During this semester, you have already learned how to create a class. Class is the basic of object oriented programming concept. In C++, there is another tool which is really similar to class and you saw it in your lecture couple times before. The topic I’m going to introduce today is “struct”.

Struct is very similar to class, which can contain sets of variables, functions. The only obviously difference is all the members (variables or functions) are default by private in class, but default by public in struct.

Here is a simple example:

#include <iostream>

#include <string>

using namespace std;

struct student

{

public:

student(int x, char y, string z)

{

grade = y;

cwid = x;

name = z;

}

void show\_record()

{

cout << "cwid: " << cwid << endl;

cout << "name: " << name << endl;

cout << "grade: " << grade << endl;

}

private:

char grade;

int cwid;

string name;

};

int main()

{

student s1(893397685, 'A', "Trout");

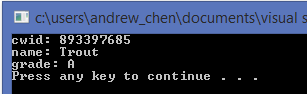
s1.show\_record();

system("pause");

return 0;

}

Output:



Since every members in the struct are default by public, so you don’t even need to declare the key word “public” in the struct, it won’t hurt. But you would like to keep the habit to declare things clearly in case of getting confuse.

The same like class, you can also declare an array of a struct type.

For example you have a struct “student”

struct student

{

char grade;

int cwid;

string name;

};

Then you can declare an array of students like this:

Int main()

{

student s[10];

}

Or just a single struct:

Int main()

{

student peter;

}

Since I didn’t declare the variables in private, they are all public by default. You can easily access and even change the value of the variables in student struct by the simble ‘.’ (a dot)

Exp:

peter.grade = ‘A’;

peter.cwid = 12345;

peter.name = “Peter”;

( If the varibles are private, you can’t directly access the variables, you need to create functions to do that.)

Now let’s go back to the pointer. Remember a pointer can point to any type of data include a class or a struct. So let’s create a pointer which points to the struct peter.

student \*p = &peter; // remember peter is a student struct type variable we declared

by this: student peter, so the pointer must be the same

type (student) to points to the student type variable;

So, now the pointer p holds the same address as peter, we all know peter contains three variables (grad, cwid, and name), how to access them just add a dot after the struct name.

like this: peter.grade = ‘A’;

If you want to output it, the same thing just: cout << peter.grade;

Notice that every thing here is the same as class, you can do the same control for public variables in a class. But if you want to access the variable through the pointer p, slightly different here:

P is a pointer which points to the address of peter( a student type variable or you can say it’s an object). If you want to access the variables in the address which p points to, you need use “->” instead the dot ‘.’ This is the rule for pointer.

For example:

cout << p->cwid << endl;

or you can do this cout << (\*p).cwid << endl;

(\*p) is to get the content of the address in p (remember \* is to get the content from an address). So you get the content which is a student struct then use ‘.’ To access the variables in the struct.

Here is the example:

int main()

{

student peter;

cout << "The address of peter: \t" << &peter << endl;

peter.grade = 'A';

peter.cwid = 12345;

peter.name = "Peter";

cout << "Get cwid by peter.cwid:\t" << peter.cwid << endl;

cout << endl;

student \*p = &peter;

cout << "The address which p points to: \t" << p << endl;

cout << "Get cwid by p->cwid: \t" << p->cwid << endl;

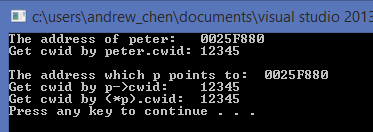
cout << "Get cwid by (\*p).cwid: \t" << (\*p).cwid << endl;

system("pause");

return 0;

}

Output:



The same thing for a class !

If you have any questions, feel free to come to any SI session. We can discuss more details and go through more examples.