

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN
KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP THỰC HÀNH
HỌC PHẦN CƠ SỞ KỸ THUẬT LẬP TRÌNH C

Trình độ đào tạo : Đại học

Hệ đào tạo : Chính quy

MỤC LỤC

BÀI 1 CÁC CẤU TRÚC ĐIỀU KHIỂN CƠ BẢN	3
I. TÓM TẮT LÝ THUYẾT	3
II. BÀI TẬP	9
III. KẾT LUẬN	17
BÀI 2 HÀM CON	18
IV. TÓM TẮT LÝ THUYẾT	18
II. BÀI TẬP	25
III. KẾT LUẬN	29
BÀI 3 MẢNG VÀ CON TRỎ	30
I. TÓM TẮT LÝ THUYẾT	30
II. BÀI TẬP	32
III. KẾT LUẬN	39
BÀI 4 CHUỖI KÝ TỰ	41
I. TÓM TẮT LÝ THUYẾT	41
II. BÀI TẬP	42
III. KẾT LUẬN	45
BÀI 5 KIỂU DỮ LIỆU CÓ CẤU TRÚC	46
I. TÓM TẮT LÝ THUYẾT	46
II. BÀI TẬP	49
III. KẾT LUẬN	55
BÀI 6 TỆP TIN	56
I. TÓM TẮT LÝ THUYẾT	56
II. BÀI TẬP	58
III. KẾT LUẬN	63

BÀI 1 CÁC CẤU TRÚC ĐIỀU KHIỂN CƠ BẢN

Tìm hiểu và cài đặt các cấu trúc rẽ nhánh, lựa chọn, lặp và các ký hiệu phép toán trong ngôn ngữ C. Mô tả cách hoạt động và hướng dẫn chạy từng bước chương trình.

I. TÓM TẮT LÝ THUYẾT

I.1. Các ký hiệu

STT	KÝ HIỆU	DIỄN GIẢI	VÍ DỤ
1	{ }	Bắt đầu và kết thúc hàm hay khối lệnh.	void main() { }
2	;	Kết thúc khai báo biến, một lệnh, một lời gọi hàm, hay khai báo nguyên mẫu hàm.	int x; void NhapMang(int a[], int &n);
3	//	Chú thích (ghi chú) cho một dòng. Chỉ có tác dụng đối với người đọc chương trình.	//Ham nay dung de nhap mang void NhapMang(int a[], int &n);
4	/* */	Tương tự như ký hiệu //, nhưng cho trường hợp nhiều dòng.	/* Dau tien nhap vao n. Sau do nhap cho tung phan tu */ void NhapMang(int a[], int &n);

I.2. Các kiểu dữ liệu cơ bản trong C

STT	KIỂU	GHI CHÚ	KÍCH THƯỚC	ĐỊNH DẠNG
KIỂU LIÊN TỤC (SỐ THỰC)				
1	float		4 bytes	%f
2	double		8 bytes	%lf
3	long double		10 bytes	%lf
KIỂU RỜI RẠC (SỐ NGUYÊN)				
1	char	Ký tự	1 byte	%c
		Số nguyên	1 byte	%d
2	unsigned char	Số nguyên dương	1 byte	%d

3	int	Số nguyên	2 bytes	%d
4	unsigned int	Số nguyên dương	2 bytes	%u
5	long	Số nguyên	4 bytes	%ld
6	unsigned long	Số nguyên dương	4 bytes	%lu
7	char *	Chuỗi		%s

I.3. Các hàm cơ bản

STT	TÊN HÀM	THƯ VIỆN	DIỄN GIẢI	VÍ DỤ
1	printf	#include<stdio.h>	Xuất ra màn hình.	<pre>#include<stdio.h> #include<conio.h> #include<dos.h> void main() { int c = 1, n; clrscr(); printf("Nhập n:"); scanf("%d", &n); do{ textcolor(c); gotoxy(20, 10); cprintf("%d", n); c++; if (c>15) c = 1; delay(200); } while(!kbhit()); }</pre>
2	scanf	#include<stdio.h>	Lấy dữ liệu từ bàn phím.	
3	gotoxy	#include<conio.h>	Di chuyển dấu nháy đến tọa độ (x, y) trên màn hình văn bản.	
4	textcolor	#include<conio.h>	Đặt màu cho chữ (có giá trị từ 0 đến 15).	
5	cprintf	#include<stdio.h>	Xuất ra màn hình với màu chữ đã định liền trước đó.	
6	delay	#include<dos.h>	Dừng thực hiện lệnh tiếp sau một khoảng thời gian.	
7	kbhit	#include<conio.h>	Kiểm tra xem có nhấn phím.	

I.4. Cấu trúc rẽ nhánh

a. Cấu trúc if

```
if (biểu thức điều kiện)
{
    <khối lệnh> ;
}
```

Nếu biểu thức điều kiện cho kết quả khác không thì thực hiện khối lệnh.

I.5. Bảng ký hiệu các phép toán

STT	PHÉP TOÁN	Ý NGHĨA	GHI CHÚ
PHÉP TOÁN SỐ HỌC			
1	+	Cộng	
2	-	Trừ	
3	*	Nhân	
4	/	Chia lấy phần nguyên	
5	%	Chia lấy phần dư	
PHÉP TOÁN QUAN HỆ			
1	>	Lớn hơn	
2	<	Nhỏ hơn	
3	>=	Lớn hơn hoặc bằng	
4	<=	Nhỏ hơn hoặc bằng	
5	= =	Bằng nhau	
6	!=	Khác nhau	
PHÉP TOÁN LOGIC			
1	!	NOT	
2	&&	AND	
3		OR	
TOÁN TỬ TĂNG GIẢM			
1	++	Tăng 1	Nếu toán tử tăng giảm đặt trước thì tăng giảm trước rồi tính biểu thức hoặc ngược lại.
2	--	Giảm 1	
PHÉP TOÁN THAO TÁC TRÊN BIT			
1	&	AND	
2		OR	

STT	PHÉP TOÁN	Ý NGHĨA	GHI CHÚ
3	\wedge	XOR	
4	\ll	Dịch trái	
5	\gg	Dịch phải	
6	\sim	Lấy phần bù theo bit	

I.6. Các hàm cơ bản

STT	TÊN HÀM	THƯ VIỆN	DIỄN GIẢI	VÍ DỤ
1	printf	#include<stdio.h>	Xuất ra màn hình.	<pre>#include<stdio.h> #include<conio.h> #include<dos.h> void main() { int c = 1, n; clrscr(); printf("Nhập n:"); scanf("%d", &n); do{ textcolor(c); gotoxy(20, 10); cprintf("%d", n); c++; if (c>15) c = 1; delay(200); } while(!kbhit()); }</pre>
2	scanf	#include<stdio.h>	Lấy dữ liệu từ bàn phím.	
3	gotoxy	#include<conio.h>	Di chuyển dấu nháy đến tọa độ (x, y) trên màn hình văn bản.	
4	textcolor	#include<conio.h>	Đặt màu cho chữ (có giá trị từ 0 đến 15).	
5	cprintf	#include<stdio.h>	Xuất ra màn hình với màu chữ đã định liền trước đó.	
6	delay	#include<dos.h>	Dừng thực hiện lệnh tiếp sau một khoảng thời gian.	
7	kbhit	#include<conio.h>	Kiểm tra xem có nhấn phím.	

I.7. Cấu trúc rẽ nhánh

a. Cấu trúc if

```
if (biểu thức điều kiện)
{
    <khối lệnh> ;
}
```

```
}
```

Nếu biểu thức điều kiện cho kết quả khác không thì thực hiện khối lệnh

b. Cấu trúc if ... else

```
if (biểu thức điều kiện)
```

```
{  
    <khối lệnh 1>;  
}
```

```
else
```

```
{  
    <khối lệnh 2>;  
}
```

Nếu biểu thức điều kiện cho kết quả khác không thì thực hiện khối lệnh 1, ngược lại thì cho thực hiện khối lệnh thứ 2. Biểu thức điều kiện phải đặt trong cặp dấu ngoặc tròn.

I.8. Cấu trúc lựa chọn switch

```
switch (biểu thức)
```

```
{
```

```
    case n1:
```

```
        các  
        câu  
        lệnh ;  
        break  
        ;
```

```
    case n2:
```

```
        các  
        câu  
        lệnh ;  
        break  
        ;
```

```
    .....
```

```
    case nk:
```

```
        <các câu  
        lệnh> ;  
        break ;
```

- n_i là các **hằng số nguyên hoặc ký tự**.
- Phụ thuộc vào giá trị của biểu thức viết sau

switch, nếu: o Giá trị này = n_i thì thực hiện câu

lệnh sau case n_i .

o Khi giá trị biểu thức không thỏa tất cả các n_i thì thực hiện câu lệnh sau **default** nếu có, hoặc thoát khỏi câu lệnh **switch**.

o Khi chương trình đã thực hiện xong câu lệnh của **case** n_i nào đó thì nó sẽ thực hiện luôn các lệnh thuộc **case** bên dưới nó mà không xét lại điều kiện (do các n_i được xem như các nhãn) Vì vậy, để chương trình thoát khỏi lệnh **switch** sau khi thực hiện xong một trường hợp, ta dùng lệnh **break**.

I.9. Cấu trúc lặp

a. *for*

for (<biểu thức khởi gán>; <biểu thức điều kiện>; <biểu thức tăng/giảm>)

```
{  
    <khởi lệnh>;  
}
```

Bất kỳ biểu thức nào trong 3 biểu thức nói trên đều có thể vắng nhưng phải giữ dấu chấm phẩy (;).

Hoạt động:

Bước 1: Khởi gán cho biểu thức 1

Bước 2: Kiểm tra điều kiện của biểu thức 2

- Nếu **biểu thức 2** $\neq 0$ thì cho thực hiện các lệnh của vòng lặp, thực hiện biểu thức 3. Quay trở lại bước 2.
- Ngược lại thoát khỏi lặp.

b. *while*

```
< Khởi gán>  
while ( <biểu thức điều kiện> )  
{  
    lệnh/ khởi lệnh;  
    < tăng/giảm chỉ số lặp>;  
}
```

Lưu ý: Cách hoạt động của **while** giống **for**

c. do ... while

```
do
{
    < khối lệnh> ;
} while (biểu thức điều kiện);
```

Thực hiện khối lệnh cho đến khi biểu thức có giá trị bằng 0.

Lặp **while** kiểm tra điều kiện trước khi thực hiện lặp, còn vòng lặp **do...while** thực hiện lệnh lặp rồi mới kiểm tra điều kiện. Do đó vòng lặp **do...while** thực hiện lệnh ít nhất một lần

II. BÀI TẬP**II.1. Phương pháp chạy tay từng bước để tìm kết quả chương trình**

Xác định chương trình có sử dụng những biến nào.

Giá trị ban đầu của mỗi biến.

Những **biến nào sẽ bị thay đổi** trong quá trình chạy chương trình thì lập thành bảng có dạng sau:

Bước (Hoặc lần thực hiện)	Biến 1	Biến 2	...	Biến n	Kết quả in ra màn hình
0	Giá trị 0	Giá trị 0	...	Giá trị 0	
1	Giá trị 1	Giá trị 1	...	Giá trị 1	
2	Giá trị 2	Giá trị 2	...	Giá trị 2	
...	
...	

Ví dụ: Cho biết kết quả của đoạn chương trình sau: **void main()**

```
{
    int
    i, a
    = 4;
    clrscr();
    ;
```

```
for(i = 0 ; i<a;
    i++)
    printf("%"
    d\n", i);
}
```

Chương trình gồm 2 biến *i* và *a*, chỉ có biến *i* có giá trị thay đổi trong quá trình chạy chương trình nên ta lập bảng sau:

a có giá trị là 4

Bước thực hiện	Giá trị của biến <i>i</i>	Kết quả in ra màn hình
0	0	0
1	1	0 1
2	2	0 1 2
3	3	0 1 2 3
4	4	

Tại bước 4, giá trị của *i* = 4 vi phạm điều kiện lặp (*i*<*a*) nên vòng lặp kết thúc. Do đó kết quả in ra màn hình:

0
1
2
3

II.2. Bài tập cơ bản

a. Cấu trúc *if* / *if..else* và *switch*

1. Cho biết kết quả của đoạn chương trình sau: *int a=9, b=6;*

a++; a=a+b--;

a=a+ (--b);

```
if(a%2==0)
    printf("Gia tri cua a la
chan"); printf("Tong cua a va b la:
%d", a+b) ;
```

2. Cho biết kết quả của đoạn chương trình

sau: `int a=7, b=8;`

```
a++;
a=a+(b--
); --b;
a--; a=(--
a)+(--b);
if(a%2!=0)
    printf("\n a la so le");
else
    printf("\n a la so
chan"); printf("\na = %d",a);
```

3. Cho biết kết quả của đoạn chương trình

sau: `int x=5, y;`

```
y=x++ + 5;
printf("x=%d, y=%d\n", x,
y); y*=6;
x=y%7;
printf("x=%d,y=%d,y/x=%d", x, y, y/x);
```

4. Nhập vào hai số nguyên a, b. In ra màn hình giá trị lớn nhất.
5. Cho ba số a, b, c đọc vào từ bàn phím. Hãy tìm giá trị lớn nhất của ba số trên và in ra kết quả.
6. Cho ba số a, b, c đọc vào từ bàn phím. Hãy in ra màn hình theo thứ tự tăng dần các số. (Chỉ được dùng thêm hai biến phụ).
7. Viết chương trình nhập vào một số nguyên n gồm ba chữ số. Xuất ra màn

hình chữ số lớn nhất ở vị trí nào?

Ví dụ: $n=291$. Chữ số lớn nhất nằm ở hàng chục (9).

Viết chương trình nhập vào số nguyên n gồm ba chữ số. Xuất ra màn hình theo thứ tự tăng dần của các chữ số

Ví dụ: $n=291$. Xuất ra 129.

9. Nhập vào ngày, tháng, năm. Kiểm tra xem ngày, tháng, năm đó có hợp lệ hay không? In kết quả ra màn hình.
10. Nhập vào giờ, phút, giây. Kiểm tra xem giờ, phút, giây đó có hợp lệ hay không? In kết quả ra màn hình.
11. Viết chương trình nhập vào ngày, tháng, năm hợp lệ. Cho biết năm này có phải là năm nhuận hay không? In kết quả ra màn hình.
12. Viết chương trình tính diện tích và chu vi các hình: tam giác, hình vuông, hình chữ nhật và hình tròn với những thông tin cần được nhập từ bàn phím.
13. Viết chương trình tính tiền cước TAXI. Biết rằng:
 - KM đầu tiên là 5000^d.
 - 200m tiếp theo là 1000^d.
 - Nếu lớn hơn 30km thì mỗi km thêm sẽ là 3000^d.Hãy nhập số km sau đó in ra số tiền phải trả.
14. Nhập vào 3 số nguyên dương a, b, c . Kiểm tra xem 3 số đó có lập thành tam giác không? Nếu có hãy cho biết tam giác đó thuộc loại nào? (Cân, vuông, đều, ...).
15. Viết chương trình nhập vào số nguyên dương n . Kiểm tra xem n có phải là số chính phương hay không? (số chính phương là số khi lấy căn bậc 2 có kết quả là nguyên).

b. Cấu trúc lặp

16. Cho biết kết quả của đoạn chương trình sau:

```
int a=18;
for(int i=1; i<=a;
    i++) if(a%i=
=0)
```

```
printf("\t %d", i);
```

17. Cho biết kết quả của đoạn chương trình sau:

```
for(int i=0; i<5; i++)  
{  
    for(int j=0; j<=i; j++)  
        printf("%d\t",  
            j);  
    printf("\n");  
}
```

18. Cho biết kết quả của đoạn chương trình sau: *int i=10, s=0;*

```
while(i>0)  
{  
    if(i%2==0) s+=i;  
    else  
        if(i>5)  
            s+=2*i;  
    i--;  
}  
printf("s = %d",s);
```

19. Cho biết kết quả của đoạn chương trình sau: *int a=18, i=1;*

```
do{  
    if(a%i==0)  
        printf("\t %d",i);  
    i++;  
} while(i<=a);
```

20. Cho biết kết quả của đoạn chương trình sau: *int a=11, b=16, i=a;*

```
while( i<b )  
{  
    if(i%2==0)  
    {
```

```
        printf("\t %d", i); break;
    }
    i++;
}
```

Cho biết kết quả của đoạn chương trình sau: *int a=10, s=0, i=0;*

```
while( i<a )
{
    i++;
    if(i%2==0)
        continue; else s=s+i;
}
printf("s=%d",s);
```

22. Cho biết kết quả của đoạn chương trình

sau: *int i=1,s=0;*

```
while(1)
{
    s=s+i++;
    if(i%2)
        i=i+2;
    else
        i=i+1;
    if(i>20)
        break;
}
printf("%d",s);
```

23. Viết chương trình in ra màn hình hình chữ nhật đặc kích thước $m \times n$ (m, n nhập từ bàn phím).

Ví dụ: Nhập $m=5, n=4$

```
* * * * *
* * * * *
```

```

* * * * *
* * * * *

```

24. Viết chương trình in ra màn hình hình chữ nhật rỗng kích thước $m \times n$ (m, n nhập từ bàn phím).

Ví dụ: Nhập $m=5, n=4$

```

* * * * *
*           *
*           *
* * * * *

```

25. Viết chương trình in ra màn hình tam giác vuông cân đặc có độ cao h (h nhập từ bàn phím).

Ví dụ: Nhập $h=4$

```

*
* *
* * *
* * * *

```

26. Viết chương trình in ra màn hình tam giác cân rỗng có độ cao h (h nhập từ bàn phím).

Ví dụ: Nhập $h=4$

```

*
* *
*   *
* * * *

```

27. Viết chương trình in ra màn hình tam giác cân đặc có độ cao h (h nhập từ bàn phím).

Ví dụ: Nhập $h=4$

```

      *
     * * *
    * * * * *
   * * * * * *
  * * * * * * *

```

28. Viết chương trình in ra màn hình tam giác cân rỗng có độ cao h (h nhập từ bàn phím).

Ví dụ: Nhập $h=4$

```

      *
     *   *
    *     *
   *       *
  * *     * *

```

29. Viết chương trình nhập số nguyên dương n . Liệt kê n số nguyên tố đầu tiên.
30. Viết chương trình nhập vào hai số nguyên dương a và b . Tìm ước số chung lớn nhất và bội số chung nhỏ nhất của a và b .
31. Viết chương trình nhập vào một số nguyên n gồm tối đa 10 chữ số (4 bytes). In ra màn hình giá trị nhị phân của số trên. (Hướng dẫn: chia lấy dư cho 2 và xuất theo thứ tự ngược lại dùng hàm gotoxy, wherex, wherey).
32. Viết chương trình đếm số ước số của số nguyên dương

N. Ví dụ: $N=12$

số ước số của 12 là 6

33. Một số hoàn thiện là một số có tổng các ước số của nó (không kể nó) bằng chính nó. Hãy liệt kê các số hoàn thiện nhỏ hơn 5000.

Ví dụ: số 6 là số hoàn thiện vì tổng các ước số là $1+2+3=6$.

34. Nhập vào ngày, tháng, năm. Cho biết đó là ngày thứ mấy trong năm.

35. In ra dãy số Fibonacci

$$f_1 = f_0 = 1 ;$$

$$f_n = f_{n-1} + f_{n-2} ; \quad (n > 1)$$

II.3. Bài tập luyện tập và nâng cao

36. Cài đặt tất cả các lưu đồ đã vẽ ở chương 1.
37. Nhập vào ngày, tháng, năm. Kiểm tra xem ngày, tháng, năm đó có hợp lệ hay không, nếu hợp lệ cho biết ngày sau đó là bao nhiêu.

Ví dụ: Nhập 31/12/2003

Ngày sau đó 01/01/2004

38. Nhập vào ngày, tháng, năm. Kiểm tra xem ngày, tháng, năm đó có hợp lệ hay không, nếu hợp lệ cho biết ngày trước đó là bao nhiêu.

Ví dụ: Nhập 01/01/2003

Ngày trước đó 31/12/2002

39. (*) Nhập vào ngày, tháng, năm của năm 2003. Hãy kiểm tra xem dữ liệu có hợp lệ hay không? Nếu hợp lệ hãy cho biết đó là ngày thứ mấy trong tuần. (hai, ba, tư, ..., CN). (Hướng dẫn: lấy ngày 01 tháng

01 năm 2003 là ngày thứ tư làm mốc).

40. Nhập vào giờ, phút, giây. Kiểm tra xem giờ, phút, giây đó có hợp lệ hay không, nếu hợp lệ cho biết giờ sau đó 1 giây là bao nhiêu.

Ví dụ: *Nhập 01:59:59*

Giờ sau đó 1 giây 02:00:00

41. Nhập vào giờ, phút, giây. Kiểm tra xem giờ, phút, giây đó có hợp lệ hay không, nếu hợp lệ cho biết giờ trước đó 1 giây là bao nhiêu.

Ví dụ: *Nhập 02:00:00*

Giờ trước đó 1 giây 01:59:59

III. KẾT LUẬN

Cấu trúc lặp và rẽ nhánh (lựa chọn) là hai cấu trúc chính hình thành nên chương trình. Dựa vào những cấu trúc điều khiển này ta có thể xây dựng thành những chương trình phức tạp hơn. Vì vậy phải nắm rõ cách hoạt động của những cấu trúc điều khiển này để cài đặt đúng yêu cầu bài toán. Khi sử dụng phải **lưu ý điều kiện thực hiện hay kết thúc** của một thao tác nào đó. **Bên trong một phát biểu điều khiển phải là một lệnh hay một khối lệnh**

(khối lệnh được đặt bên trong cặp dấu ngoặc {}).

Những **biến không phụ thuộc vào vòng lặp nên đặt bên ngoài vòng lặp**. Khi sử dụng **cấu trúc điều khiển lồng nhau phải lưu ý vị trí mở ngoặc hay đóng ngoặc cho hợp lý**.

BÀI 2 HÀM CON

Trình bày cấu trúc của một chương trình, các bước xây dựng cài đặt chương trình theo phương pháp thủ tục hàm và một số kỹ thuật liên quan.

IV. TÓM TẮT LÝ THUYẾT

I.1. Khái niệm

Hàm là một đoạn chương trình độc lập **thực hiện trọn vẹn một công việc nhất**

định sau đó trả về giá trị cho chương trình gọi nó, hay nói cách khác hàm là sự chia nhỏ của chương trình.

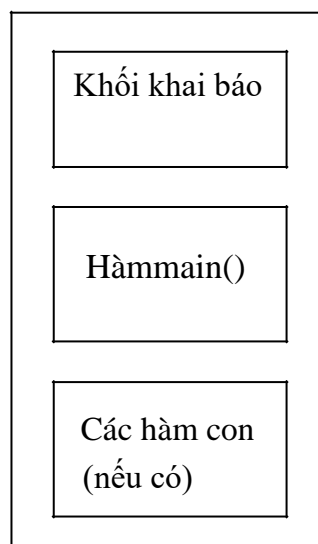
I.2. Ví dụ

```
//Khai báo thư viện hàm #include<conio.h> #include<stdio.h>
#include<string.h> #include<dos.h> #include<process.h>
//Khai báo biến toàn cục và nguyên mẫu hàm void ThayThe(char * S, char
*St );
void Doc1Sector(int vt); void Ghi1Sector(int vt);
//Hàm chính void main()
{
    unsigned char buf[512]; char S[20], St[20];
    printf("Nhap chuoi can tim: "); gets(S);
    printf("Nhap chuoi thay the:"); gets(St) ;
    printf("\nXin cho..."); TimVaThayThe(S,St,buf); printf("\n Thanh
cong."); getch();
}
//Cài đặt các hàm con
void ThayThe(char * S, char *St )
{
    int l=strlen(St); for(int i=0;i<l;i++)

        S[i]=St[i];
}
void Doc1Sector(int vt, char buf[512])
{
    if(absread(0,1,vt,buf))
    {
        printf("\n loi doc dia, nhan enter thoat"); getch();
        exit(1);
    }
```

```
    }  
}  
void Ghi1Sector(int vt, char buf[512])  
{  
    if(abswrite(0,1,vt,buf))  
    {  
        printf("\n loi ghi dia, nhan enter thoat"); getch();  
        exit(1);  
    }  
}  
void TimVaThayThe(char * S, char *St, unsigned char buf[])  
{  
    for(int i=33;i<=500;i++)  
    {  
        Doc1Sector(i, buf); char *  
        p=strstr(buf, S); if(p)  
        {  
            ThayThe(p, St); Ghi1Sector(i,  
            buf);  
        }  
    }  
}
```

I.3. Cấu trúc một chương trình C



a. **Khởi khai báo**

Bao gồm các khai báo về sử dụng thư viện, khai báo hằng số, khai báo hàm con (các nguyên mẫu hàm), khai báo các biến toàn cục và khai báo các kiểu dữ liệu tự định nghĩa.

b. Hàm chính (main())

Chứa các biến, các lệnh và các lời gọi hàm cần thiết trong chương trình.

c. Các hàm con

Được sử dụng nhằm mục đích:

- ☐ Khi có một công việc giống nhau cần thực hiện ở nhiều vị trí.
- ☐ Khi cần chia một chương trình lớn phức tạp thành các đơn thể nhỏ (hàm con) để chương trình được trong sáng, dễ hiểu trong việc xử lý, quản lý việc tính toán và giải quyết vấn đề.

d. Nguyên mẫu hàm

<Kiểu dữ liệu của hàm> Tên hàm ([danh sách các tham số]);

Nguyên mẫu hàm thực chất là dòng đầu của hàm thêm dấu chấm phẩy (;) vào cuối, tuy nhiên tham số trong nguyên mẫu hàm có thể bỏ phần tên.

I.4. Cách xây dựng một hàm con

a. Kiểu dữ liệu của hàm

Xác định dựa vào kết quả của bài toán (Output). Gồm 2 loại :

- ☐ **void**: Hàm không trả về giá trị. Những hàm loại này thường rơi vào những **nhóm chức năng: Nhập / xuất dữ liệu , thống kê, sắp xếp, liệt kê.**

void Tên_hàm (danh sách các tham số)

{

Khai báo các biến cục bộ

Các câu lệnh / khối lệnh hay lời gọi đến hàm khác.

}

□ **Kiểu dữ liệu cơ bản (rời rạc/ liên tục) hay kiểu dữ liệu có cấu trúc:**

Kiểu dữ liệu tùy theo mục đích của hàm cần trả về giá trị gì thông qua việc phân tích bài toán. Những hàm loại này thường được sử dụng trong các trường hợp: **Đếm, kiểm tra, tìm kiếm, tính trung bình, tổng, tích, ...**

<Kiểu dữ liệu> Tên_hàm ([danh sách các tham số])

{

<Kiểu dữ liệu>
kq; Khai báo các
biến cục bộ

Các câu lệnh / khối lệnh hay lời gọi đến hàm khác.

return kq;

}

✂ Đối với những hàm trả về nhiều loại giá trị cho từng trường hợp cụ thể (chẳng hạn như kiểm tra: đúng hay sai, so sánh: bằng, lớn hơn hay nhỏ hơn, ...) thì cần ghi chú rõ giá trị trả về là gì cho từng trường hợp đó.

b. Tham số

Xác định dựa vào dữ liệu đầu vào của bài toán (Input). Gồm 2 loại :

- **Tham số không là con trở (tham trị):** Không thay đổi hoặc không cần lấy giá trị mới của tham số sau lời gọi hàm. Tham số dạng này chỉ mang ý nghĩa là **dữ liệu đầu vào**.
- **Tham số con trở (tham biến):** Có sự thay đổi giá trị của tham số trong quá trình thực hiện và cần lấy lại giá trị đó sau khi ra khỏi hàm. Ứng dụng của tham số loại này có thể là **dữ liệu đầu ra** (kết quả) hoặc cũng có thể **vừa là dữ liệu đầu vào vừa là dữ liệu đầu ra**.

c. Tên hàm

Đặt tên theo **quy ước đặt tên trong C** sao cho tên gọi **đúng với chức năng hay mục đích thực hiện của hàm và gọi nhớ**.

d. Ví dụ

Ví dụ 1: Viết chương trình nhập số nguyên dương n và in ra màn hình các ước số của n

Phân tích bài toán:

- **Input:** n (Để xác định tham số)
 - Kiểu dữ liệu: số nguyên dương (*unsigned int*).
 - Giá trị n không bị thay đổi trong quá trình tìm ước số \Rightarrow Tham số của hàm không là con trở.
- **Output:** In ra các ước số của n (Để xác định kiểu dữ liệu hàm)
 - Không trả về giá trị.
 - Kiểu dữ liệu của hàm là *void*.
- **Xác định tên hàm:** Hàm này dùng in ra các ước số của n nên có thể đặt là

LietKeUocSo

Ta có nguyên mẫu hàm:

void LietKeUocSo (unsigned int n);

```
#include<conio.h>

#include<stdio.h>

//Khai bao nguyen mau ham

void LietKeUocSo ( unsigned int n );

void main()
{
    unsigned int n;

    printf("Nhap
n = ");
    scanf("%u",
&n);

    printf("Cac uoc so cua
n : ");
    LietKeUocSo(n);

    getch( );
}

void LietKeUocSo (unsigned int n)
{

    for(int i=1;
        i<=n;
        i++)
        printf("%
u\t", i);
}
```

✂ Lưu ý cách gọi hàm: Đối với hàm có kiểu dữ liệu hàm là **void** thì khi gọi không cần phải gán giá trị vào biến, ngược lại phải gọi như trong ví dụ 2 (Phải

khai báo **tương ứng kiểu** với kiểu dữ liệu hàm sẽ gọi và gán giá trị trả về vào biến đó).

Ví dụ 2: Viết chương trình nhập số nguyên dương n và tính tổng, với $n > 0$

Phân tích bài toán:

- **Input:** n (Để xác định tham số)
 - Kiểu dữ liệu: số nguyên dương (*unsigned int*).
 - Giá trị n không bị thay đổi trong quá trình tính tổng ⇨ Tham số của hàm không là con trỏ.
- **Output:** Tổng S (Để xác định kiểu dữ liệu hàm)
 - Trả về giá trị của S .
 - S là tổng các số nguyên dương nên S cũng là số nguyên dương
⇨ Kiểu trả về của hàm là unsigned int (hoặc unsigned long cho trường hợp giá trị của tổng lớn hơn 2 bytes).
- **Xác định tên hàm:** Hàm này dùng tính tổng S nên có thể đặt là TongS. Ta có nguyên mẫu hàm:

unsigned long TongS (unsigned int n);

```
#include <conio.h>

#include <stdio.h>

// Khai báo nguyên mẫu hàm

unsigned long TongS ( unsigned int n );

void main()

{

    unsigned
    int n;
    unsigned
    long kq;

    printf("Nhap n = ");
```

```

scanf("%u
",&n); kq
= TongS (
n );

printf("Tong can tinh la: %lu
", kq); getch( );

}

unsigned long TongS (unsigned int n)

{

    unsigned
    long S=0;
    int i=1;
    while(i<=n)

    {

        S+=i;

        i++;

    }

    return S;

}

```

II. BÀI TẬP

II.1. Bài tập cơ bản

1. Cài đặt lại tất cả các bài tập ở chương 2 theo phương pháp hàm.
2. Viết chương trình tính diện tích và chu vi của hình chữ nhật với chiều dài và chiều rộng được nhập từ bàn phím.
3. Viết chương trình tính diện tích và chu vi hình tròn với bán kính được nhập từ bàn phím.
4. Nhập số nguyên dương n ($n > 0$). Liệt kê tất cả các số nguyên tố nhỏ hơn n .
5. Nhập số nguyên dương n ($n > 0$). Liệt kê n số chính phương đầu tiên.

6. Nhập số nguyên dương n ($n > 0$). Đếm xem có bao nhiêu số hoàn thiện nhỏ hơn n .
7. Nhập số nguyên dương n ($0 \leq n < 1000$) và in ra cách đọc của n . Ví dụ: Nhập $n = 105$. In ra màn hình: *Mot tram le nam*.
8. Viết chương trình tính tiền thuê máy dịch vụ Internet và in ra màn hình kết quả. Với dữ liệu nhập vào là giờ bắt đầu thuê (GBD), giờ kết thúc thuê (GKT), số máy thuê (SoMay).
 - Điều kiện cho dữ liệu nhập: $6 \leq \text{GBD} < \text{GKT} \leq 21$. Giờ là số nguyên.
 - Đơn giá: 2500đ cho mỗi giờ máy trước 17:30 và 3000đ cho mỗi giờ máy sau 17:30.
9. Viết chương trình tính tiền lương ngày cho công nhân, cho biết trước giờ vào ca, giờ ra ca của mỗi người.

Giả sử rằng:

- Tiền trả cho mỗi giờ trước 12 giờ là 6000đ và sau 12 giờ là 7500đ.
 - Giờ vào ca sớm nhất là 6 giờ sáng và giờ ra ca trễ nhất là 18 giờ (*Giả sử giờ nhập vào nguyên*).
10. Nhập vào 2 số nguyên p, q và tính biểu thức sau:

$$(-q/2 + (p^3/27 + q^2/4)^{1/2})^{1/3} + (-q/2 - (p^3/27 + q^2/4)^{1/2})^{1/3}$$
 11. Nhập vào 3 số thực a, b, c và kiểm tra xem chúng có thành lập thành 3 cạnh của một tam giác hay không? Nếu có hãy tính diện tích, chiều dài mỗi đường cao của tam giác và in kết quả ra màn hình.
 - Công thức tính diện tích $s = \sqrt{p(p-a)(p-b)(p-c)}$
 - Công thức tính các đường cao: $h_a = 2s/a, h_b = 2s/b, h_c = 2s/c$.

(Với p là nửa chu vi của tam giác).

12. Viết chương trình nhập 2 số nguyên dương a, b. Tìm USCLN và BSCNN của hai số nguyên đó.
13. Viết chương trình tính tổng nghịch đảo của n giai thừa.
14. Cho 2 số nguyên a, b. Viết hàm hoán vị giá trị 2 số trên.
15. (*) Viết chương trình nhập số nguyên dương n gồm 5 chữ số, kiểm tra xem các chữ số n có phải là số đối xứng hay không.

Ví dụ: *Đối xứng:* 13531

Không đối xứng: 13921

16. Viết chương trình nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$), đếm xem n có bao nhiêu chữ số chẵn và bao nhiêu chữ số lẻ.
17. Viết chương trình nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$), đếm xem n có bao nhiêu chữ số là số nguyên tố.
18. Viết chương trình nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$), tính tổng các ước số dương của n.

Ví dụ: *Nhập n=6*

Tổng các ước số từ 1 đến n: $1+2+3+6=12$.

19. Viết chương trình nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$), tìm ước số lẻ lớn nhất của n.

Ví dụ: *Ước số lẻ lớn nhất của 27 là 9.*

21. Viết chương trình nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$), kiểm tra xem các chữ số của n có toàn lẻ hay toàn chẵn không.
22. (*) Viết chương trình nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$), sắp xếp các chữ số của n theo thứ tự tăng dần.

Ví dụ: Nhập $n=1536$

Kết quả sau khi sắp xếp: 1356.

II.2. Bài tập luyện tập và nâng cao

23. Viết chương trình nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$), sau đó nhập một số nguyên x , tìm vị trí xuất hiện của chữ số có giá trị x trong n .

Ví dụ: Nhập $n=1526$, $x=2$

Kết quả: Chu số 2 ở vị trí thứ 3.

24. Viết chương trình nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$), kiểm tra xem các chữ số của n có được sắp thứ tự không.

Ví dụ: Nhập $n=1569$ hoặc

$n=8521$ Kết quả: Có

thứ tự.

25. Viết chương trình nhập 2 số a, b sao cho: số lớn nhất trong 2 số phải là một số dương và chia hết cho 7. Nếu nhập sai phải yêu cầu nhập lại cho đến khi đúng.

26. Viết chương trình nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$), tính giá trị trung bình các chữ số chẵn trong n .

27. (*) Viết chương trình in ra màn hình ngày/tháng/năm của ngày hiện tại, cho phép sử dụng các phím mũi tên lên, xuống để tăng hoặc giảm một ngày.

28. (*) Viết chương trình in ra màn hình giờ:phút:giây hiện tại, cho phép sử dụng các phím mũi tên lên, xuống để tăng hoặc giảm một giây.

III. KẾT LUẬN

- ☐ Trước khi xây dựng một hàm ta phải **xác định mục đích của hàm** là dùng để làm gì, trên cơ sở đó, ta mới xác định được các thành phần của hàm và xây dựng nguyên mẫu hàm.

TM Mỗi hàm phải **thực hiện một chức năng độc lập** và tách biệt với các hàm khác (*không được lồng nhau*).

Đối với hàm có giá trị trả về phải lưu ý kiểu dữ liệu phải tương ứng kiểu dữ liệu cả giá trị trả về và kiểu dữ liệu của biến được gán khi gọi hàm. Trường hợp hàm trả về từ **hai loại giá trị trở lên thì phải có dòng chú thích cho trường hợp tương ứng** để khi gọi hàm biết được kết quả (*chẳng hạn như tìm kiếm, kiểm tra, so sánh, ... giá trị trả về có 2 trường hợp: Có hoặc không có phần tử cần tìm, thỏa điều kiện kiểm tra hay không? Do vậy ta phải quy ước giá trị cho từng trường hợp*).

- ☐ Nên đặt tên hàm sao cho **gọi nhớ** được chức năng, đặt tên **theo quy tắc nhất định để tránh việc gọi sai tên hàm** do lẫn lộn giữa ký tự hoa và thường, có dấu gạch nối giữa các từ trong hàm hay không?
- ☐ Khi gọi hàm phải truyền **đủ tham số, đúng kiểu dữ liệu và đúng thứ tự** của tham số.

BÀI 3 MẢNG VÀ CON TRỎ

Cách khai báo dữ liệu kiểu mảng, các thao tác nhập xuất, các kỹ thuật thao tác trên mảng. Ứng dụng các kỹ thuật này trong việc cài đặt các hàm tìm kiếm, kiểm tra, xây dựng mảng, tách và ghép mảng.

I. TÓM TẮT LÝ THUYẾT

I.1. Khái niệm

Mảng thực chất là một biến được cấp phát bộ nhớ liên tục và bao gồm nhiều biến thành phần.

Các thành phần của mảng là tập hợp các biến có cùng kiểu dữ liệu và cùng tên. Do đó để truy xuất các biến thành phần, ta dùng cơ chế chỉ mục.

I.2. Khai báo mảng

Để khai báo một mảng, ta có 2 cách khai

báo sau : Cách 1: Con trỏ hằng

< Kiểu dữ liệu > < Tên mảng > [< Số phần tử tối đa của mảng >] ;

Ví dụ:

```
int a[100];    // Khai báo mảng số nguyên a gồm 100 phần tử
float b[50];   // Khai báo mảng số thực b gồm 50 phần tử
```

Cách 2: Con trỏ

Ý nghĩa: Khi ta khai báo một mảng với kiểu dữ liệu bất kì (int, float, char,...) thì tên của mảng thực chất là một **hằng địa chỉ của phần tử đầu tiên**.

< Kiểu dữ liệu > * < Tên mảng > ;
--

Ví dụ :

```
int *p;        // khai báo con trỏ p
int b[100];
p = b;         // p trỏ vào phần tử 0 của mảng b
```

Với cách viết như trên thì ta có thể hiểu các cách viết sau là

tương đương $p[i] * (p + i)$ $b[i] * (b+i)$

Lưu ý: Khi sử dụng biến con trỏ để truy xuất mảng, theo cách như trên thì thực chất con trỏ p chỉ chiếm 2 byte bộ nhớ để chứa địa chỉ mà thôi. Để tạo mảng chứa dữ liệu thành phần thì ta phải cấp phát vùng nhớ cho con trỏ p . Dùng hàm : **malloc**, **calloc** trong thư viện **<stdlib.h>** để cấp phát vùng nhớ.

Ví dụ:

+ **Cách 1:** dùng malloc

```
int *px; //Khai báo con trỏ px
```

```
px = (int *) malloc (100); //Cấp phát 100 ô nhớ kiểu int cho con trỏ px
```

+ **Cách 2:** dùng calloc

```
int *p; //khai báo con trỏ p
```

```
p=(int *) calloc (100,sizeof (int)); //cấp phát 10 ô nhớ mỗi ô chiếm 2bytes
```

Sau khi sử dụng xong thì nên giải phóng vùng nhớ bằng hàm free

Ví dụ : free (p) ; // giải phóng vùng nhớ cho con trỏ p.

I.3. Truy xuất phần tử của mảng

Với khái niệm và cách khai báo như trên ta có hình dạng của mảng một chiều

như sau:

Ví dụ : int A[5] // Khai báo mảng A gồm tối đa 5 phần tử nguyên.

Chỉ số 0 1 2 3 4

A[0]	A[1]	A[2]	A[3]	A[4]
------	------	------	------	------

Đối với con trỏ: Lấy địa chỉ của phần tử trong mảng ta dùng

dấu “&” Ví dụ:

int a[7];

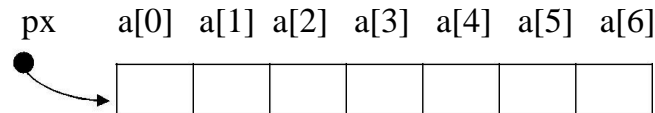
int *p = a[3]; //Lấy địa chỉ phần tử thứ 3

Ví dụ :

int a[7]; int *px;

`px = a; //px trỏ tới phần tử thứ 0`
`px = px + 4; //px trỏ tới phần tử thứ 4`

Từ ví dụ trên ta có thể mô hình hoá mảng như sau:



II. BÀI TẬP

II.1. Một số kĩ thuật cơ bản

a. Kĩ thuật đặt cờ hiệu

Áp dụng: Kĩ thuật này thường được áp dụng cho những bài toán “**kiểm tra**” hay “**đánh dấu**”.

Ví dụ: Viết hàm kiểm tra xem mảng các số nguyên có thứ tự tăng dần không?

(Trả về 1: Nếu mảng tăng dần, ngược lại trả về 0).

```
int KiemTraTang (int a[ ], int n)
{
    int flag = 1;
    for (int i = 0; i < n-1; i++)
        if ( a[i] > a[i+1] ) // Vi phạm điều kiện tăng dần
        {
            flag = 0;
            break;
        }
    return flag;
}
```

b. Kĩ thuật đặt lính canh

Áp dụng: Kĩ thuật này thường được áp dụng cho những bài tập về “tìm kiếm”, “liệt kê” theo một điều kiện nhất định nào đó.

Ví dụ: Viết hàm tìm và trả về giá trị lớn nhất trong mảng một chiều các số nguyên.

```
int TimMax (int a[], int n)
```

```
{  
    int max, i = 1;  
    max = a[0];  
    while ( i < n )  
    {  
        if ( a[i] > max ) max = a[i] ;  
        i++;  
    }  
    return max;  
}
```

II.2. Bài tập cơ bản

II.2.1. Nhập xuất mảng một chiều

Bài tập

1. Viết chương trình nhập xuất mảng một chiều các số thực.
2. Viết chương trình khởi tạo giá trị các phần tử là 0 cho mảng một chiều các số nguyên gồm n phần tử.
3. Viết chương trình phát sinh ngẫu nhiên mảng một chiều các số nguyên âm.
4. Viết chương trình phát sinh ngẫu nhiên mảng một chiều các số nguyên sao cho mảng có thứ tự tăng dần (Không sắp xếp).
5. Viết chương trình nhập mảng các số thực và xuất các phần tử âm trong mảng.
6. Viết chương trình nhập mảng các số nguyên và xuất các phần tử lẻ có trong mảng.
7. Viết chương trình nhập vào mảng một chiều các số nguyên và xuất ra các phần tử chẵn nhỏ hơn 20.
8. Viết chương trình nhập vào mảng một chiều các số nguyên và xuất ra màn hình các phần tử là số nguyên tố.
9. Viết chương trình nhập vào số nguyên n và liệt kê các số nguyên tố nhỏ hơn n, nếu mảng không tồn tại số nguyên tố nào nhỏ hơn n thì phải xuất ra

một câu thông báo.

10. Viết chương trình nhập vào mảng một chiều các số nguyên và xuất ra màn hình các phần tử là số chính phương nằm tại những vị trí lẻ trong mảng.

II.2.2. Tìm kiếm trên mảng một chiều

11. Viết hàm tìm vị trí phần tử có giá trị x xuất hiện cuối cùng trong mảng.
12. Viết hàm tìm vị trí của phần tử nhỏ nhất trong mảng các số nguyên.
13. Viết hàm tìm vị trí của phần tử lớn nhất trong mảng các số nguyên.
14. Viết hàm in vị trí các phần tử nguyên tố trong mảng các số nguyên.
15. Viết hàm in vị trí các phần tử nguyên tố lớn hơn 23.
16. Viết hàm tìm vị trí phần tử âm đầu tiên trong mảng. Nếu không có phần tử âm trả về -1 .
17. Viết hàm tìm vị trí phần tử âm lớn nhất trong mảng.
18. Viết hàm tìm vị trí phần tử dương đầu tiên trong mảng. Nếu không có phần tử âm trả về -1 .
19. Viết hàm tìm vị trí phần tử dương bé nhất trong mảng.
20. Viết hàm in các phần tử là bội của 3 và 5.
21. Viết hàm tìm số chẵn cuối cùng có trong mảng, nếu không tồn tại số chẵn hàm trả về -1 .
22. Viết hàm tìm số lẻ lớn nhất có trong mảng, nếu không tồn tại số lẻ hàm trả về -1 .
23. Viết hàm tìm và đổi chỗ phần tử lớn nhất với phần tử nhỏ nhất trong mảng.
24. Nhập vào X . Viết hàm in ra màn hình những phần tử có giá trị từ 1 đến X có trong mảng.
25. Viết chương trình nhập vào một dãy số a gồm n số thực ($n \leq 100$), nhập vào dãy số b gồm m số thực ($m \leq 100$).
 - In ra những phần tử chỉ xuất hiện trong dãy a mà không xuất hiện trong dãy b .
 - In ra những phần tử xuất hiện ở cả hai dãy.

II.2.3. Đếm tần suất

26. Viết hàm đếm các phần tử âm, dương trong mảng.

27. Viết hàm đếm các phần tử chẵn, lẻ trong mảng.
28. Viết hàm đếm số lần xuất hiện của phần tử x trong mảng.
29. Viết hàm đếm các phần tử nhỏ hơn x trong mảng.
30. Viết hàm đếm các phần tử là số nguyên tố trong mảng.
31. Viết hàm đếm các phần tử là số hoàn thiện trong mảng.
32. Viết hàm đếm các phần tử là bội của 3 và 5 trong mảng các số nguyên.

II.2.4. Tính tổng-trung bình có điều kiện

33. Viết hàm tính tổng các phần tử chẵn trong mảng.
34. Viết hàm tính tổng các phần tử lẻ trong mảng các số nguyên.
35. Viết hàm tính tổng các phần tử nguyên tố trong mảng.
36. Viết hàm tính tổng các phần tử nằm ở vị trí chẵn trong mảng các số nguyên.
37. Viết hàm tính tổng các phần tử nằm ở vị trí nguyên tố trong mảng.
38. Viết hàm tính tổng các phần tử chia hết cho 5 có trong mảng.
39. Viết hàm tính tổng các phần tử cực đại trong mảng các số nguyên
(*phần tử cực đại là phần tử lớn hơn các phần tử xung quanh nó*).

Ví dụ : 1 5 2 6 3 5 1 8 6

40. Viết hàm tính tổng các phần tử cực tiểu trong mảng các số nguyên (*phần tử cực tiểu là phần tử nhỏ hơn các phần tử xung quanh nó*).

Ví dụ : 6 4 2 9 5 3 7 1 5 8

41. Viết hàm tính tổng các phần tử là bội của 3 và 5 trong mảng các số nguyên.
42. Viết hàm tính tổng các phần tử là số hoàn thiện trong mảng các số nguyên.
43. Viết hàm tính giá trị trung bình của các số hoàn thiện trong mảng các số nguyên.

II.2.5. Sắp xếp

44. Viết hàm sắp xếp mảng theo thứ tự giảm dần.
45. Viết hàm sắp xếp mảng theo thứ tự tăng dần của các phần tử là số nguyên tố.
46. Viết hàm sắp xếp các phần tử lẻ tăng dần.
47. Viết hàm sắp xếp các phần tử chẵn giảm dần.

48. Viết hàm sắp xếp các phần tử chẵn nằm bên trái theo thứ tự tăng dần còn các phần tử lẻ bên phải theo thứ tự giảm dần.
49. Viết hàm sắp xếp các phần tử âm giảm dần từ trái sang phải, phần tử dương tăng dần từ phải sang trái.

II.2.6. Xóa mảng

Kỹ thuật cơ bản:

Duyệt mảng từ trái sang phải. Xuất phát từ vị trí cần xóa tiến hành dời lần lượt các phần tử về phía trước cho đến khi kết thúc mảng, sau đó giảm kích thước mảng. Vấn đề đặt ra là tìm vị trí cần xóa theo điều kiện bài toán rồi thực hiện xóa.

50. Viết hàm xóa phần tử tại vị trí lẻ trong mảng.
51. Viết hàm xóa phần tử có giá trị lớn nhất trong mảng.
52. Nhập vào giá trị X. Viết hàm xóa tất cả các phần tử có giá trị nhỏ hơn X.
53. Nhập vào giá trị X. Viết hàm xóa phần tử có giá trị gần X nhất.

II.2.7. Chèn

Kỹ thuật cơ bản:

Duyệt mảng từ phải sang trái. Xuất phát từ cuối mảng tiến hành đẩy lần lượt các phần tử về phía sau cho đến vị trí cần chèn, chèn phần tử cần chèn vào vị trí chèn và tăng kích thước mảng. Trước khi chèn ta phải xác định vị trí cần chèn theo điều kiện bài toán.

