

NỘI DUNG CHÍNH BUỔI 9

HÀM (tt)

Ví dụ 4: Viết các hàm kiểm tra số nguyên dương nhập vào có phải là 1 số nguyên đặc biệt sau đây không? Nếu đúng trả về 1/true, ngược lại trả về 0 / false:

a) **Số nguyên tố (SNT):** là số chỉ chia hết cho 1 và chính nó

VD: 2, 3, 5, 7, ... là các SNT.

1 không phải là SNT.

2 là SNT chẵn duy nhất, bé nhất trong các SNT.

b) **Số chính phương (SCP):** là số có căn bậc 2 là số nguyên

VD: 4 là SCP vì $\sqrt{4} = 2$ (số nguyên)

2 không phải là SNT vì $\sqrt{2} = 1.4142...$ (số thực)

1 là SCP bé nhất.

c) **Số hoàn thiện (SHT):** là số có tổng ước số nhỏ hơn nó bằng chính nó

VD: 6 là SHT vì tổng các ước số nhỏ hơn 6 là $1+2+3 = 6$

8 không phải là SHT vì tổng các ước số nhỏ hơn 8 là $1+2+4 = 7 \neq 8$

6 là SHT bé nhất.

Euclid đã khám phá ra 4 số hoàn thiện nhỏ nhất dưới dạng: $2^{n-1}(2^n - 1)$, với $2^n - 1$ là phải là SNT.

- $n = 2: 2^1(2^2-1) = 6$

- $n = 3: 2^2(2^3-1) = 28$

- $n = 5: 2^4(2^5-1) = 496$

- $n = 7: 2^6(2^7-1) = 8128$

d) **Số đối xứng:** là số có số đảo bằng chính nó (viết xuôi ngược, từ trái qua, từ phải sang đều là chính nó)

VD: 1, 22, 121, 12321, ...

Kỹ thuật đặt cờ hiệu: (dùng cho bài toán kiểm tra đúng/sai)

