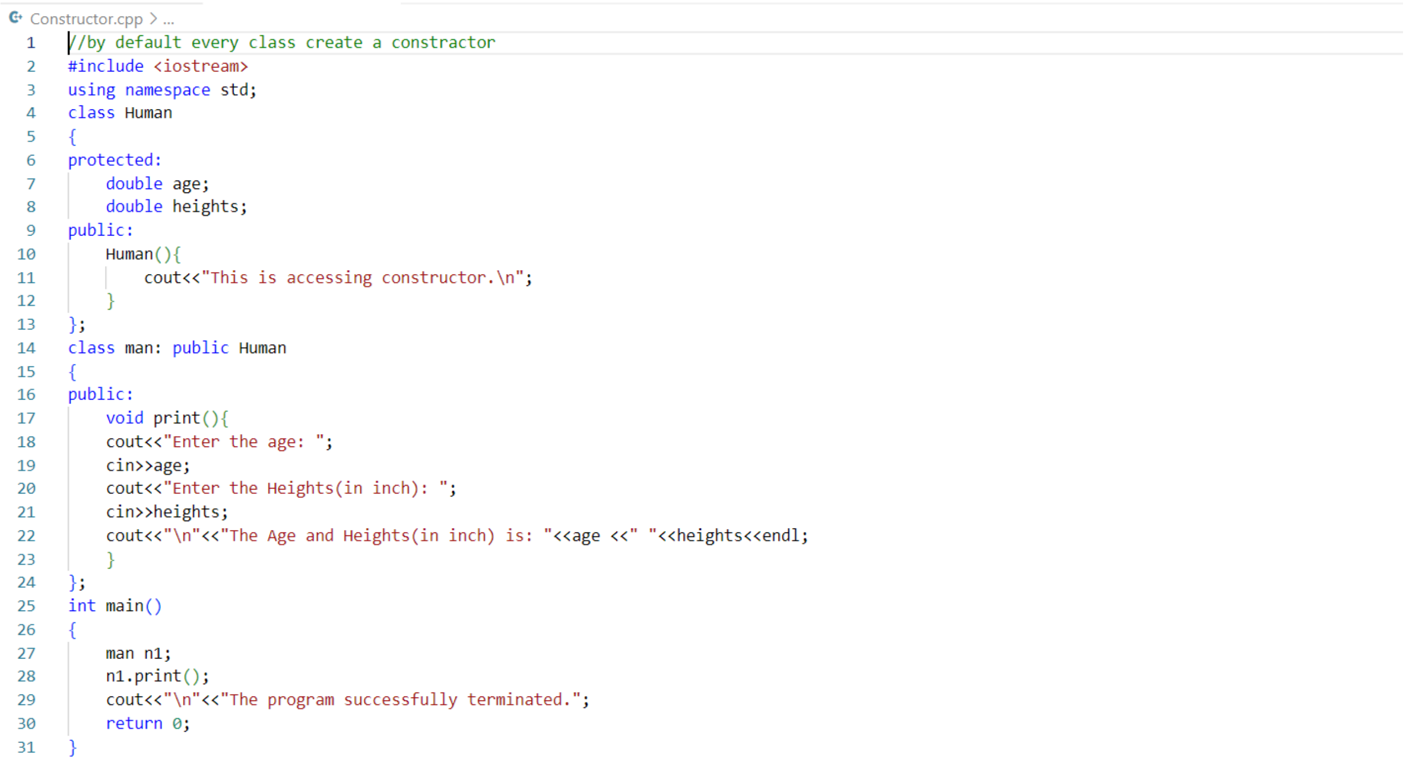
LAB-WORK 11

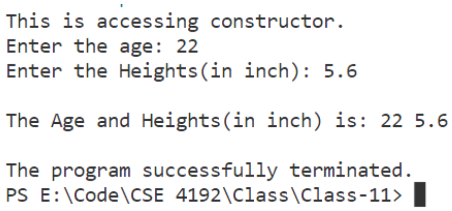
1.Write a C++ program uing a Constructor.