54. Viết hàm chèn phần tử có giá trị X vào vị trí đầu tiên của mảng.
55. Viết hàm chèn phần tử có giá trị X vào phía sau phần tử có giá trị lớn nhất trong mảng.
56. Viết hàm chèn phần tử có giá trị X vào trước phần tử có giá trị là số nguyên tố đầu tiên trong mảng.
57. Viết hàm chèn phần tử có giá trị X vào phía sau tất cả các phần tử có giá trị chẵn trong mảng.

II.3. Bài tập luyện tập và nâng cao

60. Viết chương trình nhập vào mảng A gồm n phần tử, trong quá trình nhập kiểm tra các phần tử nhập vào không được trùng, nếu trùng

thông báo và yêu cầu nhập lại.

61. Viết hàm tính tổng của từng dãy con giảm có trong mảng.
62. (*) Cho mảng các số nguyên a gồm n phần tử ($n \leq 30000$) và số dương k ($k \leq n$). Hãy chỉ ra số hạng lớn thứ k của mảng.

Ví dụ: Mảng a: 6 3 1 10 11 18

$k = 2$

Kết quả: 10

63. (*) Cho 2 dãy A, B các số nguyên (*kích thước dãy A nhỏ hơn dãy B*). Hãy kiểm tra xem A có phải là con của B hay không?
64. Viết hàm liệt kê các bộ 4 số a, b, c, d trong mảng các số nguyên (*có ít nhất 4 phần tử và đôi một khác nhau*) sao cho $a + b = c + d$.
65. (*) Viết chương trình tính trung bình cộng của các tổng các dãy tăng dần có trong mảng các số nguyên.

Ví dụ: 1 2 3 4 2 3 4 5 6 4 5 6 $\Rightarrow TB = 15$.

66. Viết chương trình tính tổng tất cả các phần tử xung quanh trên mảng các số nguyên. (Phần tử xung quanh là hai phần tử bên cạnh cộng lại bằng chính nó (Ví dụ: 1 3 2 1,2 là hai phần tử xung quanh của 3).

Ví dụ : 1 3 2 5 3 9 6 *tổng 17*

67. (**) Viết chương trình nhập vào hai số lớn a, b nguyên (a, b có từ 20 chữ số trở lên). Tính tổng, hiệu, tích, thương của hai số trên.
68. Viết hàm tính tổng các phần tử là số Armstrong (số Armstrong là số có đặc điểm như sau: số có k ký số, tổng của các lũy thừa bậc k của các ký số bằng chính số đó.

Ví dụ: 153 là số có các ký số $1^3 + 5^3 + 3^3 = 153$ là một số Armstrong).

69. Viết hàm tìm và xóa tất cả các phần tử trùng với x trong mảng một chiều các số nguyên, nếu không tồn tại phần tử x trong mảng thì trả về -1.

70. Viết hàm xoá tất cả những phần tử trùng nhau trong dãy chỉ giữ lại một phần tử trong đó.

Ví dụ: 1 6 2 3 2 4 2 6 5 1 6 2 3 4 5

71. (**) Viết hàm xoá những phần tử sao cho mảng kết quả có thứ tự tăng dần và số lần xoá là ít nhất.
72. Cho dãy a gồm n số nguyên có thứ tự tăng dần. Nhập vào một phần tử nguyên X , viết hàm chèn X vào dãy sao cho dãy vẫn có thứ tự tăng dần (*không sắp xếp*).
73. Viết chương trình tìm số lẻ nhỏ nhất lớn hơn mọi số chẵn có trong mảng.
74. Viết hàm tìm giá trị chẵn nhỏ nhất nhỏ hơn mọi giá trị lẻ trong mảng các số nguyên.
75. Viết hàm tìm phần tử xuất hiện nhiều nhất trong mảng các số nguyên.
76. Viết chương trình đếm và liệt kê các mảng con tăng dần trong mảng một chiều các số nguyên.
Ví dụ: 6 5 3 2 3 4 2 7 các dãy con tăng dần là 2 3 4 và 2 7
77. Viết chương trình tìm mảng con tăng dần có tổng lớn nhất trong mảng một chiều.
78. (*) Viết chương trình nhập vào một dãy số a gồm n số nguyên ($n \leq 100$). Tìm và in ra dãy con tăng dài nhất
Ví dụ : Nhập dãy a : 1 2 3 6 4 7 8 3 4 5 6 7 8 9 4 5
Dãy con tăng dài nhất : 3 4 5 6 7 8 9
79. (**) Viết chương trình tách 1 mảng các số nguyên thành 2 mảng a và b , sao cho kết quả thu được là:
- Mảng a chứa toàn số lẻ tăng dần.
 - Mảng b chứa toàn số chẵn giảm dần.
- (*Không dùng sắp xếp*)
- Hướng dẫn: Tìm vị trí chèn thích hợp khi trích phần tử từ mảng ban đầu.
Ví dụ: Mảng ban đầu: 9 3 8 2 7 5 1 0 10
Mảng a : 1 3 5 7 9
Mảng b : 10 8 2
80. (**) **Viết** chương trình in ra tam giác Pascal (dùng mảng một chiều).
81. Viết chương trình nhập vào dãy số a gồm n số thực ($n \leq 100$), nhập vào dãy số b gồm m số thực ($m \leq 100$).
- Hãy sắp xếp hai dãy theo thứ tự tăng dần.
 - (*) Trộn 2 dãy trên thành dãy c sao cho dãy c vẫn có thứ tự tăng.
 - Xuất dãy a, b, c ra màn hình.

82. (*) Cho mảng C có n phần tử ($n < 200$), các phần tử là các chữ số trong hệ đếm cơ số 16 (Hexa) (điều kiện mỗi phần tử $\leq n$). Hãy tách mảng C ra các mảng con theo điều kiện sau: các mảng con được giới hạn bởi hai lần xuất hiện thứ hai của con số trong dãy.

Ví dụ: **123A4518B23** có các dãy con là 123A451, 23A4518B2, 23A4518B23

83. (**) Cho hai số nguyên dương A, B. Hãy xác định hai số C, D tạo thành từ hai số A, B sao cho C là số lớn nhất, D là số nhỏ nhất. Khi gạch đi một số chữ số trong C (D), thì các số còn lại giữ nguyên tạo thành A, các chữ số bỏ đi giữ nguyên tạo thành B.

Ví dụ: $A = 52568, B = 462384 \rightarrow C = 54625682384, D = 45256236884$.

84. Viết chương trình nhập vào dãy số a gồm n số nguyên ($n \leq 100$).

- Hãy đảo ngược dãy đó.

Ví dụ: Nhập a: 3 4 5 2 0 4 1

Dãy sau khi đảo: 1 4 0 2 5 4 3

- (*) Hãy kiểm tra xem dãy đã cho có thứ tự chưa (dãy được gọi là thứ tự khi là dãy tăng hoặc dãy giảm).

85. Cho mảng A có n phần tử hãy cho biết mảng này có đối xứng hay không.

86. (**) Hãy viết chương trình phát sinh ngẫu nhiên mảng các số nguyên gồm 10.000 phần tử, mỗi phần tử có giá trị từ 0 đến 32.000 và xây dựng hàm thống kê số lần xuất hiện các phần tử trong mảng, sau đó cho biết phần tử nào xuất hiện nhiều lần nhất.

Ví dụ: Mảng: 5 6 11 4 4 5 4

5 xuất hiện 2 lần

6 xuất hiện 1 lần

11 xuất hiện 1 lần

4 xuất hiện 3 lần

4 xuất hiện nhiều lần nhất

III. KẾT LUẬN

Dữ liệu kiểu mảng dùng cho việc biểu diễn những thông tin có cùng kiểu dữ liệu liên tiếp nhau. Khi cài đặt bài tập mảng một chiều nên xây dựng thành những hàm chuẩn để dùng lại cho các bài tập khác. Các

thao tác trên mảng đều theo quy tắc nhất định, chúng ta có thể ứng dụng mảng trong việc biểu diễn số lớn, dùng bảng tra, khử đệ qui, ...

BÀI 4 CHUỖI KÝ TỰ**I. TÓM TẮT LÝ THUYẾT****I.1. Khái niệm**

Chuỗi ký tự là một dãy các phân tử, mỗi phân tử có kiểu ký tự.

Lưu ý: Chuỗi ký tự được kết thúc bằng ký tự '\0'. Do đó khi khai báo độ dài của chuỗi luôn luôn khai báo dư 1 phân tử để chứa ký tự '\0'.

Ví dụ: `char S[5]="CNTT"` //khai báo chuỗi có 5 phân tử kiểu char và gán dãy ký tự CNTT và chuỗi.

C	N	T	T	\0
Phần tử S[0]	Phần tử S[1]	S[2]	Phần tử S[3]	Phần tử S[4]

Chuỗi rỗng là chuỗi chưa có ký tự nào trong mảng ký hiệu ""

I.2. Khai báo chuỗi

Để khai báo một chuỗi, ta có 2 cách khai báo sau :

Cách 1: Con trỏ hằng

`char < Tên chuỗi > [< Số ký tự tối đa của chuỗi >] ;`

Ví dụ: `char chuoi[25];`

Ý nghĩa khai báo 1 mảng kiểu ký tự tên là chuoi có 25 phân tử (như vậy tối đa ta có thể nhập 24 ký tự vì **phần tử thứ 25 đã chứa ký tự kết thúc chuỗi '\0'**)

Cách 2: Con trỏ

`char *< Tên chuỗi >;`

Ví dụ : `char *chuoi;`

I.3. Các thao tác trên chuỗi**a. Nhập chuỗi**

Cú pháp : `char *gets(char *s);`

Nhận các ký tự nhập từ phím cho đến khi nhấn phím Enter và đưa vào s.

Ví dụ:

```
void main()
{
    char chuoil[80];
    printf("Nhap vao
    chuoil:");
    gets(chuoil);
    printf("Chuoil vua nhap la: %s\n", chuoil);
}
```

b. Xuất chuỗi

Cú pháp : int puts(const char *s);

Xuất chuỗi s ra màn hình.

Ví dụ:

<pre>void main() { char chuoil[] = "Vi du xuat chuoil\n"; puts(string); }</pre>

II. BÀI TẬP**II.1. Bài tập cơ bản**

1. Cho biết kết quả của đoạn chương trình sau:

```
char input[20]="Truong cao dang CNTT", *p,
*temp; strcpy(temp, input);
do
{
    p=
    strtok(temp, "");
    printf("%s\n"
    ,p);
    p = strtok(NULL,
    ""); strcpy(temp, p);
}while(p!=NULL);
printf("Chuoi temp: %s \n Chuoi input: %s", temp, input);
```

2. Cho biết kết quả của đoạn chương trình sau:

```
char s1[20]="Truong cao dang CNTT", s1[10]="Tp. HCM", *input, *s3;
strcpy(input, s1); strcpy(s3,"aeiou"); strcat(input, s2);
int n=strlen(input), k=0;
printf("Chuoi:
%s",input); for(int i=0;
i<n; i++)
{
    if(strchr(s3,
        input[i])) k++;
}
printf("\nKet qua: %d", k);
```

3. Viết chương trình nhập vào một chuỗi ký tự, đếm số ký tự có trong chuỗi.
4. Viết chương trình đếm có bao nhiêu khoảng trắng trong chuỗi.
5. Viết chương trình nhập vào một chuỗi, hãy loại bỏ những khoảng trắng thừa trong chuỗi.
6. Viết chương trình nhập vào hai chuỗi s1 và s2, nối chuỗi s2 vào s1. Xuất chuỗi s1 ra màn hình.
7. Đổi tất cả các ký tự có trong chuỗi thành chữ thường (không dùng hàm `strlwr`).
8. Đổi tất cả các ký tự trong chuỗi sang chữ in hoa (không dùng hàm `struppr`).
9. Viết chương trình đổi những ký tự đầu tiên của mỗi từ thành chữ in hoa.
10. Viết chương trình đổi chữ xen kẽ 1 chữ hoa và 1 chữ thường.
Ví dụ: nhập ABCDEfgh đổi thành AbCdEfGh
11. Viết chương trình đảo ngược các ký tự trong chuỗi .
Ví dụ: nhập ABCDE, xuất ra màn hình là:EDCBA
12. Viết chương trình tìm kiếm 1 ký tự xem có trong chuỗi hay không, nếu có xuất ra vị trí của từ đó.
13. Viết 1 chương trình đếm một ký tự xuất hiện bao nhiêu lần trong chuỗi.
14. Viết chương trình tìm kiếm tên trong chuỗi họ tên. Nếu có thì xuất ra là tên này đã nhập đúng, ngược lại thông báo là đã nhập sai.

