Ngôn ngữ lập trình C++

KIỂU DỮ LIỆU STRUCT

Để mô tả các đối tượng có nhiều thông tin, ta thường sử dụng kiểu dữ liệu **pair**, hoặc kiểu dữ liệu **struct**. Trong những trường hợp, đối tượng có nhiều thông tin, ta sử dụng cấu trúc dữ liệu **struct** là phù hợp nhất.

1. Cách định nghĩa kiểu dữ liệu struct

```
struct [Tên kiểu]{
    Khai báo thông tin 1;
    Khai báo thông tin 2;
    .....
    Khai báo thông tin n.
};
```

2. Cách khai báo biến struct

```
[Tên kiểu]    <tên biến>;
```

3. Cách truy cập đến các thành phần của struct

```
[tên biến].<tên thành phần cần truy cập>
```

Ví dụ 1:

Viết chương trình nhập vào N ($N \leq 100$) và N học sinh có thông tin về điểm Toán, Tin, Anh. Đưa ra tổng điểm (Toán + Tin + Anh) của N học sinh theo thứ tự được nhập vào.

```
#include<bits/stdc++.h>
using namespace std;
struct hocsinh{
    int Toan;
    int Tin;
    int Anh;
};

hocsinh ds[1000];
int n;
int main()
{
    cin>>n;
    for(int i = 1; i <= n; i++)
        cin>>ds[i].Toan>>ds[i].Tin>>ds[i].Anh;
    cout<<"Danh sach tong diem: "<<endl;
    for(int i = 1; i <= n; i++)
        cout<<ds[i].Toan + ds[i].Tin + ds[i].Anh<<endl;
    return 0;
}
```

Kết quả:

```
5
8 9 10
9 9 8
9 9 9
8 8 10
7 8 9
Danh sach tong diem:
27
26
27
26
24
```

4. Cách định nghĩa các phép toán trên kiểu dữ liệu struct

Các kiểu dữ liệu mà ngôn ngữ lập trình đã hỗ trợ sẵn đều đi kèm với một số phép toán (các thao tác) trên chúng. Đối với kiểu dữ liệu **struct,** khi muốn sử dụng các phép toán trên kiểu dữ liệu này thì người lập trình cần định nghĩa các phép toán đó.

Sau đây, ta định nghĩa phép toán so sánh để sắp xếp giữa các đối tượng.

Ví dụ 2:

Viết chương trình nhập vào N ($N \leq 100$) và N học sinh có thông tin về điểm Toán, Tin, Anh. Đưa ra thông tin về học sinh gồm:

- Điểm Toán, Tin, Anh và tổng điểm (Toán + Tin + Anh) của N học sinh theo thứ tự tổng điểm tăng tăng dần.

```
#include<bits/stdc++.h>
using namespace std;
struct hocsinh{
    int Toan;
    int Tin;
    int Anh;
};
bool compair(hocsinh a, hocsinh b)
{
    int x = a.Toan + a.Tin + a.Anh;
    int y = b.Toan + b.Tin + b.Anh;
    if(x < y) return true;
    else return false;
}
hocsinh ds[1000];
int n, x;
int main()
{
    cin>>n;
    for(int i = 1; i <= n; i++)
        cin>>ds[i].Toan>>ds[i].Tin>>ds[i].Anh;
    sort(ds + 1, ds + 1 + n, compair);
    cout<<"Danh sach hoc sinh da duoc sap xep "<<endl;
    for(int i = 1; i <= n; i++)
        {
           x = ds[i].Toan + ds[i].Tin + ds[i].Anh;
           cout<<ds[i].Toan<<' '<<ds[i].Tin<<' '<<ds[i].Anh<<' '<<x<<endl;
        }
    return 0;
}
```

Kết quả

```
5
8 8 7
7 7 7
6 7 5
9 9 9
7 6 9
Danh sach hoc sinh da duoc sap xep
6 7 5 18
7 7 7 21
7 6 9 22
8 8 7 23
9 9 9 27
```

1☼. Sắp xếp danh sách học sinh

Có danh sách gồm N học sinh. Mỗi học sinh có thông tin về điểm Toán, điểm Tin và điểm Anh. Hãy sắp xếp danh sách học sinh theo tiêu chí:

- Tổng điểm Toán, Tin, Anh giảm dần:
  - Nếu tổng điểm bằng nhau thì điểm Toán giảm dần.
    - Nếu điểm Toán bằng nhau thì điểm Tin giảm dần.

Dữ liệu cho trong file **SortPupil.Inp** gồm:

- Dòng đầu ghi số nguyên dương N ($N \leq 10^5$).
- Dòng thứ i gồm 3 số nguyên a, b, c tương ứng là điểm Toán, Tin, Anh của học sinh thứ i ($0 \leq a, b, c \leq 10$).

Kết quả ghi ra file **SortPupil.Out** gồm N dòng là thông tin của N học sinh đã được sắp xếp theo tiêu chí trên. Mỗi dòng ghi 4 số: a, b, c, x tương ứng là điểm Toán, Tin, Anh và tổng điểm.

Ví dụ:

SortPupil.Inp	SortPupil.Out
5 6 6 7 8 7 9 7 5 9 6 7 6 6 6 9	8 7 9 24 7 5 9 21 6 6 9 21 6 7 6 19 6 6 7 19

2☼. Chọn hình vuông

Có N hình vuông, hình vuông thứ i có độ dài cạnh là A_i. Hãy tìm cách chọn được nhiều nhất các hình vuông sao cho, trong các hình vuông được chọn thỏa mãn:

Diện tích của hai hình vuông chênh nhau không quá S.

Dữ liệu cho trong file **SelectSquare.Inp** gồm:

- Dòng đầu ghi hai số nguyên dương N và S.
- Dòng thứ 2 ghi N số nguyên dương $A_1, A_2, \ldots, A_N$.

Kết quả ghi ra file **SelectSquare.Out** là số hình vuông nhiều nhất chọn được.

Ví dụ:

SelectSquare.Inp	SelectSquare.Out	Giải thích
5 10 2 3 1 9 4	3	Chọn 3 hình vuông có độ dài cạnh: 2, 3, 1

Giới hạn:

- $A_i \leq 10^6$;
- $N \leq 10^5$;
- $S \leq 10^{12}$.

3☼. Chọn hình chữ nhật

Có N hình chữ nhật, hình chữ nhật thứ i có chiều dài là A_i và chiều rộng là B_i. Hãy tìm cách chọn được nhiều nhất các hình chữ nhật sao cho, trong các hình chữ nhật được chọn thỏa mãn:

- Diện tích của hai hình chữ nhật chênh nhau không quá S.
- Chiều dài của hai hình chữ nhật chênh nhau không quá T.

Dữ liệu cho trong file **SelectRect.Inp** gồm:

- Dòng đầu ghi ba số nguyên dương N, S và T.
- Dòng thứ 2 ghi N số nguyên dương $A_1, A_2, \ldots, A_N$.
- Dòng thứ 3 ghi N số nguyên dương $B_1, B_2, \ldots, B_N$.

Kết quả ghi ra file **SelectRect.Out** là số hình vuông nhiều nhất chọn được.

Ví dụ:

SelectRect.Inp	SelectRect.Out	Giải thích
5 3 3 2 3 1 9 4 1 1 1 1 1	4	Chọn 3 hình chữ nhật có độ dài chiều dài là: 2, 3, 1 và 4.

Giới hạn:

- $1 \le B_i \le A_i \le 10^6$;
- $N \le 10^3$;
- $S \le 10^{12}$;
- $T \le 10^6$.

4☼. Chọn ô trên đường chéo (3)

Cho bảng số gồm N dòng và M cột. Các dòng được đánh số từ 1 đến N (từ trên xuống dưới), các cột được đánh số từ 1 đến M (từ trái sang phải). Ô ở dòng i cột j, ta kí hiệu là ô (i, j) và có ghi số nguyên A_{ij}.

Yêu cầu: Tìm 6 ô nằm trên đường chéo:

$(i,j), (i+1,j+1), (i+2,j+2)$;

$(u,v), (u+1,v-1), (u+2,v-2)$ sao cho tổng 6 số ghi trên 6 ô này có giá trị lớn nhất.

Dữ liệu cho trong file **Diagon3.Inp** gồm:

- Dòng đầu ghi hai số nguyên dương N và M ($3 \le N, M \le 1000$).
- N dòng sau, mỗi dòng ghi M số nguyên thuộc tập [-1000; 1000].

Kết quả ghi ra file **Diagon3.Out** là tổng lớn nhất của 6 ô tìm được (chú ý là 6 ô phân biệt).

Ví dụ:

Diagon3.Inp	Diagon3.Out
4 5 1 2 3 4 5 1 1 1 1 1 9 9 9 1 1 0 0 0 0 0	25