CODE:



```cpp
//by default every class create a constractor
#include <iostream>
using namespace std;
class Human
{
protected:
    double age;
    double heights;
public:
    Human(){
        cout<<"This is accessing constructor.\n";
    }
};
class man: public Human
{
public:
    void print(){
    cout<<"Enter the age: ";
    cin>>age;
    cout<<"Enter the Heights(in inch): ";
    cin>>heights;
    cout<<"\n"<<"The Age and Heights(in inch) is: "<<age <<" "<<heights<<endl;
    }
};
int main()
{
    man n1;
    n1.print();
    cout<<"\n"<<"The program successfully terminated.";
    return 0;
}
```

OUTPUT:



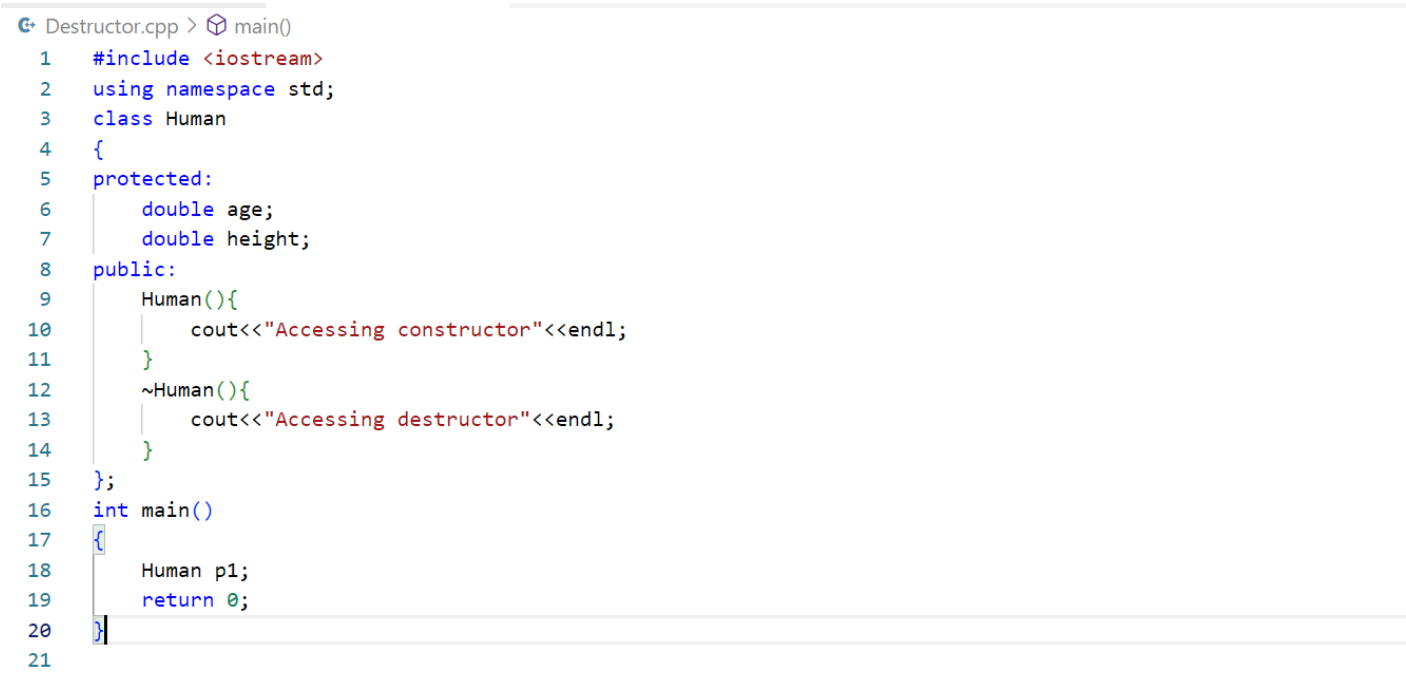
```
This is accessing constructor.
Enter the age: 22
Enter the Heights(in inch): 5.6

The Age and Heights(in inch) is: 22 5.6

The program successfully terminated.
PS E:\Code\CSE 4192\Class\Class-11>
```

02.Write a C++ program using Destructor.