15. Viết chương đảo vị trí của từ đầu và từ cuối.
Ví dụ: nhập "bo an co" xuất ra "co an bo"
16. Viết hàm cắt chuỗi họ tên thành chuỗi họ lót và chuỗi tên.
Ví dụ: chuỗi họ tên là: "Nguyễn Văn A" cắt ra 2 chuỗi là chuỗi họ lót: "Nguyễn Văn", chuỗi tên là: "A"
17. Nhập một chuỗi bất kỳ, sau đó hỏi người dùng cần tách bắt đầu từ đâu trong chuỗi trở về sau.
Ví dụ: Nhập chuỗi S1: "trường Cao Đẳng Công Nghệ Thông tin". Người nhập muốn tách bắt đầu từ chữ "Công" thì sẽ xuất ra chuỗi "Công Nghệ Thông Tin" ra màn hình.
18. Viết hàm kiểm tra xem chuỗi có đối xứng hay không?.
19. Viết hàm tra xem trong chuỗi có ký tự số hay không nếu có tách ra thành một mảng số riêng.
20. Nhập một chuỗi bất kỳ, yêu cầu nhập 1 ký tự muốn xóa. Thực hiện xóa tất cả những ký tự đó trong chuỗi.
21. Viết chương trình tìm kiếm xem ký tự nào xuất hiện nhiều nhất trong chuỗi.
22. Viết 1 chương trình xoá một từ nào đó trong chuỗi.
Ví dụ: Chuỗi ban đầu: "CAO DANG CNTT"
Nhập: "CNTT", và kết quả xuất ra: "CAO DANG"

II.2. Bài tập luyện tập và nâng cao

23. Đổi các từ ở đầu câu sang chữ hoa và những từ không phải đầu câu sang chữ thường.
Ví dụ: nGuYen vAN a đổi thành: Nguyễn Văn A
24. (*) Viết chương trình đảo ngược thứ tự các từ có trong chuỗi
Ví dụ: Nhập Truong CD CNTT TpHCM
Xuất ra màn hình là: TpHCM CNTT CD Truong
25. Nhập 1 chuỗi bất kỳ, liệt kê xem mỗi ký tự xuất hiện mấy lần.
26. Viết hàm kiểm tra xem trong 2 chuỗi có bao nhiêu ký tự giống nhau.
27. Viết chương trình mình chạy từ trái qua phải màn hình.
28. Viết 1 chương trình chèn 1 từ ở bất cứ vị trí nào mà người dùng yêu

cầu.

29. (*) Viết chương trình nhập vào một chuỗi đếm xem chuỗi có bao nhiêu từ. Các từ cách nhau bằng khoảng trắng, dấu chấm câu: dấu chấm (.), dấu phẩy (,), dấu chấm phẩy (;), dấu hỏi (?) và dấu chấm than (!).
30. (**) Viết chương trình hiển thị một chuỗi ký tự. Chương trình cho phép di chuyển dấu nháy sang trái, sang phải, lên dòng hay xuống dòng bằng phím mũi tên, chèn hay xoá ký tự tại vị trí dấu nháy.

III. KẾT LUẬN

Cũng giống như kiểu mảng một chiều, thao tác truy xuất các phần tử trên chuỗi hoàn toàn tương tự. Bên cạnh đó, kiểu dữ liệu này còn được cài đặt sẵn một số hàm thư viện rất hữu ích nên trong quá trình thao tác trên chuỗi nên khi cài đặt ta cố gắng **tận dụng tối đa những hàm liên quan. Không nên sử dụng hàm scanf()** để nhập chuỗi trong trường hợp chuỗi dữ liệu nhập vào có chứa khoảng trắng. Nếu **nhập chuỗi phía sau hàm scanf()** nên chèn hàm **fflush(stdin)** hoặc hàm **flushall()** giữa **scanf** và **gets()** để xóa vùng đệm, tránh trường hợp chương trình **bỏ qua hàm gets()** do trong vùng đệm còn lưu ký tự xuống dòng của phím **ENTER**. Khi thao tác trên chuỗi lưu ý phải **đảm bảo chuỗi được kết thúc bằng ký tự kết thúc '\0'**.

BÀI 5 KIỂU DỮ LIỆU CÓ CẤU TRÚC

Cung cấp cơ chế cho phép khai báo các kiểu dữ liệu mới để giải quyết theo yêu cầu của bài toán dựa vào những kiểu dữ liệu cơ bản được cài đặt sẵn trong ngôn ngữ lập trình.

I. TÓM TẮT LÝ THUYẾT

I.1. Khái niệm

Cấu trúc (struct) thực chất là một kiểu dữ liệu do người dùng định nghĩa bằng cách gom nhóm các kiểu dữ liệu cơ bản có sẵn trong C thành một kiểu dữ liệu phức hợp nhiều thành phần.

I.2. Định nghĩa kiểu dữ liệu

Cú pháp

```
struct < tên cấu trúc >
{
    Các kiểu dữ liệu thành phần ;
};
```

Ngoài ra ta có thể dùng từ khoá **typedef** để định nghĩa một tên mới cho kiểu dữ liệu đã có.

Cú pháp

```
typedef struct < tên cấu trúc > < tên mới >;
```

Ví dụ1: Kiểu dữ liệu DATE gồm các thành phần:

- *Thứ (thu): chuỗi có tối đa 4 ký tự.*
- *Ngày (ngay): số nguyên 1 byte.*
- *Tháng (thang): số nguyên 1 byte.*
- *Năm (nam): số nguyên 2*

bytes. Ta định nghĩa DATE

như sau:

```
struct DATE
```

```
{
```

```
    char thu[5];
```

```
    unsigned char ngay;  
  
    unsigned char thang;  
  
    int nam;  
  
};
```

Kiểu dữ liệu có cấu trúc có thể lồng vào nhau.

Ví dụ 2: Định nghĩa kiểu dữ liệu của học sinh HOCSINH gồm:

- Mã số học sinh (MSHS): chuỗi có tối đa 5 ký tự.
- Họ tên (hoten): chuỗi có tối đa 30 ký tự.
- Ngày tháng năm sinh (ngaysinh): kiểu DATE.
- Địa chỉ (diachi): chuỗi có tối đa 50 ký tự.
- Giới tính (phai): chuỗi có tối đa 3 ký tự.
- Điểm trung bình (diemtb): số thực.

Ta định nghĩa kiểu HOCSINH như sau:

```
struct DATE  
{  
  
    char thu[5];  
  
    unsigned char ngay;  
  
    unsigned char thang;  
  
    int nam;  
  
};  
  
typedef struct HOCSINH  
{  
  
    char MSHS[6];  
  
    char hoten[31];  
  
    struct DATE ngaysinh;  
  
    char diachi[51];  
  
    unsigned char phai[4];
```

```
float diemtb;  
  
};
```

Chú ý: Khi định nghĩa kiểu dữ liệu struct lồng nhau, ta cần lưu ý: Kiểu dữ liệu được sử dụng phải khai báo phía trên.

I.3. Khai báo

Khi ta định nghĩa kiểu dữ liệu tức là ta có một kiểu dữ liệu mới, muốn sử dụng ta phải khai báo biến. Cú pháp khai báo kiểu dữ liệu cũng giống như cách khai báo của các kiểu dữ liệu chuẩn.

struct < tên cấu trúc > < tên biến > ;

Ví dụ :

struct DATE x ; // Khai bao bien x co kieu du lieu DATE

Tuy nhiên nếu ta định nghĩa struct có dùng từ khoá **typedef** thì ta có thể khai báo trực tiếp mà không cần từ khoá “**struct**”.

Ví dụ :

DATE x ; // Khai bao bien x co kieu DATE

***Biến con trỏ kiểu cấu trúc:** Ngoài cách khai báo như trên ta có thể khai báo theo kiểu con trỏ như sau

struct < tên cấu trúc > * < tên biến > ;

Để sử dụng ta cũng phải cấp phát vùng nhớ giống như kiểu dữ liệu chuẩn.

Ví dụ :

*DATE *y; // Khai bao con tro y kieu cau truc DATE y*

*= (DATE *) malloc (sizeof (DATE)) ;*

I.4. Truy xuất

Để truy xuất một thành phần dữ liệu nào đó bên trong cấu trúc ta có 2 trường hợp truy xuất như sau :

- Biến x là một biến cấu trúc thông thường, ta dùng toán tử dấu chấm “.”

Cú pháp :

< Tên cấu trúc > . < Biến thành phần > ;

Ví dụ :

DATE x ; // khai bao bien x kieu

DATE x.ngay = 5 ; // gan ngay

bang 5

- Biến x là một biến con trỏ, ta dùng toán tử mũi tên “->” (Gồm dấu trừ ‘-’ và dấu lớn hơn ‘>’).

Cú pháp :

< Tên cấu trúc > -> < Biến thành phần >;
--

<i>Đối với kiểu dữ liệu có struct lồng nhau phải truy cập đến thành phần cuối cùng có kiểu dữ liệu cơ bản.</i>
--

Ví dụ: Giả sử, có kiểu HOCSINH như trên

HOCSINH hs; // khai bao bien hs kieu HOCSINH

*Muốn in học sinh A sinh vào tháng mấy ta phải truy cập như sau:
printf(“Thang sinh cua hoc sinh A la: %d”,(hs.ngaysinh).thang);*

I.5. Mảng cấu trúc

- Cách khai báo tương tự như mảng một chiều hay ma trận (Kiểu dữ liệu bây giờ là kiểu dữ liệu có cấu trúc).
- Cách truy cập phần tử trong mảng cũng như truy cập trên mảng một chiều hay ma trận. Nhưng do từng phần tử có kiểu cấu trúc nên phải chỉ định rõ cần lấy thành phần nào, tức là phải truy cập đến thành phần cuối cùng có kiểu là dữ liệu cơ bản (xem lại bảng các kiểu dữ liệu cơ bản) .

I.6. Nguyên tắc viết chương trình có mảng cấu trúc

Do kiểu dữ liệu có cấu trúc thường chứa rất nhiều thành phần nên khi viết chương trình loại này ta cần lưu ý:

- Xây dựng hàm xử lý cho một kiểu cấu trúc.
- Muốn xử lý cho mảng cấu trúc, ta gọi lại hàm xử lý cho một kiểu cấu trúc đã được xây dựng bằng cách dùng vòng lặp.

II. BÀI TẬP

II.1. Bài tập cơ bản

1. Viết chương trình sử dụng con trỏ cấu trúc để hiển thị giờ, phút, giây ra màn hình, và tính khoảng cách giữa 2 mốc thời gian.
2. Viết chương trình sử dụng con trỏ cấu trúc thể hiện ngày, tháng, năm ra màn hình, và tính khoảng cách giữa 2 ngày.
3. Viết chương trình khai báo kiểu dữ liệu thể hiện một số phức. Sử dụng kiểu này để viết hàm tính tổng, hiệu, tích của hai số phức.
4. Viết chương trình khai báo kiểu dữ liệu để biểu diễn một phân số. Hãy viết hàm thực hiện những công việc sau:
 - Tính tổng, hiệu, tích, thương hai phân số.
 - Rút gọn phân số.
 - Quy đồng hai phân số.
 - So sánh hai phân số.
5. Viết chương trình khai báo kiểu dữ liệu để biểu diễn một hỗn số. Hãy viết hàm thực hiện những công việc sau :
 - Đổi hỗn số sang phân số
 - Tính tổng, tích hai hỗn số
6. Viết chương trình khai báo kiểu dữ liệu để biểu diễn một điểm trong hệ tọa độ 0xy . Hãy viết hàm thực hiện các công việc sau:
 - Tìm những điểm đối xứng của nó qua tung độ, hoành độ, toạ độ tâm.
 - Hãy tính tổng, hiệu, tích của hai điểm trong mặt phẳng toạ độ 0xy.
 - Tính khoảng cách giữa hai điểm.
7. Cho một hình trụ có các thông tin sau: BanKinh (bán kính hình trụ kiểu số thực), ChieuCao (chiều cao hình trụ kiểu số thực). Hãy thực hiện các công việc sau.
 - Nhập dữ liệu cho hình trụ trên.
 - Tính diện tích xung quanh, diện tích toàn phần, thể tích hình trụ.

II.2. Bài Tập Luyện Tập

8. Viết chương trình tạo một mảng các số phức. Hãy viết hàm tính tổng, tích các số phức có trong mảng.

9. Viết chương trình tạo một mảng các phân số. Hãy viết hàm thực hiện các công việc sau :
- Tính tổng tất cả các phân số (kết quả dưới dạng phân số tối giản)
 - Tìm phân số lớn nhất, phân số nhỏ nhất.
 - Sắp xếp mảng tăng dần.
10. Viết chương trình khai báo kiểu dữ liệu STACK (cơ chế LIFO). Viết hàm làm những công việc sau :
- Kiểm tra STACK rỗng
 - Kiểm tra STACK đầy
 - Thêm phần tử vào STACK
 - Lấy phần tử ra khỏi STACK
11. Tổ chức dữ liệu để quản lý sinh viên bằng cấu trúc mẫu tin trong một mảng N phần tử, mỗi phần tử có cấu trúc như sau:
- *Mã sinh viên.*
 - *Tên.*
 - *Năm sinh.*
 - *Điểm toán, lý, hoá, điểm trung bình.*

Viết chương trình thực hiện những công việc sau:

- Nhập danh sách các sinh viên cho một lớp học.
 - Xuất danh sách sinh viên ra màn hình.
 - Tìm sinh viên có điểm trung bình cao nhất.
 - Sắp xếp danh sách lớp theo thứ tự tăng dần của điểm trung bình.
 - Sắp xếp danh sách lớp theo thứ tự giảm dần của điểm toán.
- Tìm kiếm và in ra các sinh viên có điểm trung bình lớn hơn 5 và không có môn nào dưới 3.
- Tìm sinh viên có tuổi lớn nhất.
 - Nhập vào tên của một sinh viên. Tìm và in ra các thông tin liên quan đến sinh viên đó (nếu có).
12. Tổ chức dữ liệu quản lý danh mục các bộ phim VIDEO, các thông tin liên quan đến bộ phim này như sau:
- *Tên phim (tựa phim).*

- *Thể loại (3 loại : hình sự, tình cảm, hài).*
- *Tên đạo diễn.*
- *Tên diễn viên nam chính.*
- *Tên diễn viên nữ chính.*
- *Năm sản xuất.*
- *Hãng sản xuất*

Viết chương trình thực hiện những công việc sau :

- Nhập vào bộ phim mới cùng với các thông tin liên quan đến bộ phim này.
 - Nhập một thể loại: In ra danh sách các bộ phim thuộc thể loại này.
 - Nhập một tên nam diễn viên. In ra các bộ phim có diễn viên này đóng.
 - Nhập tên đạo diễn. In ra danh sách các bộ phim do đạo diễn này dàn dựng.
13. Một thư viện cần quản lý thông tin về các đầu sách. Mỗi đầu sách bao gồm các thông tin sau : MaSSach (mã số sách), TenSach (tên sách), TacGia (tác giả), SL (số lượng các cuốn sách của đầu sách). Viết chương trình thực hiện các chức năng sau:
- Nhập vào một danh sách các đầu sách (tối đa là 100 đầu sách)
 - Nhập vào tên của quyển sách. In ra thông tin đầy đủ về các sách có tên đó, nếu không có thì tên của quyển sách đó thì báo là :Không Tìm Thấy.
 - Tính tổng số sách có trong thư viện.
14. Viết chương trình tạo một mảng danh sách các máy tính của một cửa hàng, thông tin của một máy tính bao gồm :
- *Loại máy*
 - *Nơi sản xuất*
 - *Thời gian bảo hành*
 - Viết hàm nhập một dãy các loại máy tính có thông tin như trên.
 - Hãy viết hàm thống kê xem có bao nhiêu máy có thời gian bảo hành là 1 năm.
 - In ra danh sách các máy tính có xuất xứ từ Mỹ.

15. Để lắp ráp một máy vi tính hoàn chỉnh cần phải có tối thiểu 10 linh kiện loại

A và có thể lắp bổ sung thêm vào khoảng tối đa 8 linh kiện loại B. Tại một cửa hàng vì tính cần quản lý bán hàng các loại linh kiện tại cửa hàng. Thông tin về một loại linh kiện gồm có: Tên linh kiện, quy cách, loại, đơn giá loại 1 (chất lượng tốt – số nguyên), đơn giá loại 2 (chất lượng thường – số nguyên). Viết chương trình thực hiện những công việc sau :

- Nhập vào thông tin về các linh kiện có ở cửa hàng.
- Xuất danh sách các linh kiện đã nhập theo thứ tự tăng dần của loại linh kiện và tên linh kiện.
- Cho biết đã có đủ 10 linh kiện loại A cần thiết lắp ráp máy hay chưa?

16. Một cửa hàng cần quản lý các mặt hàng, thông tin một mặt hàng bao gồm:

- *Mã hàng.*
- *Tên mặt hàng.*
- *Số lượng.*
- *Đơn giá.*
- *Số lượng tồn.*
- *Thời gian bảo hành (tính theo đơn vị tháng).*
- Hãy nhập vào một danh sách các mặt hàng.
- Tìm mặt hàng có số lượng tồn nhiều nhất.
- Tìm mặt hàng có số lượng tồn ít nhất.
- Tìm mặt hàng có giá tiền cao nhất.
- In ra những mặt hàng có thời gian bảo hành lớn hơn 12 tháng.
- Sắp xếp các mặt hàng theo thứ tự tăng dần của số lượng tồn.

17. Viết chương trình quản lý hồ sơ nhân viên trong một công ty, chương trình thực hiện những công việc sau :

- *Họ và tên.*
- *Phái.*
- *Ngày sinh.*
- *Địa chỉ.*
- *Lương cơ bản.*
- *Bảo hiểm xã hội.*

- *Thưởng.*
- *Phạt.*
- $Lương\ thực\ lĩnh = lương\ cơ\ bản + thưởng - BH\ xã\ hội - phạt.$
- Nhập vào hồ sơ của các nhân viên trong công ty.
- Xuất danh sách các nhân viên theo lương thực lĩnh giảm dần bằng 2 cách sau :
 - *Cấp phát vùng nhớ tĩnh.*
 - *Cấp phát vùng nhớ động.*

18.(*) Viết chương trình quản lý lớp học của một trường. Các thông tin của một lớp học như sau :

- *Tên lớp.*
- *Sĩ số.*
- *Danh sách các sinh viên trong lớp.*
- Nhập vào danh sách các lớp với thông tin yêu cầu như trên.
- In danh sách các lớp có trên 5 sinh viên có điểm trung bình loại giỏi.
- Tìm lớp có nhiều sinh viên nhất.
- Tìm lớp có ít sinh viên nhất.
- Tìm sinh viên có điểm trung bình cao nhất.
- Tìm lớp có số lượng sinh viên đạt điểm trung bình loại giỏi nhiều nhất.

19. Viết chương trình quản lý vé tàu, thông tin một vé tàu như sau :

- *Ngày giờ khởi hành, ngày giờ đến.*
- *Ga đi, ga đến.*
- *Loại tàu, loại chỗ ngồi (ngồi, nằm, cứng, mềm).*
- *Số toa, số ghế.*
- Viết hàm nhập vào danh sách các vé tàu.
- In danh sách các vé tàu có ga đến là Huế.
- In danh sách các vé tàu có ga đến là Hà Nội và đi ngày 8/6/2005.
- Đếm xem có bao nhiêu khách đi tàu loại chỗ ngồi là *nằm cứng*.
 - Viết chương trình tính tiền điện hàng tháng của các hộ gia

đình, thông tin các khách hàng như sau :

Kỳ thu, từ ngày.....đến ngày.

Tên khách hàng, mã khách hàng.

Địa chỉ.

Điện năng tiêu thụ (Kwh).

Nhập vào danh sách các khách hàng.

Xuất danh sách hoá đơn theo thứ tự tăng dần của điện năng tiêu thụ.

Tính tiền điện của các khách hàng theo quy định sau.

100 kw đầu tiên là 550 đ / kw

50 kw tiếp theo là 900 đ / kw

50 kw tiếp theo là 1210 đ / kw

Thuế 10 % trên tổng số tiền phải trả

Tính tổng số tiền thu được của các khách hàng.

III. KẾT LUẬN

Kiểu dữ liệu có cấu trúc cho phép ta định nghĩa những kiểu dữ liệu bất kỳ trên cơ sở là những kiểu dữ liệu cơ bản có sẵn trong ngôn ngữ lập trình. Khi xây dựng xong kiểu dữ liệu mới ta phải định nghĩa những thao tác cho kiểu dữ liệu đó. Những kiểu dữ liệu tự định nghĩa này thông thường có rất nhiều thành phần, mỗi thành phần cũng có thể là một kiểu dữ liệu tự định nghĩa, vấn đề là ta chọn kiểu dữ liệu cơ bản nào để xây dựng nên chúng sao cho phù hợp về mặt kiểu dữ liệu và phù hợp về kích thước lưu trữ (vừa đủ). Các sử dụng những kiểu dữ liệu tự định nghĩa cũng giống như các kiểu dữ liệu cơ bản. Muốn sử dụng phải khai báo biến, khi truy cập các thành phần phải truy cập theo quy ước. Nếu thành phần cấu trúc có kiểu dữ liệu là số thực thì khi sử dụng hàm scanf() phải thông qua biến trung gian rồi gán lại cho thành phần cấu trúc đó. Đối với mảng các kiểu dữ liệu có cấu trúc ta nên xử lý cho từng thành phần cấu trúc rồi mới xử lý cho mảng cấu trúc bằng cách dùng vòng lặp.

BÀI 6 TẬP TIN**I. TÓM TẮT LÝ THUYẾT****I.1. Khái niệm**

Trong các chương trình trước thì các dữ liệu đưa vào chương trình chỉ được tồn tại trong RAM, khi thoát chương trình thì tất cả dữ liệu đều bị mất. Để khắc phục tình trạng này Borland C cung cấp cho ta các hàm để lưu trữ và truy xuất tập tin, đó là kiểu **FILE**. Và ở đây ta chỉ đề cập đến 2 loại tập tin :

- Tập tin văn bản: là tập tin dùng để ghi các ký tự lên đĩa theo các dòng.
- Tập tin nhị phân: là tập tin dùng để ghi các cấu trúc dạng nhị phân (được mã hoá).

I.2. Thao tác với tập tin

Quá trình thao tác trên tập tin thông qua 4 bước:

Bước 1: Khai báo con trỏ trỏ đến tập tin.

Bước 2: Mở tập tin.

Bước 3: Các xử lý trên tập tin.

Bước 4: Đóng tập tin.

a. Khai báo

FILE *< tên biến >;

Ví dụ : **FILE *f; // Khai báo biến con trỏ file f**

b. Mở tập tin

fopen (< đường dẫn tên tập tin> , < kiểu truy nhập >);

Ví dụ : **FILE *f; // Khai báo biến con trỏ f**

= fopen ("C:\\VD1.txt" ,

"rt") ;

Các kiểu truy nhập tập tin thông dụng:

t là kiểu truy nhập tập tin đối với dạng tập tin văn bản (text).

b là kiểu truy nhập tập tin đối với dạng tập tin nhị phân (binary). **r** mở ra để đọc (ready only).

w mở ra để ghi (create / write). **a** mở ra để thêm vào (append). **r+** mở ra để đọc và ghi (modify).

c. Các hàm đọc ghi nội dung tập tin

• Tập tin văn bản

STT	TÊN HÀM	Ý NGHĨA SỬ DỤNG	VÍ DỤ
ĐỌC TẬP TIN			
1	<code>fscanf(<FILE *>, <định dạng>, <các tham biến>);</code>	Dữ liệu từ một tập tin theo định dạng.	<code>fscanf(f, "%d", &x);</code>
2	<code>fgets(<vùng nhớ>, <kích thước tối đa>, <FILE *>);</code>	Đọc một chuỗi ký tự từ một tập tin với kích thước tối đa cho phép, hoặc gặp ký tự xuống dòng.	<code>char s[80]; fgets(s, 80, f);</code>
3	<code>getc(< FILE * >);</code>	Đọc một ký tự từ tập tin đang mở.	<code>char c = getc(f);</code>
GHI TẬP TIN			
1	<code>fprintf(<FILE *>, <định dạng>[, <các tham biến>]);</code>	Ghi dữ liệu theo một định dạng nào đó vào tập tin.	<code>fprintf(f, "%d", x);</code>
2	<code>fputs(<chuỗi ký tự>, <FILE *>);</code>	Ghi một chuỗi ký tự vào tập tin đang mở.	<code>fputs("Giao trình BT", f);</code>

• Tập tin nhị phân

STT	TÊN HÀM	Ý NGHĨA SỬ DỤNG	VÍ DỤ
ĐỌC TẬP TIN			
1	<code>fread(<&ptr>, <size>, <len>, <FILE *>);</code>	<ul style="list-style-type: none"> ptr: vùng nhớ để lưu dữ liệu đọc. size: kích thước mỗi ô nhớ (tính bằng byte). len: độ dài dữ liệu cần đọc. FILE: đọc từ tập tin nhị phân nào.	<code>int a[30], b, n; fread(a, sizeof(int), n, f); Fread(&b, sizeof(int), 1, f);</code>
GHI TẬP TIN			
1	<code>fwrite(<&prt>, <size>, <len>, <FILE *>);</code>	Tham số tương tự như hàm fread.	<code>fwrite(a, sizeof(int), n, f);</code>

d. Đóng tập tin

Sau khi không còn làm việc với tập tin, để đảm bảo an toàn cho dữ liệu thì nhất thiết ta phải đóng tập tin lại.

