



National University of Computer & Emerging Sciences, Karachi
Computer Science Department
Fall 2021, Lab Manual – 04



Course Code: CL-217	Course : Object Oriented Programming Lab
----------------------------	---

Lab # 04

Contents:

- Introduction to Constructor
- Types of Constructors
- Destructor

INTRODUCTION TO Constructor

Constructor and its Types

- **Constructor** is the special type of member function in C++ classes, which are automatically invoked when an object is being created. It is special because its name is same as the class name.
- **To initialize data member of class:** In the constructor member function (which will be declared by the programmer) we can initialize the default values to the data members and they can be used further for processing.
- **To allocate memory for data member:** Constructor can also be used to declare run time memory (dynamic memory for the data members).
- Constructor has the same name as the class name. It is case sensitive.
- Constructor does not have return type.
- We can overload constructor, it means we can create more than one constructor of class.
- It must be public type.

Types of Constructors

- **Default Constructors:** Default constructor is the constructor which doesn't take any argument. It has no parameters.
- **Parameterized Constructors:** It is possible to pass arguments to constructors. Typically, these arguments help initialize an object when it is created. To create a parameterized constructor, simply add parameters to it the way you would to any other function. When you define the constructor's body, use the parameters to initialize the object.
- **Copy Constructor:** A copy constructor is a member function which initializes an object using another object of the same class. The copy constructor in C++ is used to copy data of one object to another.

Destructors

- A destructor is a special member function that works just opposite to constructor, unlike constructors that are used for initializing an object, destructors destroy (or delete) the object.
- Destructor function is automatically invoked when the objects are destroyed.
- It cannot be declared static or const.
- The destructor does not have arguments.
- It has no return type not even void.
- An object of a class with a Destructor cannot become a member of the union.
- A destructor should be declared in the public section of the class.
- The programmer cannot access the address of destructor.

Data Members & its Types

- **Data Members:** The variables which are declared in any class by using any fundamental data types (like int, char, float etc) or derived data type (like class, structure, pointer etc.) are known as Data Members.

Types of Data Members

- **Private members:** The members which are declared in private section of the class (using private access modifier) are known as private members. Private members can also be accessible within the same class in which they are declared.
- **Public members:** The members which are declared in public section of the class (using public access modifier) are known as public members. Public members can access within the class and outside of the class by using the object name of the class in which they are declared.

Sample C++ Code:

Code#1 (Default Constructors)

```
#include <iostream>
using namespace std;

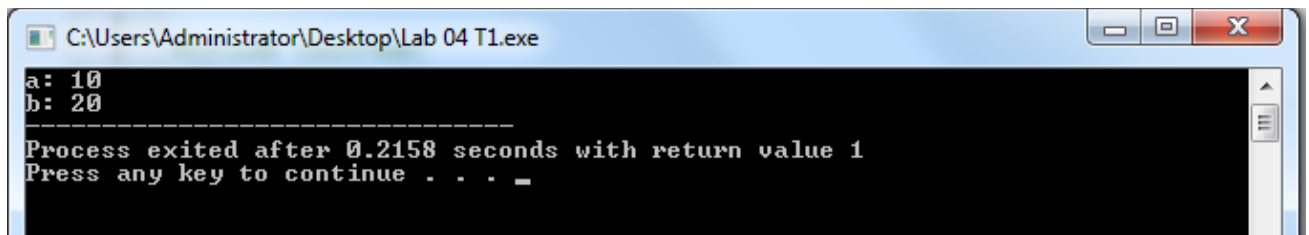
class construct
{
public:
    int a, b;

    construct()
    {
        a = 10;
        b = 20;
    }
};

int main()
{

    construct c;
    cout << "a: " << c.a << endl
        << "b: " << c.b;
```

```
    return 1;  
}
```



```
C:\Users\Administrator\Desktop\Lab 04 T1.exe  
a: 10  
b: 20  
-----  
Process exited after 0.2158 seconds with return value 1  
Press any key to continue . . . _
```

Code#2 (Parameterized Constructors)

```
#include <iostream>  
using namespace std;
```

```
class Point  
{  
private:  
    int x, y;
```

```
public:
```

```
    Point(int x1, int y1)  
    {  
        x = x1;  
        y = y1;  
    }
```

```
    int getX()  
    {  
        return x;  
    }  
    int getY()  
    {  
        return y;  
    }
```

