4/3/2025

OBJECT ORIENTED PROGRAMMING

ASSIGNMENT#3

NAME: AHTISHAM KHAN
ENROLLMENT: 09-131242-014
DEPARTMENT: BSE-2B
COURSE TEACHER: FAISAL ZIA

Lab # 3

Classes using Constructor and Destructor

Objective:

The objective of this lab is to understand the concept of constructors and destructors in C++. By the end of this lab, you should be able to:

1. Define and use constructors to initialize objects.
2. Understand the difference between default and parameterized constructors.
3. Use destructors to clean up resources when an object is destroyed.
4. Apply constructors and destructors in real-world scenarios like creating a student database.

---

1. Constructors

- A constructor is a special member function of a class that is automatically called when an object of the class is created.
- It has the same name as the class and no return type (not even void).
- Constructors are used to initialize the data members of an object.
- Types of Constructors:
  - Default Constructor: A constructor with no parameters.
  - Parameterized Constructor: A constructor that takes parameters to initialize the object with specific values.

2. Destructors

- A destructor is a special member function of a class that is automatically called when an object goes out of scope or is explicitly deleted.
- It has the same name as the class preceded by a tilde (~).
- Destructors are used to release resources (e.g., memory, file handles) allocated by the object.

Lab Activities:

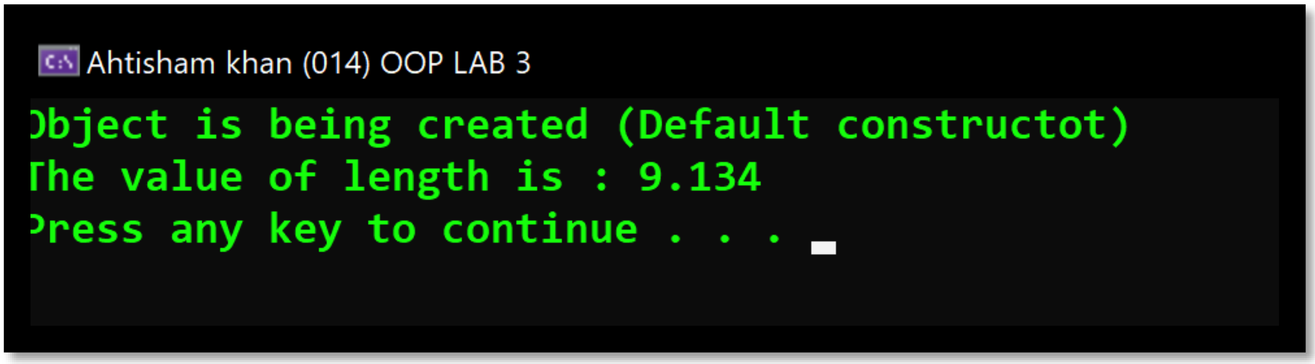
Activity 1: Default Constructor

Code:

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

class Line {
public:
   void setLength(double len);
   double getLength();
   Line(); // Default constructor

private:
   double length;
};

// Default constructor definition
Line::Line() {
   cout << "Object is being created (Default Constructor)" << endl;
   length = 0.0; // Initialize length to 0
}

void Line::setLength(double len) {
   length = len;
}

double Line::getLength() {
   return length;
}

int main() {
   Line line; // Object created, default constructor called

   // Set line length
   line.setLength(6.0);
   cout << "Length of line: " << line.getLength() << endl;

   return 0;
}
```

Your Output Here….

In case of output snippet please make sure output snippet contains student name and id. AliAhmed_123_Lab03_A1.exe



```
Object is being created (Default constructot)
The value of length is : 9.134
Press any key to continue . . .
```

Activity 2: Parameterized Constructor

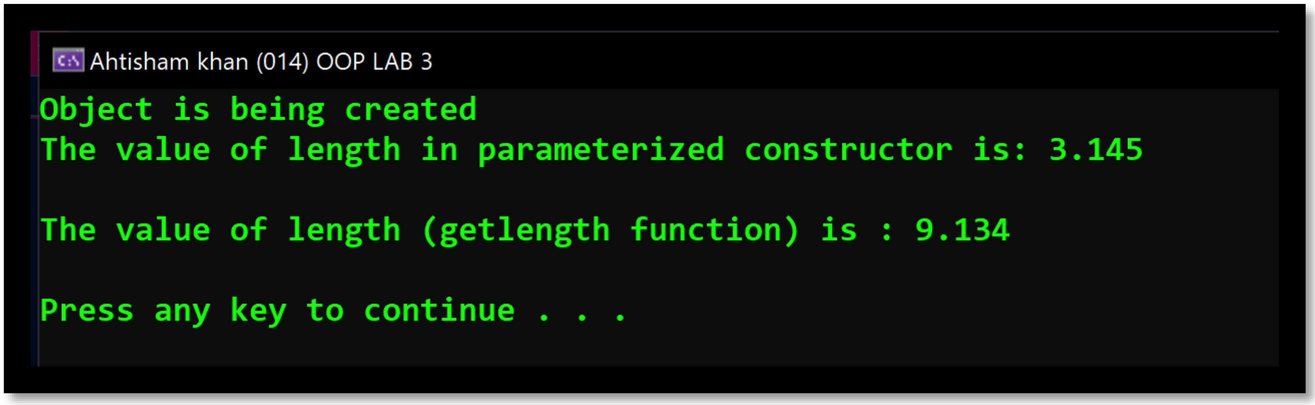
Code:

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

class Line {
public:
   void setLength(double len);
   double getLength();
   Line(double len); // Parameterized constructor

private:
   double length;
};

// Parameterized constructor definition
Line::Line(double len) {
   cout << "Object is being created, length = " << len << endl;
   length = len;
}

void Line::setLength(double len) {
   length = len;
}
```

```
double Line::getLength() {
   return length;
}

int main() {
   Line line(10.0); // Object created, parameterized constructor called

   // Get initially set length
   cout << "Length of line: " << line.getLength() << endl;

   // Set line length again
   line.setLength(6.0);
   cout << "Length of line: " << line.getLength() << endl;

   return 0;
}
```

Your Output Here….

`In case of output snippet please make sure output snippet contains student name and id. AliAhmed_123_Lab03_A2.exe`



```
Object is being created
The value of length in parameterized constructor is: 3.145

The value of length (getlength function) is : 9.134

Press any key to continue . . .
```

Activity 3: Student Database Using Constructor and Destructor

Write a C++ program to create a student database using a class. The program should store the following details:

1. Name of the student
2. Roll number of the student
3. Height of the student
4. Weight of the student

Use a constructor to initialize the data members and a destructor to display a message when the object is destroyed.

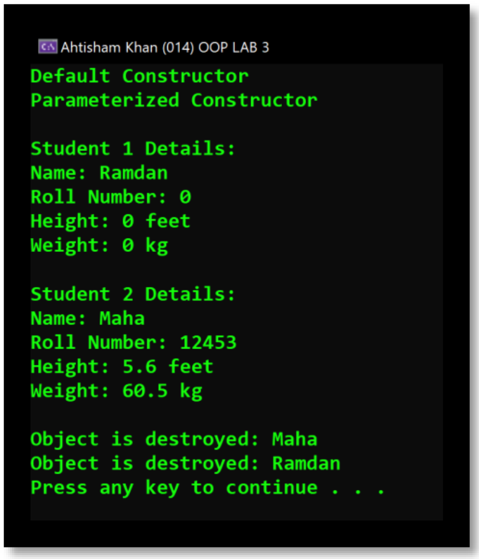
Code:

```
#include <iostream>
#include <cstring> // For strcpy
using namespace std;

class Student {
private:
    char name[25];
    int roll;
    float height, weight;

public:
    // Default constructor
    Student() {
        strcpy(name, "Ram");
        roll = 0;
        height = 0.0;
        weight = 0.0;
        cout << "Object created (Default Constructor)" << endl;
    }

    // Parameterized constructor
    Student(const char* n, int r, float h, float w) {
        strcpy(name, n);
        roll = r;
        height = h;
        weight = w;
        cout << "Object created (Parameterized Constructor)" << endl;
    }

    // Destructor
    ~Student() {
        cout << "Object destroyed for student: " << name << endl;
    }

    // Function to display student details
    void display() {
        cout << "\nName: " << name << endl;
        cout << "Roll No: " << roll << endl;
        cout << "Height: " << height << " feet" << endl;
        cout << "Weight: " << weight << " kg" << endl;
    }
};
```

```
int main() {
    // Create objects using default and parameterized constructors
    Student student1; // Default constructor
    Student student2("John", 101, 5.8, 65.5); // Parameterized constructor

    // Display student details
    cout << "\nStudent 1 Details:" << endl;
    student1.display();

    cout << "\nStudent 2 Details:" << endl;
    student2.display();

    return 0;
}
```

Your Output Here….

`In case of output snippet please make sure output snippet contains student name and id. AliAhmed_123_Lab03_A3.exe`



Ahtisham Khan (014) OOP LAB 3

```
Default Constructor
Parameterized Constructor

Student 1 Details:
Name: Ramdan
Roll Number: 0
Height: 0 feet
Weight: 0 kg

Student 2 Details:
Name: Maha
Roll Number: 12453
Height: 5.6 feet
Weight: 60.5 kg

Object is destroyed: Maha
Object is destroyed: Ramdan
Press any key to continue . . .
```

Conclusion:

In this lab, we learned:

1. How to define and use constructors to initialize objects.
2. The difference between default and parameterized constructors.
3. How to use destructors to clean up resources.
4. How to apply constructors and destructors in real-world scenarios like creating a student database.

