 strlen

int strlen(const char *s); ☞ Phải khai báo **string.h**

Trả về độ dài chuỗi s.

Ví dụ: char *s = "Borland C";

int len_s;

len_s = strlen(s); //kết quả len_s = 9

16.3 Các hàm toán học

16.3.1 abs

int abs(int x); ☞ Phải khai báo **stdlib.h**

Cho giá trị tuyệt đối của số nguyên x.

Ví dụ: int num = - 123;

num = abs(num); //kết quả num = 123

16.3.2 labs

long int labs(long int x); ☞ Phải khai báo **stdlib.h**

Cho giá trị tuyệt đối của số nguyên dài x.

Ví dụ: int num = - 12345678L;

num = labs(num); //kết quả num = 12345678

16.3.3 rand

int rand(void); ☞ Phải khai báo **stdlib.h**

Cho 1 giá trị ngẫu nhiên từ 0 đến 32767

Ví dụ: int num;

randomize(); //dùng hàm này để khởi đầu bộ số ngẫu nhiên

num = rand(); //kết quả num = 1 con số trong khoảng 0..32767

16.3.4 random

int random(int num); ☞ Phải khai báo **stdlib.h**

Cho 1 giá trị ngẫu nhiên từ 0 đến 32767

Ví dụ: int n;

randomize();

n = random(100); //kết quả n = 1 con số trong khoảng 0..99

16.3.5 pow

double pow(double x, double y); ☞ Phải khai báo **math.h**

Tính x mũ y

Ví dụ: double x = 2.0, y = 3.0, z;

z = pow(x, y); //kết quả z = 8.0

16.3.6 sqrt

double sqrt(double x); ☞ Phải khai báo **math.h**

Tính căn bậc 2 của x.

Ví dụ: double x = 4.0, y;

y = sqrt(x); //kết quả y = 2.0

16.4 Các hàm xử lý file

16.4.1 rewind

```
void rewind(FILE *stream);
```

☞ Phải khai báo **stdio.h**

Đưa con trỏ về đầu file.

16.4.2 ftell

```
long ftell(FILE *stream);
```

☞ Phải khai báo **stdio.h**

Trả về vị trí con trỏ file hiện tại.

16.4.3 fseek

```
int fseek(FILE *stream, long offset, int whence);
```

 ☞ Phải khai báo **stdio.h**

Di chuyển con trỏ file đến vị trí mong muốn

- **long offset**: chỉ ra số byte kể từ vị trí trước đó đến vị trí bắt đầu đọc
- **int whence**: chỉ ra điểm xuất phát để tính offset gồm các giá trị sau: **SEEK_SET** (đầu tập tin), **SEEK_CUR** (tại vị trí con trỏ hiện hành), **SEEK_END** (cuối tập tin).