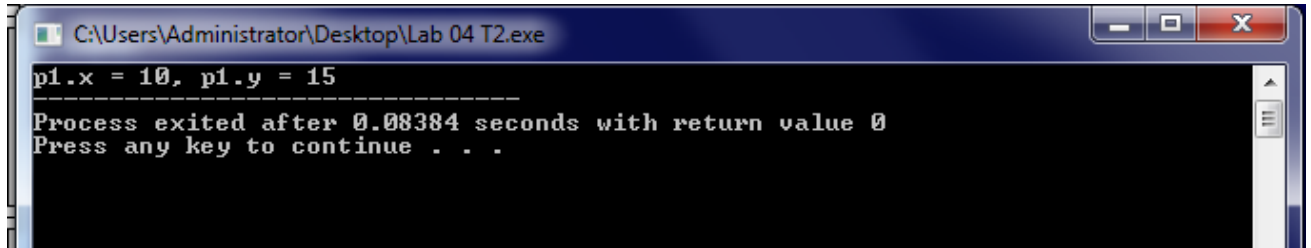
```
};
```

```
int main()  
{
```

```
    Point p1(10, 15);
```

```
    cout << "p1.x = " << p1.getX() << ", p1.y = " << p1.getY();

    return 0;
}
```



Code#3 (Copy Constructor)

```
#include<iostream>
#include<conio.h>

using namespace std;

class Example {

    int a, b;
public:

    Example(int x, int y) {

        a = x;
        b = y;
        cout << "\nIm Constructor";
    }

    Example(const Example& obj) {

        a = obj.a;
        b = obj.b;
        cout << "\nIm Copy Constructor";
    }
}
```

```

void Display() {
    cout << "\nValues :" << a << "\t" << b;
}

};

int main() {

    Example Object(10, 20);

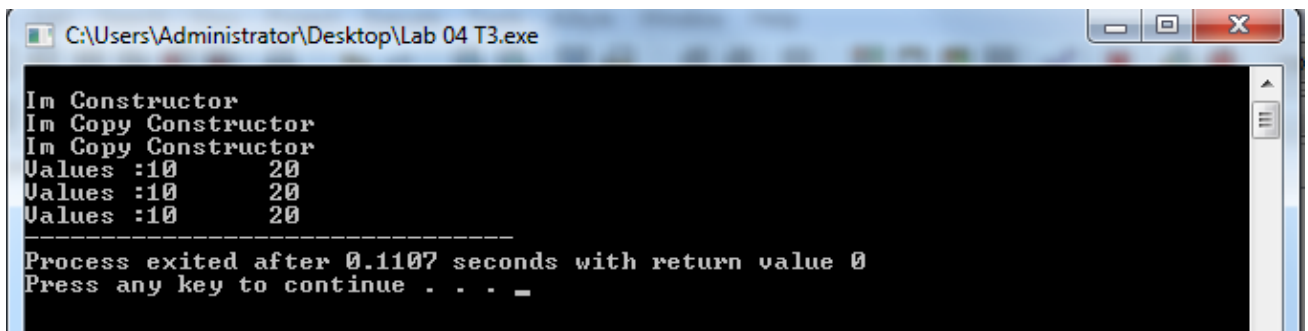
    Example Object2(Object);

    Example Object3 = Object;

    Object.Display();
    Object2.Display();
    Object3.Display();

    return 0;
}

```



```

C:\Users\Administrator\Desktop\Lab 04 T3.exe
Im Constructor
Im Copy Constructor
Im Copy Constructor
Values :10    20
Values :10    20
Values :10    20
-----
Process exited after 0.1107 seconds with return value 0
Press any key to continue . . . _

