CHƯƠNG 4: CẤU TRÚC VÒNG LẶP

Mã chương/ bài:MH18-04

Mục tiêu:

- Trình bày ý nghĩa của vòng lặp ;
- Trình bày cú pháp, công dụng của lệnh for, while, do…while;
- Trình bày ý nghĩa và cách sử dụng lệnh break, continue;
- Giải một số bài toán sử dụng lệnh for, while, do…while ;
- Sử dụng được các vòng lặp lồng nhau.
- Thực hiện các thao tác an toàn với máy tính.

4.1. Lệnh for

Mục tiêu: Hiểu cú pháp và có thể vận dụng vòng lặp for để giải quyết bài toán.

Cú pháp : for (biểu thức 1; biểu thức 2; biểu thức 3)

Lệnh hoặc khối lệnh ;

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.

Lưu ý :

+ Từ khóa for phải viết bằng chữ thường

+ Nếu là khối lệnh thì phải đặt trong dấu { }

+ 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 : Nhập n và tính tổng S = 1 + 2 + …..+ n

```
#include <stdio.h>
#include <conio.h>
void main()
{
            int i, n, s = 0;
            printf("Nhap vao so n: ");
      scanf("%d", &n);
            i = 0;
            for(i = 0; i<=n; i++)
            s = s + i;         //hoac s += i;
            printf("Tong: %d", s);
      getch();
}
```

Bài tập : Dùng vòng lặp for để :

1. Viết chương trình tính tổng bình phương các số lẻ từ 1 đến N. (N nhập vào từ bàn phím) .

2. Viết chương trình nhập vào N rồi tính giai thừa của N. (N nhập vào từ bàn phím).

3. Hãy làm theo yêu cầu sau:

-Viết chương trình kiểm tra n có phải là số nguyên tố hay không, với số n được nhập vào từ bàn phím

-Viết chương trình nhập vào số n và in ra các số nguyên tố có từ 1-> n, đếm có bao nhiêu số nguyên tố như vậy.

4. Tính các tổng sau:

S=1+2+3+…+n

S= $1^2+2^2+3+2+\ldots+n^2$

S=1/1+1/2+1/3+…+1/n

5. Viết chương trình in ra các số từ 1 đến 100 theo dạng sau:

1 2... 10

11 12 13.....20

.....

92 93..... 100

<u>Hướng dẫn</u>

3.

❖Thuật toán:

- Khai báo biến n, i
- Nhập số n
- Thuật toán kiểm tra n có phải là số nguyên tố không:
  - Cho i chạy từ 2 đến n
    - Nếu n%i = 0 thì thoát;
  - Nếu i = n thì in ra màn hình n là số nguyên tố
  - Ngược lại, n không phải là số nguyên tố

❖Thuật toán:

- Khai báo biến n, i, j
- Nhập số n
- Tìm các số nguyên tố có được từ 1->n:
  - Khởi tạo biến đếm count=0
  - Cho i chạy từ 2 đến n
    - Cho j chạy từ 2 đến i
      - Nếu i%j = 0 thì thoát
    - Nếu j=i thì:
      - i là số nguyên tố
      - count=count+1;

- In giá trị count và các số nguyên tố tìm được ra màn hình

❖Chương trình:

4.

❖Thuật toán:
- Khai báo biến S, i, n
- Nhập n từ bàn phím
- Tính tổng S:
    - S=0
    - Cho i chạy từ 1 đến n
        - S=S+i
- Xuất S ra màn hình

❖Thuật toán:
- Khai báo biến S, i, n
- Nhập n từ bàn phím
- Tính tổng S:
    - S=0
    - Cho i chạy từ 1 đến n
        - S=S+i*i hoặc S=S+pow(i,2)
- Xuất S ra màn hình

❖Thuật toán:
- Khai báo biến S, i, n
- Nhập n từ bàn phím
- Tính tổng S:
    - S=0
    - Cho i chạy từ 1 đến n
        - S=S+1/i
- Xuất S ra màn hình

5:

❖Thuật toán:
- Khai báo biến i
- Tạo bảng số:
    - Cho i chạy từ 1 đến 100 làm
        - Nếu i%10= =1 thì in xuống dòng và i ra màn hình
        - Ngược lại, in giá trị i ra màn hình

4.2. Lệnh break

***Mục tiêu**: Hiểu và biết cách sử dụng lệnh break trong chương trình.*

Thông thường lệnh break dùng để thoát khỏi vòng lặp không xác định điều kiện dừng hoặc người dùng muốn dừng vòng lặp theo điều kiện do người dùng 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: Viết chương trình đọc từ bàn phím một số nguyên n (1≤n≤10) rồi đưa ra tiếng Anh của số đó. Chẳng hạn, nếu gõ vào n = 4 thì in ra Four.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
main()
{
      clrscr();
      //khai bao bien
      int so;
      char *kq;
      //nhap tuoi
      printf("nhap so bat ky nam trong khoang [1,10]: ");
      scanf("%d",&so);
      ///chuyen so vua nhap sang tieng anh
      switch(so)
      {
           case 1: kq="One";break;
           case 2: kq="Two";break;
           case 3: kq="Three";break;
           case 4: kq="Four";break;
           case 5: kq="Five";break;
           case 6: kq="Six";break;
           case 7: kq="Seven";break;
           case 8: kq="Eight";break;
           case 9: kq="Nine";break;
           case 10: kq="Ten";break;
      }
      printf("Trong tieng Anh\n");
      printf("%d : %s",so,kq);
      getch();
}
```

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

4.3. Lệnh continue

***Mục tiêu**: Hiểu và biết cách sử dụng lệnh break trong chương trình.*

Được dùng trong vòng lặp for, while, do…while. Khi lệnh continue thi hành thì 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.

4.4. Lệnh while

***Mục tiêu**: Hiểu và biết cách sử dụng vòng lặp while khi giải quyết bài toán.*

Cú pháp : while (biểu thức)

Khối lệnh;

Giải thích :

+ 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.

Chú ý:

Từ khóa while phải viết bằng chữ thường

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

Ví dụ: Viết chương trình nhập vào một số n và in ra tổng các số đó

```
#include <stdio.h>
main()
{
      int n,s;
      printf("\n Chuong trinh tinh tong cac so tu nhien tu
1 den");
      scanf("%d",&n);
      s=0;
      while(n>0)
           {
                s=s+n;
                n=n-1;
           }
      printf("\nTong tim duoc la: %d",s);
```

```
        getch();
}
```

Bài tập: Dùng vòng lặp while để:

1. Viết chương trình tính tổng bình phương các số lẻ từ 1 đến N.

2. Viết chương trình nhập vào N rồi tính giai thừa của N.

3. Hãy làm theo yêu cầu sau:

-Viết chương trình kiểm tra n có phải là số nguyên tố hay không, với số n được nhập vào từ bàn phím

-Viết chương trình nhập vào số n và in ra các số nguyên tố có từ 1-> n, đếm có bao nhiêu số nguyên tố như vậy.

4. Tính các tổng sau:

S=1+2+3+…+n

S= $1^2+2^2+3^2+\ldots+n^2$

S=1/1+1/2+1/3+…+1/n

5. Viết chương trình in ra các số từ 1 đến 100 theo dạng sau:

1 2... 10

11 12 13.....20

.....

92 93..... 100

4.5. Lệnh do..while

***Mục tiêu**: Hiểu và biết cách sử dụng vòng lặp do..while khi giải quyết bài toán.*

❖ ***Cú pháp:***

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

❖ ***Giải thích:***

+ 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.

Chú ý:

Từ khóa do, while phải viết bằng chữ thường

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

Ví dụ : Viết chương trình nhập vào một số n và in ra tổng các số đó (sinh viên tự làm)

Bài tập: Dùng vòng lặp do..while để:

1. Viết chương trình tính tổng bình phương các số lẻ từ 1 đến N.
2. Viết chương trình nhập vào N rồi tính giai thừa của N.
3. Hãy làm theo yêu cầu sau:

-Viết chương trình kiểm tra n có phải là số nguyên tố hay không, với số n được nhập vào từ bàn phím

-Viết chương trình nhập vào số n và in ra các số nguyên tố có từ 1-> n, đếm có bao nhiêu số nguyên tố như vậy.

4. Tính các tổng sau:

S=1+2+3+…+n

S= $1^2+2^2+3^2+\ldots+n^2$

S=1/1+1/2+1/3+…+1/n

5. Viết chương trình in ra các số từ 1 đến 100 theo dạng sau:

1 2... 10

11 12 13.....20

.....

92 93..... 100

4.6. Vòng lặp lồng nhau

***Mục tiêu**: Biết cách kết hợp để các vòng lặp lồng nhau khi giải quyết bài toán có tính phức tạp.*

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 dễ gây nhầm lẫn khi lập trình cũng như kiểm soát chương trình.

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

```
#include <stdio.h>
#include <conio.h>
void main(void)
{
      int i, ij, idai, irong;
      printf("Nhap vao chieu dai: ");
      scanf("%d", &idai);
      printf("Nhap vao chieu rong: ");
      scanf("%d", &irong);
      for (i = 1; i <= irong; i++)
      {
            for (ij = 1; ij <= idai; ij++)
            //in mot hang voi chieu dai dau *
                  printf("*");
            printf("\n");
      }
getch();
}
```

Bài tập : Sử dụng vòng lặp lồng nhau để :

1. Viết chương trình in các số nguyên tố từ 1 đến n.(n được nhập từ bàn phím)

2. Viết chương trình tính :

S = 1 + 2! + 3! +...+n!

4.7. So sánh sự khác nhau của các vòng lặp.

***Mục tiêu**: Hiểu và biết ở trường hợp nào thì dùng vòng lặp nào để giải quyết các bài toán.*

- 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.

4.8. Bài tập thực hành

1. Viết chương trình in ra bảng mã ASCII

2. Viết chương trình nhập vào một số nguyên rồi in ra tất cả các ước số của số đó.

3. Viết chương trình vẽ một tam giác cân bằng các dấu *

4. Viết chương trình tính tổng nghịch đảo của N số nguyên đầu tiên theo công thức : S = 1 + 1/2 + 1/3 + … + 1/N

5. Viết chương trình tìm USCLN, BSCNN của 2 số.

6. Viết chương trình vẽ một tam giác cân rỗng bằng các dấu *.

7. Viết chương trình vẽ hình chữ nhật rỗng bằng các dấu *.

8. Viết chương trình nhập vào một số và kiểm tra xem số đó có phải là số nguyên tố hay không?

9. Viết chương trình tính x^n với x, n được nhập vào từ bàn phím.

10. 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.

CHƯƠNG 5: HÀM

Mã chương: MH18-05

Ý nghĩa:

Chương trình viết bằng ngôn ngữ C gồm 1 dãy các hàm trong đó có 1 hàm chính là main và chương trình bắt đầu từ main.

Mục tiêu:

- Trình bày khái niệm và cách khai báo về hàm ;
- Trình bày được cách truyền tham số, tham biến, tham trị ;
- Sử dụng biến cục bộ, toàn cục trong hàm ;
- Sử dụng tiền xử lý #define.
- Thực hiện các thao tác an toàn với máy tính.

5.1. Các ví dụ về hàm

Mục tiêu:

- *Trình bày khái niệm và cách khai báo về hàm ;*
- *Thực hiện các thao tác an toàn với máy tính.*

5.1.1. Khái niệm về hàm

Trong những chương trình lớn, có thể có những đoạn chương trình viết lặp đi lặp lại nhiều lần, để tránh rườm rà và mất thời gian khi viết chương trình; người ta thường phân chia chương trình thành nhiều module, mỗi module giải quyết một công việc nào đó. Các module như vậy gọi là các chương trình con.

Một tiện lợi khác của việc sử dụng chương trình con là ta có thể dễ dàng kiểm tra xác định tính đúng đắn của nó trước khi ráp nối vào chương trình chính và do đó việc xác định sai sót để tiến hành hiệu đính trong chương trình chính sẽ thuận lợi hơn. Trong C, chương trình con được gọi là hàm. Hàm trong C có thể trả về kết quả thông qua tên hàm hay có thể không trả về kết quả.

Hàm có hai loại: hàm chuẩn và hàm tự định nghĩa. Trong chương này, ta chú trọng đến cách định nghĩa hàm và cách sử dụng các hàm đó.

Một hàm khi được định nghĩa thì có thể sử dụng bất cứ đâu trong chương trình. Trong C, một chương trình bắt đầu thực thi bằng hàm main.

Ví dụ 1: Ta có hàm max để tìm số lớn giữa 2 số nguyên a, b như sau:

```
int max(int a, int b)
{
      return (a>b) ? a:b;
}
```

Ví dụ 2: Ta có chương trình chính (hàm main) dùng để nhập vào 2 số nguyên a,b và in ra màn hình số lớn trong 2 số

```
#include <stdio.h>
#include <conio.h>
int max(int a, int b)
{
      return (a>b) ? a:b;
}

void main()
{
      int a, b, c;
      printf("\n Nhap vao 3 so a, b,c    ");
      scanf("%d%d%d",&a,&b,&c);
      printf("\n So lon la %d",max(a, max(b,c)));
      getch();
}
```

❖ ***Hàm thư viện***

Hàm thư viện là những hàm đã được định nghĩa sẵn trong một thư viện nào đó, muốn sử dụng các hàm thư viện thì phải khai báo thư viện trước khi sử dụng bằng lệnh:

#include <tên thư viện.h>

Một số thư viện:

alloc.h	assert.h	bcd.h	bios.h	complex.h
conio.h	ctype.h	dir.h	dirent.h	dos.h
errno.h	fcntl.h	float.h	fstream.h	grneric.h
graphics.h	io.h	iomanip.h	iostream.h	limits.h

locale.h	malloc.h	math.h	mem.h	process.h
setjmp.h	share.h	signal.h	stdarg.h	stddef.h
stdio.h	stdiostr.h	stdlib.h	stream.h	string.h
strstrea.h	sys\stat.h	sys\timeb.h	sys\types.h	time.h
values.h				

Ý nghĩa của một số thư viện thường dùng:

1. stdio.h: Thư viện chứa các hàm vào/ ra chuẩn (standard input/output). Gồm các hàm printf(), scanf(), getc(), putc(), gets(), puts(), fflush(), fopen(), fclose(), fread(), fwrite(), getchar(), putchar(), getw(), putw()…

2. conio.h: Thư viện chứa các hàm vào ra trong chế độ DOS (DOS console). Gồm các hàm clrscr(), getch(), getche(), getpass(), cgets(), cputs(), putch(), clreol(),…

3. math.h: Thư viện chứa các hàm tính toán gồm các hàm abs(), sqrt(), log(). log10(), sin(), cos(), tan(), acos(), asin(), atan(), pow(), exp(),…

4. alloc.h: Thư viện chứa các hàm liên quan đến việc quản lý bộ nhơ. Gồm các hàm calloc(), realloc(), malloc(), free(), farmalloc(), farcalloc(), farfree(), …

5. io.h: Thư viện chứa các hàm vào ra cấp thấp. Gồm các hàm open(), _open(), read(), _read(), close(), _close(), creat(), _creat(), creatnew(), eof(), filelength(), lock(),…

6. graphics.h: Thư viện chứa các hàm liên quan đến đồ họa. Gồm initgraph(), line(), circle(), putpixel(), getpixel(), setcolor(), …

...

Muốn sử dụng các hàm thư viện thì ta phải xem cú pháp của các hàm và sử dụng theo đúng cú pháp (xem trong phần trợ giúp của Turbo C).

❖ ***Hàm người dùng***

Hàm người dùng là những hàm do người lập trình tự tạo ra nhằm đáp ứng nhu cầu xử lý của mình.

5.1.2. Các ví dụ về hàm

Ví dụ 1:

