```
/*

Q5.   WAP to run stack class with push() and pop() functions and following
details
a.   Derive 3 classes arraystack, linkedstack and multistack from base
class stack
b.   Make it compulsary for every derive class to override push() , pop()
and display methods of base class stack.
c.   Every derived class can have its own data members as follows:
i.   Arraystack has a dynamic array
ii.  Linkedstack will have a structure object for linkedlist
iii. Multostact will hav a 2D array
d.   Implement this program using dynamic binding.
*/

#include <iostream>
using namespace std;

class person          //base class
{   int age;
    string name;
public:
    person()
    {   int age=0;
        name="na";
    }
    void input();
    void output();
    ~person()                           //destructor
    {  cout<<"Person object deleted";
    }
};
void person::input()
{
     cout<<"Enter your age :";
        cin>>age;
        cout<<"Enter your name :";
        cin>>name;
}
void person::output()
{   cout<<"Age is"<<age<<endl;
    cout<<"Name is"<<name<<endl;
}

class employee:public person
```

```
{
    int empid;
public:
    employee()                          //constructor
    {    empid=0;
    }
    void input1();
    void output1();
    ~employee()                          //destructor for class Employee
    {  cout<<"Employee Object Deleted";
    }
};
void employee::input1()
{   input();
    cout<<"Enter employee ID:";
    cin>>empid;
}
void employee::output1()
{
    output();
    cout<<"Employee ID:"<<empid<<endl;
}
class manager:protected employee
{
    int bonus;
public:
    manager()
    {  bonus=0;
    }
    void input2();
    void output2();
    ~manager()                              //destructor for class
manager
    {  cout<<"manager Object Deleted ";
    }
};
 void manager::input2()
    {   input1();
        cout<<"enter your Bonus ";
        cin>>bonus;
    }
void manager::output2()
    {
        output1();
        cout<<"Bonus is\n"<<bonus;
```

```
    }
class CEO:private employee
{   int experience;
    public:
        CEO()                           //constructor
        {
            experience=0;
        }
     void input3();
    void output3();
    ~CEO()
    {
        cout<<"CEO object deleted ";
    }

};
void CEO::input3()
{   // input2();  //can not be access

    cout<<"Enter your Exp ";
    cin>>experience;
}
void CEO::output3()
{    cout<<"Experience :"<<experience<<endl;
}

int main()
{
    employee e1;
    e1.input1();
    e1.output1();
    manager m1;
    m1.input2();
    m1.output2();
    CEO c1;
    c1.input3();
    c1.output3();

}
```

OUTPUT

```
D:\Learning\codeblock\ASS3Q2\bin\Debug\ASS3Q2.exe
Enter your age :35
Enter your name :Helium
Enter employee ID:2
Age is35
Name isHelium
Employee ID:2
Enter your age :25
Enter your name :Oxygen
Enter employee ID:8
enter your Bonus 700
Age is25
Name isOxygen
Employee ID:8
Bonus is
700Enter your Exp 5
Experience :5
```