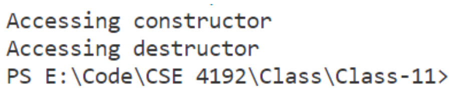
CODE:



```cpp
#include <iostream>
using namespace std;
class Human
{
protected:
    double age;
    double height;
public:
    Human(){
        cout<<"Accessing constructor"<<endl;
    }
    ~Human(){
        cout<<"Accessing destructor"<<endl;
    }
};
int main()
{
    Human p1;
    return 0;
}

```

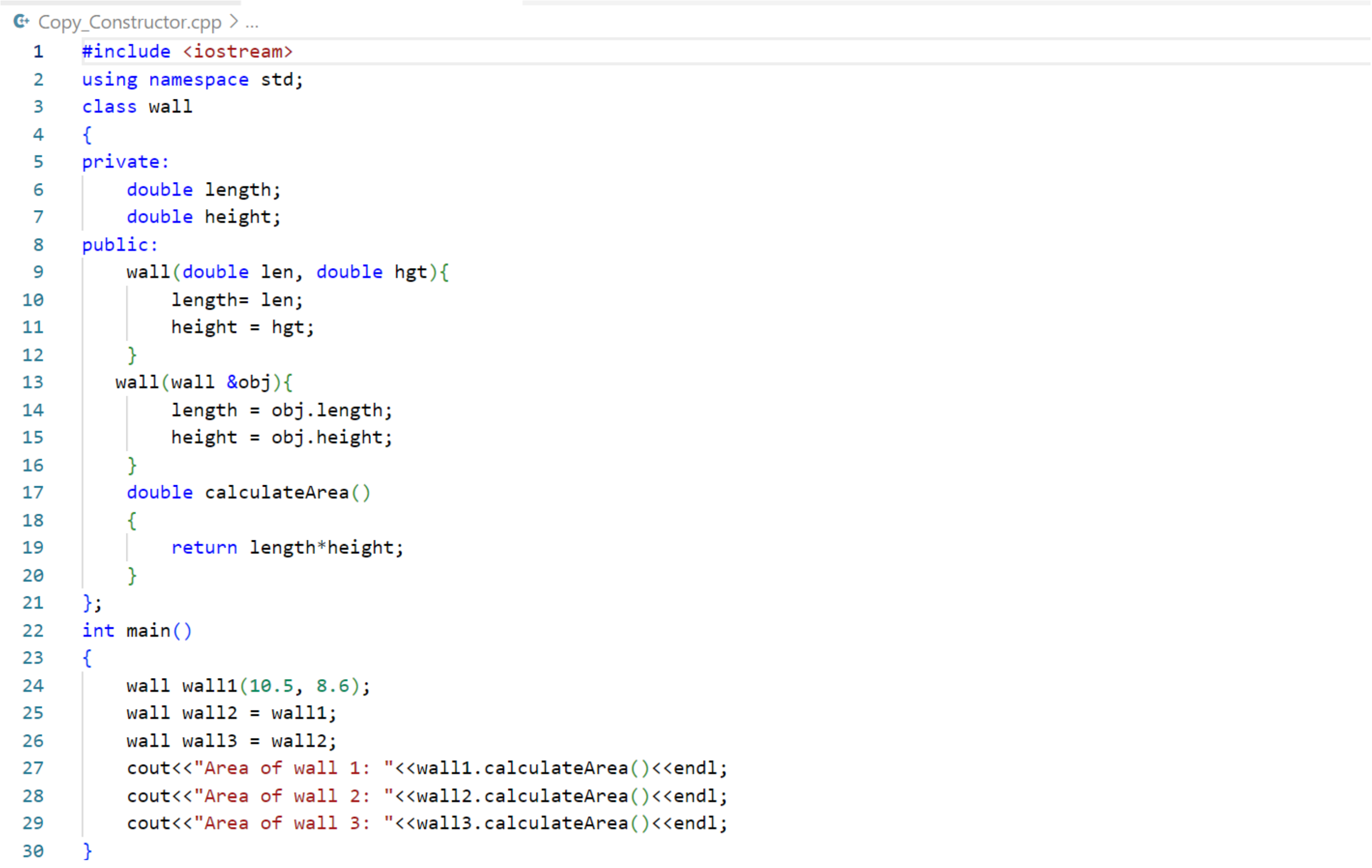
OUTPUT:



```
Accessing constructor
Accessing destructor
PS E:\Code\CSE 4192\Class\Class-11>
```

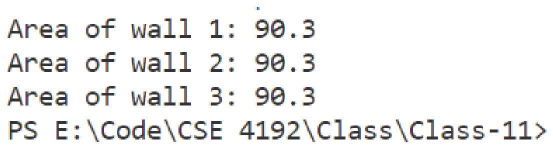
03.Write a C++ Program using Copy Constructor.

CODE:



```cpp
#include <iostream>
using namespace std;
class wall
{
private:
    double length;
    double height;
public:
    wall(double len, double hgt){
        length= len;
        height = hgt;
    }
   wall(wall &obj){
        length = obj.length;
        height = obj.height;
    }
    double calculateArea()
    {
        return length*height;
    }
};
int main()
{
    wall wall1(10.5, 8.6);
    wall wall2 = wall1;
    wall wall3 = wall2;
    cout<<"Area of wall 1: "<<wall1.calculateArea()<<endl;
    cout<<"Area of wall 2: "<<wall2.calculateArea()<<endl;
    cout<<"Area of wall 3: "<<wall3.calculateArea()<<endl;
}
```

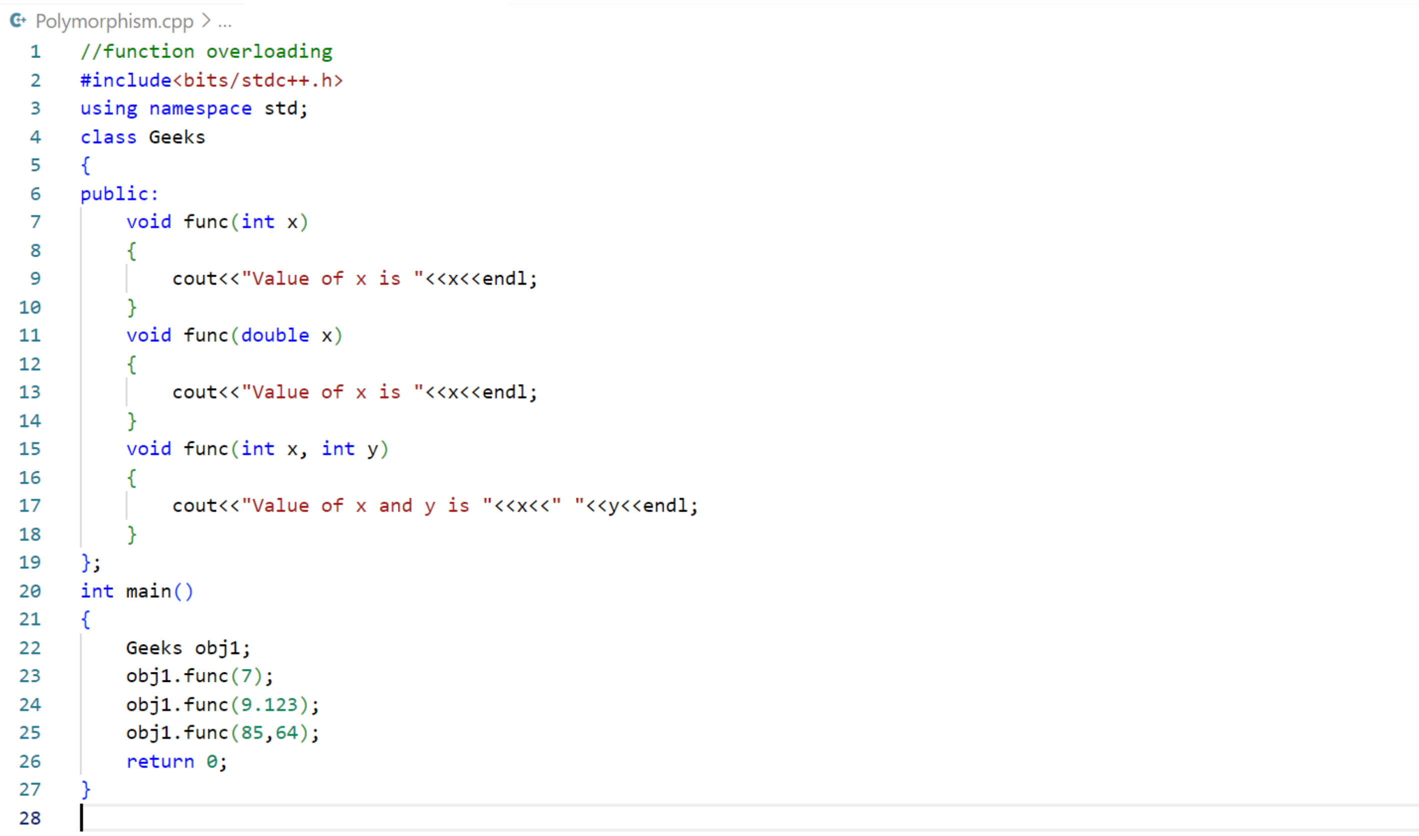
OUTPUT:



```
Area of wall 1: 90.3
Area of wall 2: 90.3
Area of wall 3: 90.3
PS E:\Code\CSE 4192\Class\Class-11>
```

04.Write a C++ program using polymorphism (overloading).

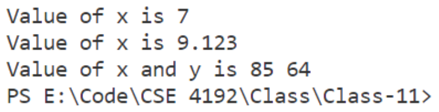
CODE:



```cpp
//function overloading
#include<bits/stdc++.h>
using namespace std;
class Geeks
{
public:
    void func(int x)
    {
        cout<<"Value of x is "<<x<<endl;
    }
    void func(double x)
    {
        cout<<"Value of x is "<<x<<endl;
    }
    void func(int x, int y)
    {
        cout<<"Value of x and y is "<<x<<" "<<y<<endl;
    }
};
int main()
{
    Geeks obj1;
    obj1.func(7);
    obj1.func(9.123);
    obj1.func(85,64);
    return 0;
}

```

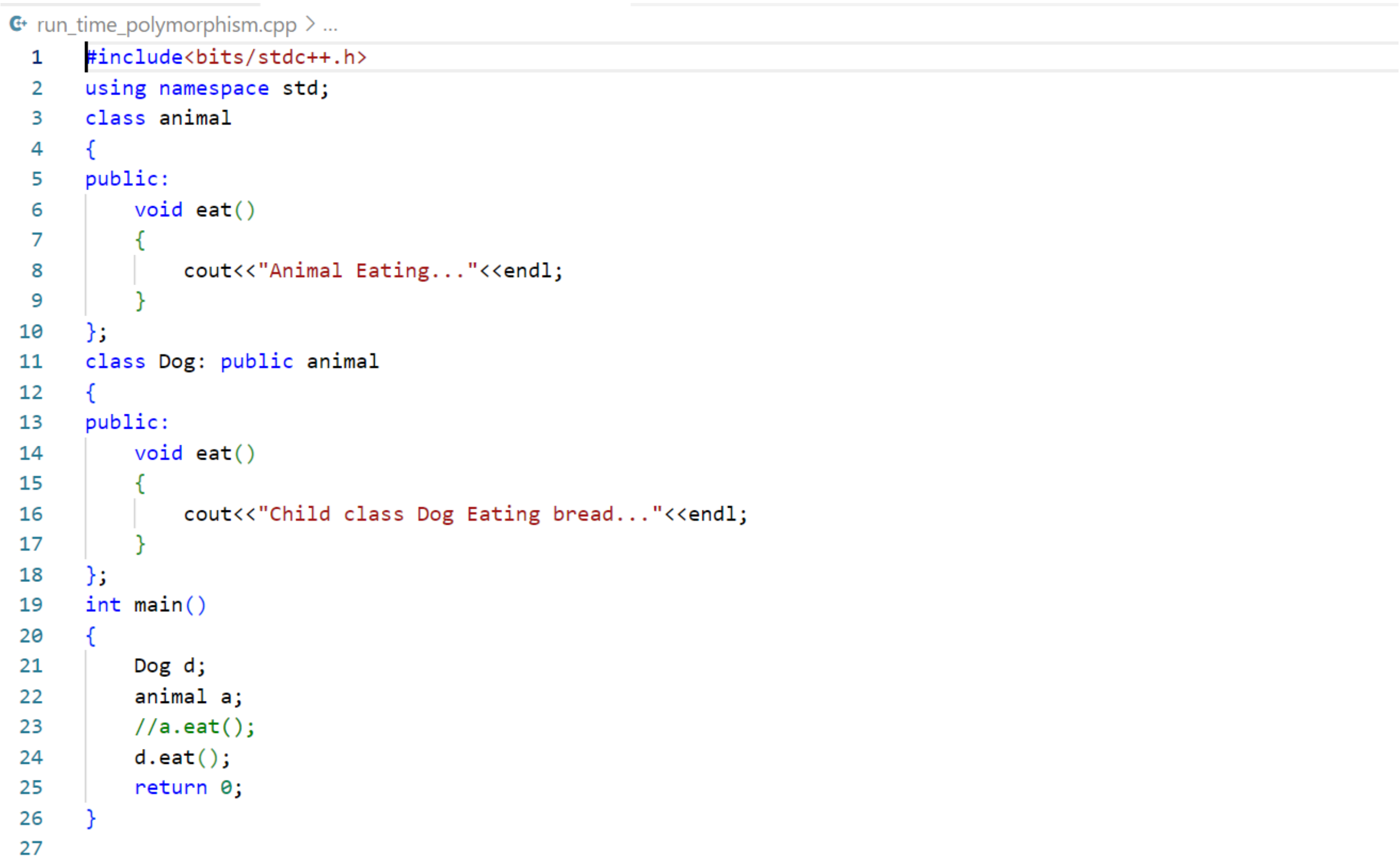
OUTPUT:



```
Value of x is 7
Value of x is 9.123
Value of x and y is 85 64
PS E:\Code\CSE 4192\Class\Class-11>
```

05.Write a C++ program using run time polymorphism.

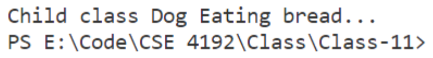
CODE:



```cpp
#include<bits/stdc++.h>
using namespace std;
class animal
{
public:
    void eat()
    {
        cout<<"Animal Eating..."<<endl;
    }
};
class Dog: public animal
{
public:
    void eat()
    {
        cout<<"Child class Dog Eating bread..."<<endl;
    }
};
int main()
{
    Dog d;
    animal a;
    //a.eat();
    d.eat();
    return 0;
}

```

OUTPUT:



```
Child class Dog Eating bread...
PS E:\Code\CSE 4192\Class\Class-11>
```