```

A. Code#4 (Destructors)

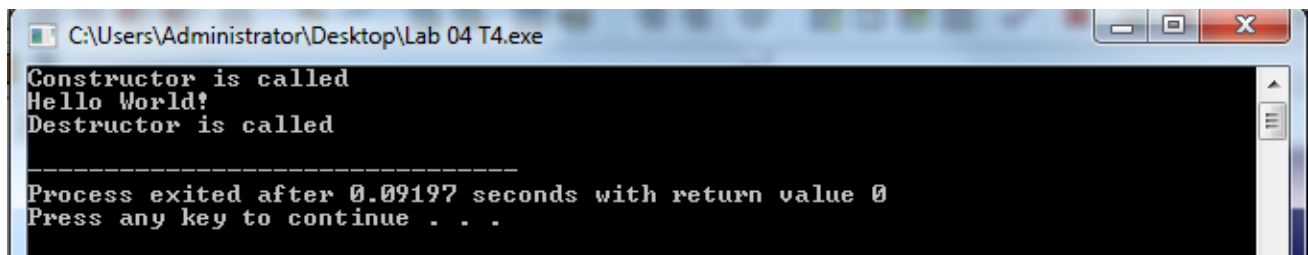
```

#include <iostream>
using namespace std;
class HelloWorld{
public:

    HelloWorld(){
        cout<<"Constructor is called"<<endl;
    }
}

```

```
~HelloWorld(){  
    cout<<"Destructor is called"<<endl;  
}  
  
void display(){  
    cout<<"Hello World!"<<endl;  
}  
};  
int main(){  
  
    HelloWorld obj;  
  
    obj.display();  
    return 0;  
}
```



```
C:\Users\Administrator\Desktop\Lab 04 T4.exe  
Constructor is called  
Hello World!  
Destructor is called  
-----  
Process exited after 0.09197 seconds with return value 0  
Press any key to continue . . .
```

Lab Tasks

Question # 01:

Create a class called **Triangle** which has the following data members:

- Base
- Height

Create the following methods for your class:

- Constructors –default, parametrized and copy constructors
- Getters and setters for Base and Height
- A method called **calculateArea** which returns the area of the triangle (area = height*base/2)

Create two triangles and then test out if your copy constructor is working properly.

Question # 02:

Create a class named “**counterType**” which contains the following data members:

- counterCurrent
- counterPrevious
- counterNext

It should have accessor and mutator functions for each of its data members. Furthermore, it has the following two methods:

- incrementCounter – increments the value of counterCurrent and updates the other two members accordingly
- decrementCounter – decrements the value of counterCurrent and updates the other two members accordingly
- A destructor that resets all of the values of the counters to 0. It must also print that it has done so after changing the values.

Create a default constructor and a parametrized constructor which takes only one value as a parameter. It then updates the data members accordingly.

A few constraints to keep in mind: counterPrevious should always be equal to counterCurrent – 1, similarly, counterNext should always be equal to counterCurrent + 1.

The counter values should never be less than 0.

Question # 03:

A Movie CD rental shop maintains records of all the CDs it owns. A CD record normally stores information about the name of the Movie, the producers, the studio information. Normally a rental shop has more than one copy of a movie. Therefore, there is a need to store how many copies are currently available for rent at a given time. Similarly, there needs to be a record for how many CDs have been rented out for a particular movie.

Create a menu-driven program that allows the rental shop owners to:

- Check the number of CDs available for rent for a particular movie
- Check the number of CDs currently on rent for a particular movie
- Rent a movie CD to a customer
- Receive a movie CD from a customer

Note: The methods should increment or decrement the CDs rented/available accordingly.

Homework

Question # 04:

Create a class called **Member**.

- Each object of **Member** can hold the name of a person, member ID, number of CDs rented, and amount spent.
- Include the member functions to perform the various operations on the objects of **Member** — for example, modify, set, and show a person's name. Similarly, update, modify, and show the number of movies rented and the amount spent.
- Add the appropriate constructors.
- Write the definitions of the member functions of **Member**.
- Write a program to test various operations of your class **Member**. (You can hard code these, no need to be menu driven)

Question # 05:

Using the classes designed in Lab Task 03 and 04, write a program to simulate a Movie Rental Store. The store has two types of customers.

- Members
- Non-members

Members receive a 5% discount on CDs that they rent; however, non-members do not get any discount. Use the class **Member** to keep track of the books that a member purchases (You can set some initial values through parametrized constructors)

When renting, every 5th CD rented by a member gets a discount of an additional flat 20\$

Create a menu-driven program that helps the store owner manage the CD details, and calculate the bill appropriately for members and non-members.