- Khai báo và gán giá trị cho biến cờ hiệu.
- Xét các trường hợp làm thay đổi giá trị biến cờ hiệu.
- Trả về giá trị biến cờ hiệu

```
//Hàm kiểm tra số nguyên tố trả về 1(đúng)/0(sai)
int KiemTraSNT(int n)
{
    //Khai báo và gán giá trị cho biến cờ hiệu
    int snt=1;//snt=1: n là SNT; snt=0: n không phải SNT

    //Xét các trường hợp làm thay đổi giá trị biến cờ hiệu
    if(n<2)
        snt=0;//n không phải SNT
    else
        //Lần lượt xét các số i từ 2 => n-1, thử lấy n chia cho các số nguyên i này
        for(int i=2;i<n;i++)/*viết lại tối ưu hơn là for(int i=2;i<=n/2;i++) hoặc
                               for(int i=2;i<=sqrt((float)n);i++)*/
            if(n%i==0)//nếu n chia hết cho i bất kỳ
            {
                snt=0;//n không phải SNT vì n chia hết cho 1 số khác 1 và n
                break;//thoát khỏi vòng lặp ngay vì n đã vi phạm tính chất SNT
            }
    return snt;//trả về biến cờ hiệu
}

void main()
{
    int n;
    //Nhập số nguyên n
    if(KiemTraSNT(n)==1)//Gọi hàm kiểm tra n có phải là SNT không? Nếu đúng (trả về 1) thì
        printf("%d là SNT",n);
    else //Nếu sai (trả về 0) thì
        printf("%d không là SNT",n);
}
```

```
//Hàm kiểm tra số nguyên tố trả về true(đúng)/false(sai)
bool KiemTraSNT(int n)
{
    //Khái báo và gán giá trị cho biến cờ hiệu
    bool snt=true;//snt=true: n là SNT; snt=false: n không phải SNT

    //Xét các trường hợp làm thay đổi giá trị biến cờ hiệu
    if(n<2)
        snt=false;//n không phải SNT
    else
        //Lần lượt xét các số i từ 2 => n-1, thử lấy n chia cho các số nguyên i này
        for(int i=2;i<=n/2;i++)
            if(n%i==0)//nếu n chia hết cho i bất kỳ
            {
                snt= false;//n không phải SNT vì n chia hết cho 1 số khác 1 và n
                break;//thoát khỏi vòng lặp ngay vì n đã vi phạm tính chất SNT
            }
    return snt;//trả về biến cờ hiệu
}

void main()
{
    int n;
    //Nhập số nguyên n
    if(KiemTraSNT(n)== true)//hoặc if(KiemTraSNT(n))
        printf("%d là SNT",n);
    else //Nếu sai (trả về false)
        printf("%d không là SNT",n);
}

void main()
{
    int n;
    //Nhập số nguyên n
    if(KiemTraSNT(n)== false)//hoặc if(!KiemTraSNT(n))
        printf("%d không là SNT",n);
    else //Nếu sai (trả về false)
        printf("%d là SNT",n);
}
```

```
//Hàm kiểm tra số chính phương trả về 1(đúng)/0(sai)
int KiemTraSCP(int n)
{
    //Khai báo và gán giá trị cho biến cờ hiệu
    int scp=0; //scp=1: n là SCP; scp=0: n không phải SCP

    //Xét các trường hợp làm thay đổi giá trị biến cờ hiệu
    if(n<1)
        scp=0; //n không phải SCP
    else
        //Lần lượt xét các số nguyên i từ 1 =>  $\sqrt{n}$ 
        for(int i=1; i<=sqrt(n); i++)
            if(i*i==n) //nếu tích i*i bất kỳ = n, có nghĩa là  $\sqrt{n} = i$  nguyên
            {
                scp=1; //n là SCP vì tìm thấy i nguyên để  $\sqrt{n} = i$  nguyên
                break; //thoát khỏi vòng lặp ngay vì n đã thỏa tính chất SCP
            }
    return scp; //trả về biến cờ hiệu
}

//Hàm kiểm tra số chính phương trả về true(đúng)/false(sai)
bool KiemTraSCP(int n)
{
    //Khai báo và gán giá trị cho biến cờ hiệu
    int scp=false; //scp=true: n là SCP; scp=false: n không phải SCP

    //Xét các trường hợp làm thay đổi giá trị biến cờ hiệu
    if(n<1)
        scp= false; //n không phải SCP
    else
        //Lần lượt xét các số nguyên i từ 1 =>  $\sqrt{n}$ 
        for(int i=1; i<=sqrt(n); i++)
            if(i*i==n) //nếu tích i*i bất kỳ = n, có nghĩa là  $\sqrt{n} = i$  nguyên
            {
                scp=true; //n là SCP vì tìm thấy i nguyên để  $\sqrt{n} = i$  nguyên
                break; //thoát khỏi vòng lặp ngay vì n đã thỏa tính chất SCP
            }
    return scp; //trả về biến cờ hiệu
}
```

```
//Hàm kiểm tra số chính phương trả về 1(đúng)/0(sai): sử dụng ép kiểu
int SCP(int n)
{
    return sqrt((float)n)==(int)sqrt((float)n)? 1:0;
}

//Hàm kiểm tra số chính phương trả về true(đúng)/false(sai): sử dụng ép kiểu
bool SCP(int n)
{
    return sqrt((float)n)==(int)sqrt((float)n)? true:false;
}

//Hàm kiểm tra số hoàn thiện trả về giá trị true/false
bool KiemTraSHT(int n)
{
    //Khai báo biến cờ hiệu
    bool sht;//sht=true: n là SHT; sht=false: n không phải SHT
    //Xét các trường hợp làm thay đổi giá trị biến cờ hiệu
    if(n<6)
        sht= false;//n không phải SHT
    else
    {
        int tongus=0; //Khai báo biến tongus để lưu tổng giá trị các ước số của n
        for(int i=1;i<=n/2;i++) //Lần lượt xét các số nguyên i từ 1 => n/2
            if(n%i==0)//nếu n chia hết cho i bất kỳ từ 1 => n/2
                tongus+=i;//cộng dồn giá trị của i vào biến tongus
        sht=(tongus==n)?true:false; /*nếu tổng giá trị các ước số nhỏ hơn n bằng chính n thì n
                                   chính là SHT, ngược lại thì không phải là SHT*/
    }
    return sht;//trả về biến cờ hiệu
}

//Tương tự, xây dựng hàm kiểm tra số hoàn thiện trả về 1/0
```

```
//Hàm kiểm tra số đối xứng
bool KiemTraSDX(int n)
{
    //Khai báo biến cờ hiệu
    bool sdx;//sdx=true: n là SDX; sdx=false: n không phải SDX
    //Xét các trường hợp làm thay đổi giá trị biến cờ hiệu
    if(n<1)
        sdx=false;//n không phải SDX
    else
    {
        /*Khai báo biến m nhận giá trị của n để tính toán, dao để lưu số đảo, dv để lưu chữ
        số hàng đơn vị*/
        int m=n,dao=0,dv;
        //Lặp trong khi m>0 (chưa hết các chữ số của m)
        while(m>0)
        {
            dv=m%10; //lấy chữ số phải nhất của m (hàng đơn vị) và lưu vào biến dv
            dao=dao*10+dv; //ghép chữ số phải nhất của m (hàng đơn vị) vào số đảo
            m/=10; //bỏ chữ số phải nhất của m (hàng đơn vị)
        }
        sdx=(dao==n)?true:false;//nếu số đảo bằng n thì n là SDX, ngược lại n không là SDX
    }
    return sdx;//trả về biến cờ hiệu
}

//Tương tự, xây dựng hàm kiểm tra số đối xứng trả về 1/0
```

=> **Xây dựng CT hoàn chỉnh có menu cho phép người dùng chọn lựa các xử lý như sau:**

1. Nhập số nguyên dương n.
2. Kiểm tra n có phải là số nguyên tố không?
3. Kiểm tra n có phải là số chính phương không?
4. Kiểm tra n có phải là số hoàn thiện không?
5. Kiểm tra n có phải là số đối xứng không?
0. Thoát CT.

