**Classes**

Public and Private Access modifiers

**Why use classes instead of structures:-**

Classes and structures are somewhat the same but still, they have some differences. For example, we cannot hide data in structures which means that everything is public and can be accessed easily which is a major drawback of the structure because structures cannot be used where data security is a major concern. Another drawback of structures is that we cannot add functions in it.

**Classes:**

Classes are user-defined data-types and are a template for creating objects. Classes consist of variables and functions which are also called class members.

**Public Access :**

All the variables and functions declared under public access modifier will be available for everyone. They can be accessed both inside and outside the class. Dot (.) operator is used in the program to access public data members directly.

**Private Access :**

All the variables and functions declared under a private access modifier can only be used inside the class. They are not permissible to be used by any object or function outside the class.

**Nesting of Member Functions:**

If one member function is called inside the other member function of the same class it is called nesting of a member function.

#include<bits/stdc++.h>

using namespace std;

class Employee{

    private:

        int a, b, c;

    public:

        int d, e;

        void setData(void);

        void getData(void);

};

void Employee :: setData(void){

    cout<<"enter the value of a b and c:"<<endl;

    cin>>a>>b>>c;

}

void Employee :: getData(void){

    setData(); // Nesting of member function

    cout<<"The value of a is "<<a<<endl;

    cout<<"The value of b is "<<b<<endl;

    cout<<"The value of c is "<<c<<endl;

    cout<<"The value of d is "<<d<<endl;

    cout<<"The value of e is "<<e<<endl;

}

int main(){

    Employee harry;

harry.d = 45; //  harry.e = 30;

    harry.getData();

    return 0;

}

**Run Program:**

enter the value of a b and c:

**Input :** 4 5 6

**Output:**

The value of a is 4

The value of b is 5

The value of c is 6

The value of d is 45

The value of e is 4194432 (Garbage value e)

**Example 2 : Swap the binary number**

#include<bits/stdc++.h>

using namespace std;

class binary{

//  private:

        string s;

    public:

        void read(void);

        void chk\_bin(void);

        void ones\_complement(void);

        void display(void);

};

void binary :: read(void){

    cout<<"Enter a binary Nmuber :"<<endl;

    cin>>s;

}

void binary ::  chk\_bin(void){

    for (int i=0; i<s.length(); i++){

        if (s.at(i)!='0' && s.at(i)!='1'){

            cout<<"incorret binary number "<<endl;

            exit (0);

        }

    }

}

void binary :: ones\_complement(void){

    chk\_bin();  // Nesting of member function

    for (int i=0; i<s.length(); i++){

        if(s.at(i)=='0')

        {

            s.at(i)='1';

        }

        else{

            s.at(i)='0';

        }

    }

}

void binary :: display(void){

    cout<<"Displaying your Binary number :"<<endl;

    for (int i = 0; i < s.length(); i++){

        cout<<s.at(i);

    }

    cout<<endl;

}

int main(){

    binary b;

    b.read();

    //b.chk\_bin();

    cout<<"first Display :"<<endl;

    b.display();

    b.ones\_complement();

    cout<<"Second Display :"<<endl;

    b.display();

    return 0;

}

**Run Program:**

Enter a binary Nmuber :

**Input :** 1010000111

**Output:**

first Display :

Displaying your Binary number :

1010000111

Second Display :

Displaying your Binary number :

0101111000

Run Program: