# Department of Electrical Engineering

**CS212**

**Object Oriented Programming**



# Lab 6: Constructors and Destructors

**Class**: BEE - 12C

**Date**: November 1st, 2021

**Time**: Monday (1400 – 1700)

**Name**: Muhammad Umer

**CMS ID**: 345834

**Learning Objectives**

* Understand the importance of access specifiers, constructors and destructors.
* Understand the difference between a parameterized constructor and non-parameterized  
  constructor
* Develop a default copy constructor and facilitate the copying of data from one object  
  to the other.

**Tools**

* Microsoft Visual Studio 2013

**Constructors**

A constructor is a function that is automatically called when an object is created. This function can exhibit the regular behavior of any function except that it cannot return a value. The reason why constructors are needed is that unlike regular functions which need to deliberately called, a constructor will be automatically called when an object is created. Every constructor has a body from where we can call regular member functions.

A very important question which is often asked is that how does the compiler know that the constructor function needs to be called automatically? The answer is very simple. A constructor is a function that has the same name as the class. Whenever an object is created the compiler searches for a function having the same name as the class i.e. the constructor. Given below is a sample code that shows the class constructor. Generally, the constructor is defined as *public*. Also, the constructor can be overloaded like a regular member function. An important point regarding a constructor is that it cannot return a value. In fact, writing the keyword void is strictly prohibited.



**Destructors:**Constructors are designed to help initialize/ create an object. Destructors on the other hand do  
exactly the opposite. Destructors are called whenever an object goes out of scope. When this  
happens, **it is necessary to perform cleanup procedures especially when you have used  
dynamic memory or you have been working with pointers in your code**. The destructor  
function can be used to free up memory that you have allocated or dereference pointers that were referenced. The rules for a destructor are as follows:



**Activity**Your task is to carefully understand the following code. Analyze the code and suggest any  
corrections that may be needed. There are both syntax and logical errors in the code so consider both when designing the correct solution. Submit the correct code to the lab instructor.

#include <iostream>

using namespace std;

class Student // 'S' of class name should be Capital

    {

    int age, cnic, semester;

    char name;

public:

    void setall(int a, int c, int s, char n) // No return type so void is used

    {

        age = a;

        cnic = c; // Variable should be assigned value rather than an input

        semester = s;

        name = n;

    }

    void setage(int a) // Set age, previously, didn't exist

    {

        age = a;

    }

    void displayall();

}; // Missing semicolon in Class's syntax

void Student::displayall()

{

    cout << "Student's Information\n";

// No data member called student.data, alternative approach used

    cout << "\tAge: " << age << endl;

    cout << "\tSemester: " << semester << endl;

    cout << "\tName: " << name << endl;

}

int main()

{

    Student obj;

    obj.setall(16, 3740617872917, 3, 'Umer');

// Previously missing arguments/ too few arguments

    obj.displayall();

    obj.setage(19); // No function by this name exists

    Student anotherobj;

    anotherobj.setall(18, 3740617872913, 3, 'Ali');

// Previously, invalid usage of scope resolution operator

    return 0;

}

**Lab Tasks**

**Q1: Write a C++ program that creates a program for a new ice cream vendor called Ben&Jerry. The management of Ben & Jerry has decided that they are going to sell their ice cream in 7 different flavors namely chocolate, vanilla, strawberry, mango, tutti fruit, almond crunch and coffee.**

**Carefully design the program by observing the following rules.**

* Ben&Jerry is charging Rs 200 for two scoops and Rs 250 for three scoops. Hence you will need a function to determine the number of scoops and based on that the price. If a user enters more than three scoops your program should display invalid input and it should exit.
* Ben&Jerry allows its customers to purchase a vanilla wafer with their ice cream. If the  
  customer wants to purchase the wafer, he will have to pay an additional Rs 50. This amount should be added to the total amount payable by the user.
* If the customer asks for chocolate flavor, then he will have to pay an additional amount i.e. Rs 220 for two scoops and Rs 280 for three scopes. Design a function that will be called if the customer chooses flavored ice cream.
* The program should show a menu that asks the customer for his requirements and then  
  displays the final payable amount with full details about the flavor, number of scoops and wafer
* In the end create a class destructor that displays a thank you message to the user.

Design your program using sound OOP practices. Carefully determine the data members, member functions, access specifiers, activities to be performed in the constructor. Make sure that you use good naming conventions in your code.



#include <iostream>

#include <string>

using namespace std;

class BenAndJerry {

 private:

  string flavour;

  string wafer\_opt;

  int n\_scoops;

  float price, wafer\_price = 50;

 public:

  void getChoice() {

    cout << "Enter your flavour choice: ";

    getline(cin, flavour);

    //////////////////////////////

    if (flavour != "Chocolate" && flavour != "Vanilla" &&

        flavour != "Strawbaerry" && flavour != "Tutti Fruiti" &&

        flavour != "Mango" && flavour != "Almong Crunch" &&

        flavour != "Coffee") {

      cout << "Invalid Choice!";

      exit(0);

    }

    //////////////////////////////

    cout << "Enter your wafer choice: ";

    getline(cin, wafer\_opt);

    if (wafer\_opt != "Yes" && wafer\_opt != "No") {

      cout << "Invalid Choice!";

      exit(0);