(6) Xây dựng thư viện hàm

- Xây dựng thư viện chứa các hàm tự định nghĩa để tái sử dụng khi cần
 - ❖ **Bước 1:** Tạo file header **ten_thu_vien_ham.h** để khai báo các nguyên hàm:

```
#ifndef ten_thu_vien
#define ten_thu_vien
Nguyên hàm 1;
Nguyên hàm 2;
...
Nguyên hàm n;
#endif
```

VD: Tạo file header **kiem_tra_so.h** như sau:

```
#ifndef ktso
#define ktso
int KiemTraSNT(int n);
int KiemTraSCP(int n);
int KiemTraSHT(int n);
int KiemTraSDX(int n);
#endif
```

- ❖ **Bước 2:** Tạo file source **ten_thu_vien_ham.cpp** để định nghĩa các hàm tương ứng với các nguyên hàm đã khai báo:

```
#include <...>
Hàm 1;
Hàm 2;
...
Hàm n;
```

VD: Tạo file source **kiem_tra_so.cpp** như sau:

```
#include<cmath>
```

```
int KiemTraSNT(int n)
{
    int snt=1;
    if(n<2)
        snt=0;
    else
        for(int i=2;i<n;i++)
            if(n%i==0)
            {
                snt=0;
                break;
            }
    return snt;
}

int KiemTraSCP(int n)
{
    int scp=0;
    if(n<1)
        scp=0;
    else
        for(int i=1;i<=n/2;i++)
            if(i*i==n)
            {
                scp=1;
                break;
            }
    return scp;
}
```



```
int KiemTraSHT(int n)
{
    int sht;
    if(n<6)
        sht=0;
    else
    {
        int tongus=0;
        for(int i=1;i<=n/2;i++)
            if(n%i==0)
                tongus+=i;
        sht=(tongus==n)?1:0;
    }
    return sht;
}

int KiemTraSDX(int n)
{
    int sdx;
    if(n<1)
        sdx=0;
    else
    {
        int m=n,dao=0,dv;
        while(m>0)
        {
            dv=m%10;
            dao=dao*10+dv;
            m/=10;
        }
        sdx=(dao==n)?1:0;
    }
    return sdx;
}
```

❖ Bước 3: Sử dụng thư viện hàm đã xây dựng:

```
#include <ten_thu_vien_chuan>
#include "ten_thu_vien_ham.h"
```

Lưu ý:

- Sử dụng cặp dấu <...> đối với các **thư viện chuẩn có sẵn** trong C => Khi biên dịch, CT sẽ tìm thư viện trong ổ đĩa cài đặt phần mềm (C:\Microsoft Visual Studio\VC\include) để liên kết.
- Sử dụng cặp dấu "..." đối với các **thư viện tự định nghĩa** => Khi biên dịch, CT sẽ tìm thư viện trong thư mục lưu dự án hiện hành, nếu không tìm thấy CT sẽ tiếp tục tìm thư viện trong ổ đĩa cài đặt phần mềm (C:\Microsoft Visual Studio\VC\include) để liên kết.
- Trường hợp file sử dụng thư viện không cùng vị trí lưu trữ với file thư viện thì phải sử dụng đường dẫn tuyệt đối / tương đối đến file thư viện

VD:

```
#include "D:\KTLT\KiemTraSo\kiem_tra_so.h" /*đường dẫn tuyệt đối tính từ ổ đĩa gốc
truy xuất đến vị trí file thư viện*/
#include "..\KTLT\KiemTraSo\kiem_tra_so.h" /*đường dẫn tương đối tính từ vị trí file
sử dụng thư viện truy xuất đến vị trí
file thư viện*/
```

VD: Sử dụng thư viện kiểm tra số đã khai báo và định nghĩa như sau:

```
#include<stdio.h>
#include<conio.h>
#include"kiem_tra_so.h"
void main()
{
    int n;
    //Nhập n
    if(KiemTraSNT(n)==1)
        printf("%d la SNT",n);
    else
        printf("%d khong la SNT",n);
    ...
}
```