fclose (< biến con trỏ tập tin >) ; hoặc fcloseall () ;
--

Ví dụ :fclose (f) ;

e. Các thao tác khác trên tập tin*** Xoá tập tin :**

remove (< đường dẫn tập tin >);
--

*** Đổi tên tập tin :**

rename (< tên tập tin cũ >, < tên tập tin mới >);
--

*** Di chuyển con trỏ tập tin :**

fseek (< FILE * >, < độ dời >, < mốc >);

II. BÀI TẬP**II.1. Bài tập cơ bản**

1. Viết chương trình tạo tập tin văn bản chứa 1 dãy số nguyên bất kỳ.
2. Viết chương trình tạo tập tin nhị phân chứa 10000 số nguyên bất kỳ ghi vào file SONGUYEN.INP. Mỗi dòng 10 số, sau đó viết chương trình đọc file SONGUYEN.INP, sắp xếp theo thứ tự tăng dần và lưu kết quả vào file SONGUYEN.OUT.
3. Viết chương trình tạo một file chứa 10000 số nguyên ngẫu nhiên đôi một khác nhau trong phạm vi từ 1 đến 32767 và đặt tên là “SONGUYEN.INP”.
4. Viết chương trình tạo một file chứa các số nguyên có tên SONGUYEN.INP. Sau đó đọc file SONGUYEN.INP và ghi các số chẵn vào file SOCHAN.OUT và những số lẻ vào file SOLE.OUT.
5. Viết chương trình ghi vào tập tin SOCHAN.DAT các số nguyên chẵn từ 0 đến 100.
6. Viết chương trình đọc tập tin SOCHAN.DAT và xuất ra màn hình, mỗi dòng 30 số.
7. Viết chương trình giả lập lệnh **COPY CON** để tạo tập tin văn bản. Khi

- kết
thúc tập tin nhấn phím **F6** để lưu.
8. Viết chương trình giả lập lệnh **TYPE** để in nội dung của tập tin văn bản ra màn hình.
 9. Viết chương trình kiểm tra một tập tin nào đó có trong một thư mục được chỉ định hay không?
 10. Viết chương trình giả lập lệnh **DEL** để xóa tập tin. Yêu cầu nhập đường dẫn và tên tập tin, kiểm tra sự tồn tại của tập tin, nếu có thì xóa tập tin được chỉ định.
 11. Viết chương trình giả lập lệnh **RENAME** để đổi tên một tập tin.
 12. Viết chương trình tạo file văn bản có tên là “MATRIX.INP” có cấu trúc như sau:
 - Dòng đầu ghi hai số m, n .
 - Trong m dòng tiếp theo mỗi dòng ghi n số và các số cách nhau một khoảng cách.

Hãy kiểm tra xem trong file đó có bao nhiêu số nguyên tố.
 Kết quả cần ghi vào file “MATRIX.OUT” có nội dung là một số nguyên đó là số lượng các số nguyên tố trong file “MATRIX.INP”.
 13. Cho số nguyên n , hãy in tam giác PASCAL gồm n dòng
Dữ liệu vào: tập tin văn bản PAS.INP gồm 1 dòng chứa giá trị n .
Kết quả: đưa ra tập tin văn bản PAS.OUT thể hiện một tam giác PASCAL n dòng.
 14. Cho mảng các số nguyên, hãy sắp xếp mảng theo thứ tự tăng dần. Dữ liệu vào: tập tin văn bản ARRAY.INP gồm 2 dòng
 - Dòng 1 chứa số nguyên n ($n \leq 100$).
 - Dòng 2 chứa n số nguyên.

Kết quả: Đưa ra tập tin văn bản ARRAY.OUT gồm hai dòng

 - Dòng 1 chứa n phần tử của mảng các số nguyên.
 - Dòng 2 chứa n số nguyên được xếp tăng dần.
 15. Cho mảng các số nguyên, tìm phần tử lớn nhất của mảng.
Dữ liệu vào: tập tin văn bản ARRAY.INP gồm hai dòng:
 - Dòng 1 chứa số nguyên n ($n \leq 100$).

- Dòng 2 chứa n số nguyên.

Kết quả: Đưa ra tập tin văn bản ARRAY.OUT gồm 1 dòng ghi 2 giá trị x, y trong đó x là giá trị lớn nhất, y là vị trí của x trong mảng

II.2. Bài tập luyện tập và nâng cao

16. Cho mảng các số nguyên, tính tổng các phần tử của mảng. Dữ liệu vào : tập tin văn bản ARRAY.INP gồm hai dòng

- Dòng 1 chứa số nguyên n ($n \leq 10$)
- Dòng 2 chứa n số nguyên

Kết quả : Đưa ra tập tin văn bản ARRAY.OUT gồm một dòng ghi tổng các phần tử trong mảng.

17. Cho mảng các số nguyên, hãy liệt kê các phần tử là số nguyên tố Dữ liệu vào : tập tin văn bản NT.INP gồm hai dòng

- Dòng 1 chứa số nguyên n ($n \leq 100$)
- Dòng 2 chứa n số nguyên

Kết quả : đưa ra tập tin văn bản NT.OUT gồm hai dòng:

- Dòng 1 chứa số lượng các phần tử nguyên tố trong mảng.
- Dòng 2 liệt kê các số nguyên tố đó.

18. (*) Tạo file văn bản có tên là "INPUT.TXT" có cấu trúc như sau:

- Dòng đầu tiên ghi N (N là số nguyên dương nhập từ bàn phím).
- Trong các dòng tiếp theo ghi N số nguyên ngẫu nhiên trong phạm vi từ 0 đến 100, mỗi dòng 10 số (các số cách nhau ít nhất một khoảng trắng).

Hãy đọc dữ liệu của file "INPUT.TXT" và lưu vào mảng một chiều

A. Thực hiện các công việc sau :

- Tìm giá trị lớn nhất của mảng A.
- Đếm số lượng số chẵn, số lượng số lẻ của mảng A.
- Hãy sắp xếp các phần tử theo thứ tự tăng dần.

Hãy ghi các kết quả vào file văn bản có tên OUTPUT.TXT theo mẫu

sau:

INPUT.TXT	OUTPUT.TXT
18	Cau a: 99
87 39 78 19 89 4 40 98 29	Cau b: 9 9
65	Cau c:
20 43 1 99 38 34 58 4	1 4 4 19 20 29 34 38 39 40
	43 58 65 78 87 89 98 99

19. (*) Viết chương trình nhập và lưu hồ sơ của sinh viên vào một file có tên là “DSSV.TXT”. Sau đó đọc file “DSSV.TXT” và cất vào mảng, hãy sắp

xếp các hồ sơ sinh viên theo thứ tự giảm dần theo điểm trung bình môn

học rồi in ra màn hình hồ sơ các sinh viên theo thứ tự đó ra màn hình có thông tin như sau :

- Mã số sinh viên.
- Họ và tên sinh viên.
- Điểm trung bình kiểm tra.
- Điểm thi hết môn.
- Điểm trung bình môn học (tính bằng (điểm TBKT+điểm thi)/2).

20. (*) Tạo một file text có tên là “INPUT.TXT” có cấu trúc như sau :

- Dòng đầu tiên ghi hai số M và N (M, N là hai số nguyên dương nhập từ bàn phím).
- Trong M dòng tiếp theo mỗi dòng ghi N số nguyên ngẫu nhiên trong phạm vi từ 0 đến 100 (các số này cách nhau ít nhất một khoảng trắng).

Hãy đọc dữ liệu từ file trên và lưu vào mảng hai chiều. Rồi thực hiện các công việc sau:

- Tìm giá trị lớn nhất của ma trận.
- Đếm số lượng số chẵn, lẻ, nguyên tố có trong ma trận.
- Hãy tính tổng các phần tử trên mỗi dòng của ma trận.

Hãy ghi kết quả này vào filetext có tên là “OUTPUT.TXT”

INPUT.TXT	OUTPUT.TXT
-----------	------------

6 6	
41 17 33 23 12 1	Cau a: 49
44 24 23 49 5 24	Cau b: 17 19
33 20 17 25 33 19	Cau c: 127 169 147 214 132
0 48 45 48 41 32	146
10 24 36 19 19 24	
30 4 23 26 27 36	

21. (**) Xét dãy số a_1, a_2, \dots, a_N . Một đoạn con của dãy là dãy các phần tử liên tiếp nhau được xác định bởi chỉ số của số bắt đầu (L) và chỉ số của số cuối cùng (R). Tổng các số trên đoạn được gọi là tổng đoạn.

Yêu cầu : Cho dãy (a_N), hãy tìm đoạn con có tổng đoạn lớn nhất

(T) Dữ liệu được cho trong tập tin văn bản SUMMAX.INP

- Dòng thứ nhất chứa số nguyên N ($0 < N \leq 30000$)
- N dòng tiếp theo, mỗi dòng chứa một số là các số của dãy đã cho theo đúng thứ tự. Giá trị tuyệt đối của mỗi số không vượt quá 30000

Kết quả tìm được ghi vào tập tin văn bản SUMMAX.OUT gồm 1 dòng ghi 3 số T, L, R.

Ví dụ :

SUMMAX.INP	SUMMAX.OUT
5	8 2 5
-1	
5	
-3	
2	
4	

22. (*) Cho dãy (a_N), hãy tìm đoạn con tăng dần có tổng lớn nhất Dữ liệu : được cho trong tập tin AMAX.INP

- Dòng 1 chứa số nguyên N ($0 < N \leq 30000$).
- N dòng tiếp theo, mỗi dòng chứa một số là các số của dãy đã cho theo đúng thứ tự. Giá trị tuyệt đối của mỗi số không vượt quá 30000.

Kết quả tìm được ghi vào tập tin văn bản AMAX.OUT gồm hai dòng:

- Dòng 1 ghi tổng của dãy con.
- Dòng 2 ghi mảng con tăng dần có tổng lớn nhất.

23. Viết chương trình nhập lý lịch một nhân viên vào danh sách các nhân

viên. Khi không nhập nữa bấm phím **Esc** và ghi vào tập tin NHANVIEN.DAT sau đó :

- Đọc từ tập tin NHANVIEN.DAT vừa tạo và in danh sách các nhân viên lên màn hình.
- Tìm và in lý lịch một nhân viên bằng các nhập và họ tên hoặc mã số nhân viên.

24. (**) Để lắp ráp một máy vi tính hoàn chỉnh cần phải có tối thiểu 10 linh kiện loại A và có thể lắp bổ sung thêm vào khoảng tối đa 8 linh kiện loại B. Tại một cửa hàng vi tính cần quản lý bán hàng các loại linh kiện tại cửa hàng. Thông tin về một loại linh kiện gồm có: Tên linh kiện, quy cách , loại, đơn giá loại 1 (chất lượng tốt – số nguyên), đơn giá loại 2 (chất lượng thường – số nguyên). Viết chương trình thực hiện những công việc sau :

Nhập vào thông tin của các loại linh kiện có ở cửa hàng. Xuất danh sách các linh kiện đã nhập theo thứ tự tăng dần của loại linh kiện và tên linh kiện. Cho biết đã có đủ 10 linh kiện loại A cần thiết để lắp ráp máy tính hay chưa?

Với giả định là cửa hàng đã có đủ 10 linh kiện loại A để lắp ráp máy. Nhập vào một số tiền để lắp ráp một máy tính. Có thể lắp được một máy tính hoàn chỉnh với các linh kiện toàn bộ theo đơn giá loại 1 hay đơn giá loại 2 hay không? Nếu số tiền trong khoảng giữa thì hãy tìm một phương án gồm những linh kiện theo đơn giá 1 và linh kiện theo đơn giá 2 để lắp?

Tất cả dữ liệu phải lưu ở tập tin.

III. KẾT LUẬN

Mục đích của kiểu dữ liệu tập tin cho phép chúng ta lưu lại những thông tin cần thiết tương đối lớn: những dữ liệu đầu vào, những kết quả của chương trình hoặc những dữ liệu dùng để test chương trình, ...

Khi thao tác trên tập tin phải thông qua 4 bước: ***Khai báo con trỏ trỏ đến tập tin, Mở tập tin, Xử lý trên tập tin và cuối cùng là Đóng tập tin.***

Lưu ý khi mở tập tin để ghi thì phải cẩn thận với thao tác tạo mới hay chỉnh sửa nội dung tập tin, di chuyển con trỏ hợp lý để tránh mất thông tin. Sử dụng hàm thao tác trên tập tin phải dùng **đúng loại hàm** cho tập tin kiểu nhị phân hay kiểu văn bản.