    }

    //////////////////////////////

    cout << "How many scoops would you like? ";

    cin >> n\_scoops;

    if (n\_scoops != 2 && n\_scoops != 3) {

      cout << "Invalid Choice!";

      exit(0);

    }

  }

  void displayChoice() {

    cout << "\nYour choice of Ice Cream is as follows: " << endl;

    cout << "\nFlavour = " << flavour;

    cout << "\nNumber of Scoops = " << n\_scoops;

    cout << "\nWafer = " << wafer\_opt;

    cout << "\nTotal price is " << priceCalculator();

    cout << endl;

  }

  int priceCalculator() {

    price = 0;

    if (flavour == "Chocolate") {

      if (n\_scoops == 2) {

        price = 220;

      }

      if (n\_scoops == 3) {

        price = 280;

      }

    }

    else {

      if (n\_scoops == 2) {

        price = 200;

      }

      if (n\_scoops == 3) {

        price = 250;

      }

    }

    if (wafer\_opt == "Yes") {

      price += wafer\_price;

    }

    return price;

  }

  void displayMenu() {

    cout << "\t\tWelcome to Ben & Jerry!" << endl;

    cout << "\tFill and shape out your Ice Cream!" << endl;

  }

  ~BenAndJerry() { cout << "Thank you for visiting Ben & Jerry!"; }

};

int main() {

  BenAndJerry cust;

  cust.displayMenu();

  cust.getChoice();

  cust.displayChoice();

  return 0;

}

**Terminal Output**

PS D:\NUST\Semester 3\Object Oriented Programming\Labs\Lab 6\lab6.exe

Welcome to Ben & Jerry!

Fill and shape out your Ice Cream!

Enter your flavour choice: Vanilla

Enter your wafer choice: No

How many scoops would you like? 1

Invalid Choice!

PS D:\NUST\Semester 3\Object Oriented Programming\Labs\Lab 6\lab6.exe

Welcome to Ben & Jerry!

Fill and shape out your Ice Cream!

Enter your flavour choice: Chocolate

Enter your wafer choice: Yes

How many scoops would you like? 3

Your choice of Ice Cream is as follows:

Flavour = Chocolate

Number of Scoops = 3

Wafer = Yes

Total price is 330

Thank you for visiting Ben & Jerry!

**Q1: VISION is a world leader in manufacturing LCD Televisions. The company has decided that it will allow its customers to give the dimensions of the TV (in length and width). Once the length and width are ordered the company will manufacture the TV according to your requirements. In this regard they want you to create a program that will assist them. Carefully read all the instructions and follow the requirements.**

* Create a class called vision
* Create three constructors as follows
* A non-parameterized constructor that calls the setlength() and setwidth() function
* A parameterized constructor that will receive the length and width as integers
* A parameterized constructor that will receive the length and width in float
* By using a special function calculate the area of the TV
* Create a function to calculate the price of the TV by multiplying the area with Rs. 65.
* Create a display() function to show the details of the purchased TV.
* Use the default copy constructor to show that copying of simple objects can be accomplished using the default copy constructor.

In the main, you will construct four objects that demonstrate the use of the four constructors. After calling the constructor it will take over and will handover control to the area function, and then the price calculation function. Remember that the user should not have access to modifying the price.

#include <iostream>

#include <string>

using namespace std;

class Vision {

 private:

  double length, width, price, area;

 public:

  void setLength() {

    cout << "Input your desired TV length: ";

    cin >> length;

  }

  void setWidth() {

    cout << "Input your desired TV width: ";

    cin >> width;

  }

  void display(int a) {

    cout << "\nThe details of order # " << a << " are: " << endl;

    cout << "Length: " << length << endl;

    cout << "Width: " << width << endl;

    cout << "Price: " << priceCalculator() << " Rs" << endl;

  }

  int areaCalculator() {

    area = length \* width;

    return area;

  }

  int priceCalculator() {

    price = areaCalculator() \* 65;

    return price;

  }

  Vision(int a, int b) {

    length = a;

    width = b;

  }

  Vision(float a, float b) {

    length = a;

    width = b;

  }

  Vision() {

    setLength();

    setWidth();

  }

};

int main() {

  cout << "Welcome to VISION TV Portal!\n";

  Vision t1;

  t1.display(1);

  Vision t2(int(25), int(32));

  t2.display(2);

  Vision t3(float(27.5), float(32.75));

  t3.display(3);

  Vision t4(t3);

  t4.display(4);

  return 0;

}

**Terminal Output**

PS D:\NUST\Semester 3\Object Oriented Programming\Labs\Lab 6\lab6.exe

Welcome to VISION TV Portal!

Input your desired TV length: 32

Input your desired TV width: 45

The details of order # 1 are:

Length: 32

Width: 45

Price: 93600 Rs

The details of order # 2 are:

Length: 25

Width: 32

Price: 52000 Rs

The details of order # 3 are:

Length: 27.5

Width: 32.75

Price: 58500 Rs

The details of order # 4 are:

Length: 27.5

Width: 32.75

Price: 58500 Rs