```
1         #include <stdio.h>
2         #include <conio.h>
3
4
5         // khai bao prototype 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        }
```

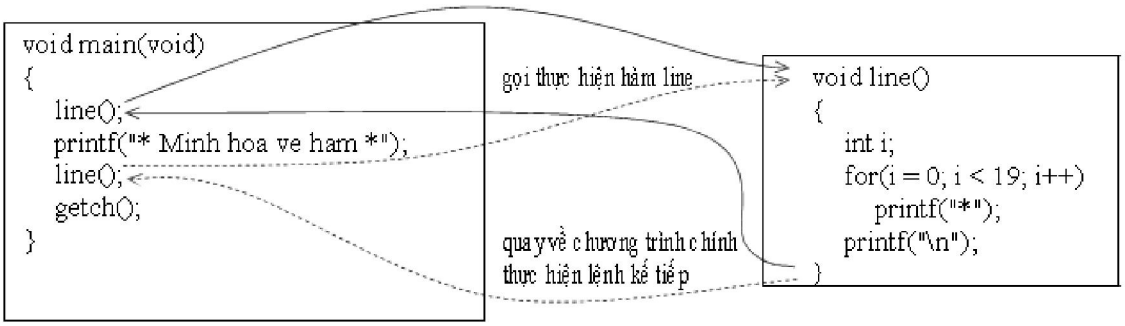
Giải thích chương trình

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:

```
1       #include <stdio.h>
2       #include <conio.h>
3
4       // khai bao prototype int power(int, int);
5
6       // ham tinh so mu
7       int power(int ix, int in)
8       {
9             int i, ip = 1;
10            for(i = 1; i <= in; i++)
11                 ip *= ix;
12            return ip;
13      }
14
15      void main(void)
16      {
17            printf("2 mu 2 = %d.\n", power(2, 2));
18            printf("2 mu 3 = %d.\n", power(2, 3));
19            getch();
20      }
```

Kết quả in ra màn hình

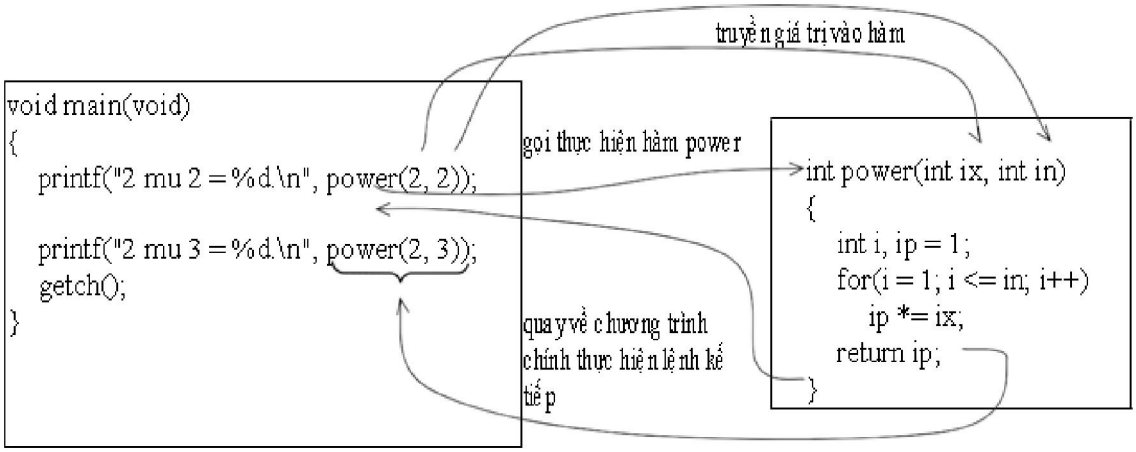
2 mu 2 = 4.

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:

```
1       #include <stdio.h>
2       #include <conio.h>
3       // khai bao prototype
4       void time(int & , int &);
5       // ham doi phut thanh gio:phut
6       void time(int &ig, int &ip)
7       {
8             ig = ip / 60;
9             ip %= 60;
10      }
11      void main(void)
12      {
13            int igio, iphut;
14            printf("Nhap vao so phut : ");
15            scanf("%d", &iphut); time(igio, iphut);
16            printf("%02d:%02d\n", igio, iphut);
17            getch();
18      }
```

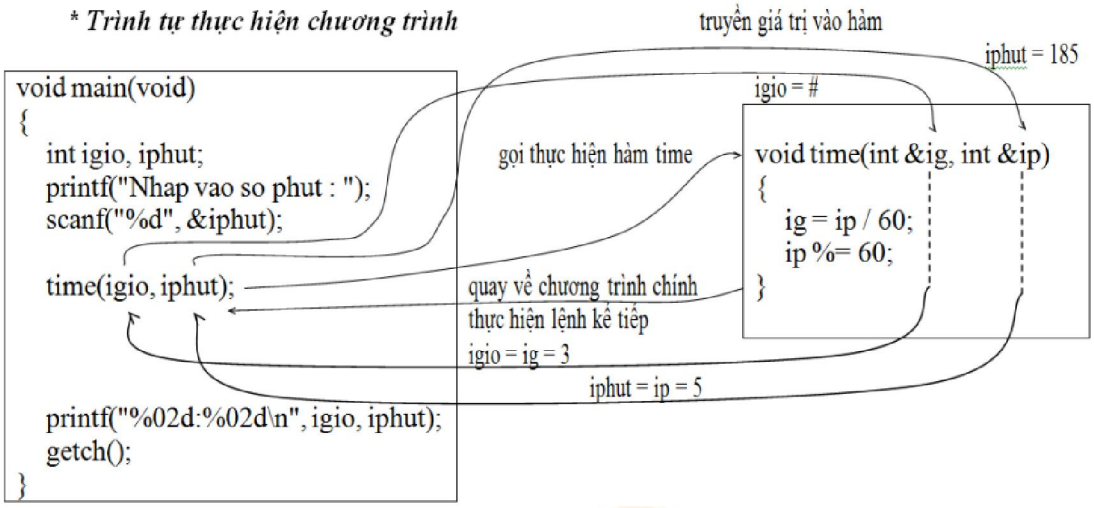
❖ Kết quả in ra màn hình

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.



5.2. THAM SỐ DẠNG THAM BIẾN VÀ THAM TRỊ

Mục tiêu:

- *Trình bày được cách truyền tham số, tham biến, tham trị ;*
- *Thực hiện các thao tác an toàn với máy tính.*

5.2.1. Tham số dạng tham trị

Mặc nhiên, việc truyền tham số cho hàm trong C là truyền theo giá trị; nghĩa là các giá trị thực (tham số thực) không bị thay đổi giá trị khi truyền cho các tham số hình thức

Ví dụ 1: Giả sử ta muốn in ra nhiều dòng, mỗi dòng 50 ký tự nào đó. Để đơn giản ta viết một hàm, nhiệm vụ của hàm này là in ra trên một dòng 50 ký tự nào đó. Hàm này có tên là InKT.

```
#include <stdio.h>
#include <conio.h>
void InKT(char ch)
{
      int i;
      for(i=1;i<=50;i++) printf("%c",ch);
      printf("\n");
}
int main()
{
      char c = 'A';
      InKT('*'); /* In ra 50 dau * */ InKT('+');
      InKT(c);
      return 0;
```

```
}
```

Chú ý:

- Trong hàm InKT ở trên, biến ch gọi là tham số hình thức được truyền bằng giá trị (gọi là tham trị của hàm). Các tham trị của hàm coi như là một biến cục bộ trong hàm và chúng được sử dụng như là dữ liệu đầu vào của hàm.

- Khi chương trình con được gọi để thi hành, tham trị được cấp ô nhớ và nhận giá trị là bản sao giá trị của tham số thực. Do đó, mặc dù tham trị cũng là biến, nhưng việc thay đổi giá trị của chúng không có ý nghĩa gì đối với bên ngoài hàm, không ảnh hưởng đến chương trình chính, nghĩa là không làm ảnh hưởng đến tham số thực tương ứng.

Ví dụ 2: Ta xét chương trình sau đây:

```
#include <stdio.h>
#include <conio.h>
int   hoanvi(int a, int b)
{
      int t;
      t=a; /*Đoạn này hoán vị  giá trị  của 2 biến a, b*/
      a=b;
      b=t;
      printf("\Ben trong ham a=%d   , b=%d",a,b);
      return 0;
}
void main()
{
      int a, b;
      clrscr();
      printf("\n Nhap vao 2 so nguyen a, b:");
      scanf("%d%d",&a,&b);
      printf("\n Truoc khi goi ham hoan vi a=%d, b=%d", a,
 b);
      hoanvi(a,b);
      printf("\n Sau khi goi ham hoan vi a=%d ,b=%d",a,b);
      getch();
}
```

Kết quả thực hiện chương trình:

```
Nhap vao 2 so nguyen a, b:6 5

Truoc khi goi ham hoan vi a=6  ,b=5
Ben trong ham a=5   , b=6
Sau khi goi ham hoan vi a=6  ,b=5
```

❖ ***Giải thích:***

- Nhập vào 2 số 6 và 5 (a=6, b=5)

- Trước khi gọi hàm hoán vị thì a=6, b=5
- Bên trong hàm hoán vị a=5, b=6
- Khi ra khỏi hàm hoán vị thì a=6, b=5

5.2.2. Bài tập thực hành về tham trị

1. Viết chương trình giải phương trình bậc nhất ax+b=0.

Hướng dẫn:

- Xây dựng hàm giải phương trình bậc nhất: void giaiptb1(int a, int b)
- Truyền tham trị cho hàm giaiptb1(int a, int b)
- Gọi hàm giaiptb1(int a, int b) vừa xây dựng thực hiện trong chương trình chính

2. Viết chương trình tính tổng các số nguyên chẵn từ 1 đến n.

Hướng dẫn:

- Xây dựng hàm tính tổng các số nguyên chẵn từ 1 đến n:

  int tinhtongchan (int n)
- Truyền tham trị cho hàm int tinhtongchan (int n)
- Gọi hàm tinhtongchan (int n) vừa xây dựng thực hiện trong chương trình chính

5.2.3. **Tham số dạng tham biến**

Trong đoạn chương trình trên, nếu ta muốn sau khi kết thúc chương trình con giá trị của a, b thay đổi thì ta phải đặt tham số hình thức là các con trỏ, còn tham số thực tế là địa chỉ của các biến.

Lúc này mọi sự thay đổi trên vùng nhớ được quản lý bởi con trỏ là các tham số hình thức của hàm thì sẽ ảnh hưởng đến vùng nhớ đang được quản lý bởi tham số thực tế tương ứng (cần để ý rằng vùng nhớ này chính là các biến ta cần thay đổi giá trị). Người ta thường áp dụng cách này đối với các dữ liệu đầu ra của hàm.

Ví dụ: Xét chương trình sau đây:

```
#include <stdio.h>
#include <conio.h>
long  hoanvi(long *a, long *b)
/* Khai báo tham số  hình thức *a, *b là các con trỏ  kiểu
long */
```

```
{
      long t;
      t=*a;     /*gán nội dung của x cho t*/
      *a=*b;    /*Gán nội dung của b cho a*/
      *b=t;     /*Gán nội dung của t cho b*/
      printf("\n Ben trong ham a=%ld     , b=%ld",*a,*b);
      /*In ra nội dung của a, b*/
      return 0;
}

void main()
{
      long a, b;
      clrscr();
      printf("\n Nhap vao 2 so nguyen a, b:");
      scanf("%ld%ld",&a,&b);
     printf("\n Truoc khi goi ham hoan vi a=%ld, b=%ld",
a,b);
     hoanvi(&a,&b); /* Phải là địa chỉ  của a và b */
     printf("\n Sau khi goi ham hoan vi a=%ld, b=%ld",
a,b);
     getch();
}
```

Kết quả thực hiện chương trình sau.

```
Nhap vao 2 so nguyen a, b: 5 6

Truoc khi goi ham hoan vi a=5  ,b=6
Ben trong ham a=6   , b=5
Sau khi goi ham hoan vi a=6  ,b=5
```

❖ ***Giải thích:***

- Nhập vào 2 số 5, 6 (a=5, b=6)
- Trước khi gọi hàm hoanvi thì a=5, b=6
- Trong hàm hoanvi (khi đã hoán vị) thì a=6, b=5
- Khi ra khỏi hàm hoán vị thì a=6, b=6

5.2.4. Bài tập thực hành

1. Viết chương trình giải phương trình bậc nhất ax+b=0 (a#0).

Hướng dẫn:

- Xây dựng hàm giải phương trình bậc nhất:

  float giaiptb1(int a, int b, float *x)
- Truyền tham trị cho hàm giaiptb1(int a, int b, float *x)
- Gọi hàm giaiptb1(int a, int b, float *x) vừa xây dựng thực hiện trong chương trình chính