NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCE

CL 1004 - Object Oriented Programming Lab

Semester: Spring 2022

Lab 04

CONSTRUCTORS - DESTRUCTORS

Instructor: Muhammad Sudais

Email: muhammad.sudais.v@nu.edu.pk

Outline

- Default Constructor
- Parameterized Constructor
- Copy Constructor
- Destructor

CONSTRUCTORS

A constructor is a special function that is a member of a class.

- C++ requires a constructor call for each object that's created, which helps ensure that each object is initialized properly before it's used in a program. The constructor call occurs implicitly when the object is created. This ensures that objects will always have valid data to work on.
- Normally, constructors are declared public.
- Constructors can be overloaded, just like other functions.

DIFFERENCE BETWEEN CONSTRUCTORS AND OTHER FUNCTIONS

CONSTRUCTOR	FUNCTION
Can NOT return any value	Can return value
Constructor name is always class name	Function name can NOT be class name

DEFAULT CONSTRUCTOR

- A constructor without parameters is referred to as a default constructor.
- If a class does not contain a constructor definition, the compiler will create a minimal version of the default constructor as a public member. However, this constructor will not perform initialization. An uninitialized variable typically contains a "garbage" value.

By contrast, if a class contains at least one constructor, a default constructor must be defined explicitly.

See following example:

```
#include<iostream> #include<string>
using namespace std;
class Employee
{  int salary;
string name;
public:
Employee()
   {
      salary = 60000;
            name = "ABCD";
   }
void display()
{
               cout<<"name  :"<<name<<"  "<<"salary  :"<<salary<<endl;
}
};
 int main()
{
Employee e1;
e1.display();
return 0;
}
```

Parameterized Constructor:

The following example demonstrates how we can pass argument in a constructor.

Example 1:

```
#include<iostream>
#include<string>
using namespace std;
class Employee  {  double salary;
public:
Employee(double annualSalary)
   {
      salary = annualSalary;
   }
void display()
{
                cout<<"salary  :"<<salary<<endl;
}
};
 int main()
{
   Employee e1(19876);      e1.display();
return 0;
   }
```

In the following example, we have used constructor with 2 arguments to initialize class object.

Example 2:

```
#include<iostream>
using namespace std;
class CoOrds
{
public:
int x, y;
// constructor with two arguments:
public:
CoOrds(int x, int y)
   {
(*this).x = x;
(*this).y = y;
   }
 void display()
   {
cout<<"x = {0}, y = {1} : "<<x<<"  "<<y<<endl;
   }
};

int main()
{
CoOrds p1(5, 3);
p1.display();
}
```

MULTIPLE CONSTRUCTORS

When a class have two or more constructors, the concept is said to be is said to be overloaded. Any class member function may be overloaded, including the constructor. For example, one constructor might take an integer argument, while another constructor takes a double.

Example:

```
#include<iostream>
using namespace std;
class CoOrds
{
public:
int x, y;
// constructor with two arguments:
public:
CoOrds(int x, int y)
    {
(*this).x = x;
(*this).y = y;
    }
   // constructor with one arguments:
CoOrds(int cor)
    {        x = cor;
y =
cor;
    }

 void display()
    {
cout<<"x = {0}, y = {1} : "<<x<<"  "<<y<<endl;
    }
};

int main()
{           CoOrds p(15);
            p.display();
               CoOrds p1(5, 3);
               p1.display();

}
```

COPY CONSTRUCTOR:

The copy constructor is a constructor which creates an object by initializing it with an object of the same class, which has been created previously. The copy constructor is used to

- Initialize one object from another of the same type.
- Copy an object to pass it as an argument to a function.
- Copy an object to return it from a function.

Syntax:

```
classname (const classname  &obj)
```

```
{
   // body of constructor }
```

If a copy constructor is not defined in a class, the compiler itself defines one.

Example 1:

```
#include <iostream>

using namespace std;

class oneD {
private:       int i;
public:        int
getLength(  )
       {
         return i;
        }
            // simple constructor
      oneD ( int len ){
   cout << "Normal constructor allocating i" << endl;

   i = len;
             }
              // copy constructor
      oneD (oneD &obj)
      {
   cout << "Copy constructor allocating i" << endl;

   i = obj.i; // copy the value
       }
      // destructor

~ oneD ()
  {
   cout << "Freeing memory!" << endl;

}
};

 void display(oneD obj) {
   cout << "Length of line : " << obj.getLength() <<endl; }

int main() {

   oneD a(10);

  oneD b=a;
  display(a);  // This also calls copy constructor
display(b);
return 0;}
```

If the class has pointer variables and has some dynamic memory allocations, then it is a must to have a copy constructor. **Example 2:**

```
#include <iostream>

using namespace std;

class oneD {
private:
         int *ptr;
         public:

oneD (int len) {    cout << "Normal constructor
allocating ptr" << endl;

  // allocate memory for the pointer;
ptr = new int;
  *ptr = len;
}



oneD (const oneD &obj) {    cout << "Copy
constructor allocating ptr." << endl;    ptr = new
int;    *ptr = *obj.ptr; // copy the value  }

~ oneD () {    cout << "Freeing
memory!" << endl;    delete ptr;
}

int getLength( ) {
return *ptr;
}
};
void display(oneD obj) {    cout << "Length of line : " <<
obj.getLength() <<endl; }



int main() {
oneD a(10);    oneD
b=a;    display(a);
display(b);

  return 0;
}
```

DESTRUCTOR

A destructor is a special function that is a member of a class.

- A destructor is a special member function that is called when the lifetime of an object ends.
- The purpose of the destructor is to free the resources that the object may have acquired during its lifetime.
- Normally, destructors are declared public.

- Destructor have the same name as class name followed by a ~. Called when an object is destroyed.

Example:

```
class Example
 {
          public:
          Example()
{
cout<< “Welcome”;
}

          ~Example()
          {

                    cout<<”Good Bye”;
          }
};
```

EXERCISE

QUESTION # 1

Create a class named “Lab” that has the following members:

- Int LabNumber
- Int NoOfTasks
- String Topic
- Float Attendance s

Create its Constructor and Destructor

QUESTION # 2

Write a program in which a class named Person has member variables Name, Height, Age and Interests (Array). Use a parameterized constructor to initialize the variables with your details. Print all data calling some public method. Call the destructor then.

QUESTION # 3

An address, such as House No 123, Block A, Sector 17, Karachi, Pakistan. can be thought of as having five parts: the house number (123) the block (A), sector (17),city (Karachi), and the Country (8214). Write a program that uses a class Address to store these parts of an address in specific attributes. Add a constructor that accept a string address and separate these elements from that string. Write a display function that display the details of the street.

Sample Program output:
Please Enter Your Address: House No 123, Block A, Sector 17, Karachi, Pakistan
House Number : 123
Block: A
Sector: 17
City: Karachi
Country: Pakistan

QUESTION # 4

Create a class Circle with attributes radius and pie, which has a default constructor setting value to 2 and 3.14 respectively. Also provide a 2 argument constructor which set the value of pie to 3.14 and radius to the value in argument. Provide constructor with single Boolean argument which when true calculate the circumference and the area of the circle and stores it in a new attribute of class Circle named Circumference and does nothing if false.

Cirle()

Circle (radius, pie)

Circle (bool circum)

QUESTION# 5

Create a class Car with 3 private variables CarID of type integer, CarName of type string , and Model of type integer.

- Use a default constructor to initialize all variables with any values.
- Use a constructor to take user input in all variables to display data.
- Use a parameterized constructor to initialize the variables with values of your choice.
- Use a destructor