Additional Notes:

- Constructors are essential for initializing objects with valid states.
- Destructors are crucial for releasing resources and preventing memory leaks.

- Always ensure that constructors and destructors are used appropriately in your programs.

Lab Tasks/ Homework

Task 01: Write a class Result that contains the following:

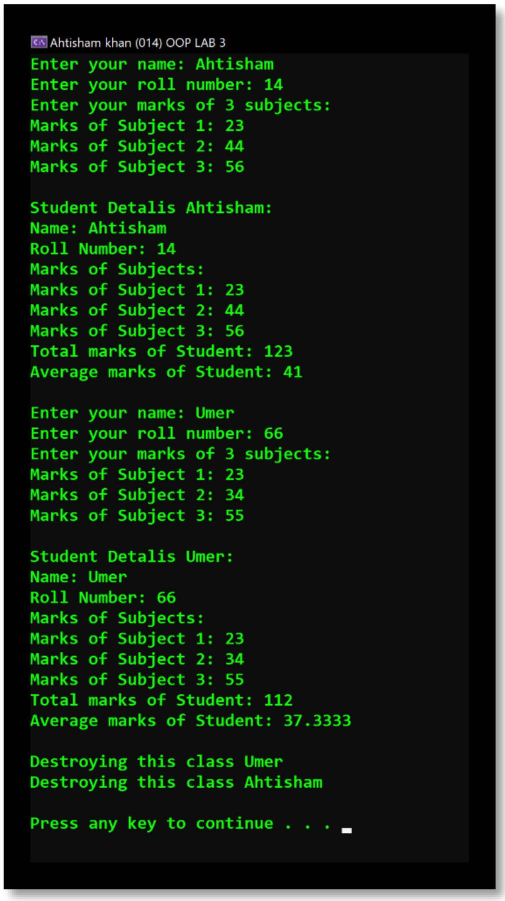
- Data members: rollNo, name, and marks (an array of integers for three subjects marks).
- Member functions:
    - parameterized constructor to initialize rollNo, name, and marks.
    - destructor to display a message when the object is destroyed.
    - void input(); // input values for rollNo, name, and marks.
    - void show() ; // display the values of rollNo, name, and marks.
    - int total(); // calculate and return the total marks of the student.
    - float avg(); // calculate and return the average marks of the student.

Solution:

```
Enter your code here…

#include <iostream>
using namespace std;
// class with name Result
class Result {
private:
    string Name;
    int rollnumber;
    int marks[3];

public:
    //Default constructor
    Result() {

    }
    //parameterized constructor
    Result(string na, int num, int arr[]) {
        Name = na;
        rollnumber = num;
        for (int i = 0; i < 3; i++) {
            marks[i] = arr[i];
        }
    }
    //Destructer
    ~Result() {
        cout << "Destroying this class "<< Name << endl;
    }
    //Function for entering values
    void entervalues() {
        cout << "Enter your name: ";
        cin >> Name;
        cout << "Enter your roll number: ";
        cin >> rollnumber;
        cout << "Enter your marks of 3 subjects: " << endl;
        for (int i = 0; i < 3; i++) {
            cout << "Marks of Subject " << i + 1 << ": ";
            cin >> marks[i];
        }
    }
    //Function for displaying values
    void displayvalues() {
        cout << "\nStudent Detalis "<<Name<<":" << endl;
        cout << "Name: " << Name << endl;
        cout << "Roll Number: " << rollnumber << endl;
        cout << "Marks of Subjects: " << endl;
        for (int i = 0; i < 3; i++) {
            cout << "Marks of Subject " << i + 1 << ": " << marks[i]
<< endl;
        }
    }
    //function to calculate total sum
    int total() {
        int total=0;
        for (int i = 0;i < 3;i++) {
            total += marks[i];
        }
        return total;
    }
    //function to calculate total average
    float avg() {
        return (float) total ()/ 3;
    }
};

int main() {
    system("title Ahtisham khan (014) OOP LAB 3");
    {
        //object 1
        Result s1;
        s1.entervalues();
        s1.displayvalues();
        cout << "Total marks of Student: " << s1.total() << endl;
        cout << "Average marks of Student: " << s1.avg() << endl;
        cout << endl;
        //object 2
        Result s2;
        s2.entervalues();
        s2.displayvalues();
        cout << "Total marks of Student: " << s2.total() << endl;
        cout << "Average marks of Student: " <<s2.avg()<< endl;
        cout << endl;
    }
    cout << endl;
    system("pause");
    return 0;
}
```

Output:

Your Output here…



```
Enter your name: Ahtisham
Enter your roll number: 14
Enter your marks of 3 subjects:
Marks of Subject 1: 23
Marks of Subject 2: 44
Marks of Subject 3: 56

Student Detalis Ahtisham:
Name: Ahtisham
Roll Number: 14
Marks of Subjects:
Marks of Subject 1: 23
Marks of Subject 2: 44
Marks of Subject 3: 56
Total marks of Student: 123
Average marks of Student: 41

Enter your name: Umer
Enter your roll number: 66
Enter your marks of 3 subjects:
Marks of Subject 1: 23
Marks of Subject 2: 34
Marks of Subject 3: 55

Student Detalis Umer:
Name: Umer
Roll Number: 66
Marks of Subjects:
Marks of Subject 1: 23
Marks of Subject 2: 34
Marks of Subject 3: 55
Total marks of Student: 112
Average marks of Student: 37.3333

Destroying this class Umer
Destroying this class Ahtisham

Press any key to continue . . .
```

Task 02: Create a class Rectangle with the following:

- Data members: length and width (both default to 1).
- Member functions:
    - parameterized constructor to initialize length and width.
    - destructor to display a message when the object is destroyed.
    - void setLength(float l); // To set the length (verify that it is between 0.0 and 20.0).
    - void setWidth(float w); // To set the width (verify that it is between 0.0 and 20.0).
    - float getLength(): // To return the length.
    - float getWidth(): // To return the width.
    - float perimeter(): // To calculate and return the perimeter of the rectangle.
    - float area(): // To calculate and return the area of the rectangle.

Solution:

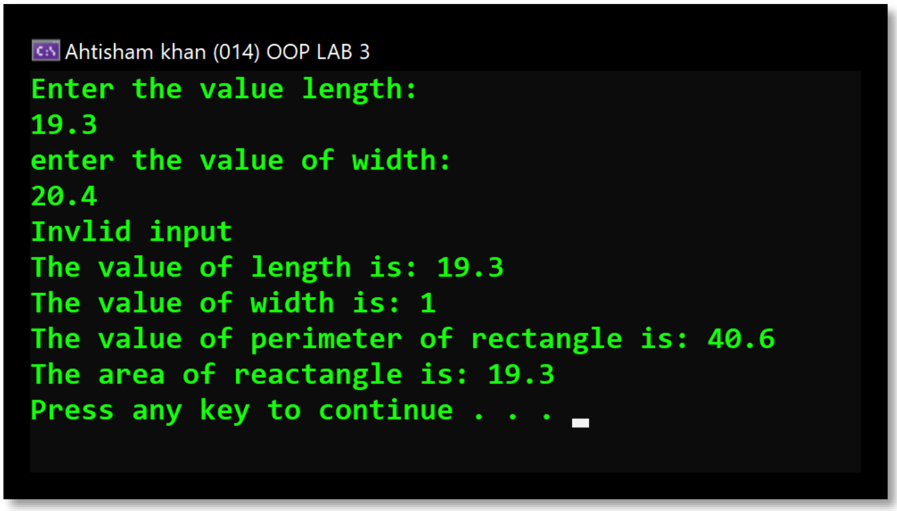
`Enter your code here…`

```
#include<iostream>
using namespace std;
class Rectangle {
private:
        float length;
        float width;
public:
        //Dedault constructor
        Rectangle() {
                length = 1;
                width = 1;
        }
        //parameterized constructor
        Rectangle(float l,float w) {
                length = l;
                width = w;
        }
        //function to get length
        void enterlength() {
                //temporary variable to store value of length and check the conditions if wrong then compiler use default value that we assign is 1
                float len;
                cout << "Enter the value length: " << endl;
                cin >> len;
                if (len>= 0.0 && len<=20.0) {
                  //length = len
                    length = len;
                }
                else {
                    cout << "Invlid input" << endl;
                }
        }
        void enterwidth(){
                //temporary variable to store value of width and check the conditions if wrong then compiler use default value.
                float wid;
                cout << "enter the value of width: " << endl;
                cin >> wid;
                if (wid>=0.0 && wid<=20.0) {

                    width = wid;
                }
                else {
                    cout << "Invlid input" << endl;
                }

        }
        // display length
        float displaylength() {
                return length;
        }
        float displaywidth() {
                return width;
        }
        //finding perimeter
        float perimeter() {
                return 2 * (length + width);
        }
        //finding area
        float areaofrectangle() {
                return length * width;
        }
```

```
};
int main() {
        system("title Ahtisham khan (014) OOP LAB 3");
        {
                Rectangle R;
                R.enterlength();
                R.enterwidth();
                cout << "The value of length is: " << R.displaylength() << endl;
                cout << "The value of width is: " << R.displaywidth() << endl;
                cout << "The value of perimeter of rectangle is: " << R.perimeter() << endl;
                cout << "The area of reactangle is: " << R.areaofrectangle() << endl;
        }
        system("pause");
}
```

Output:

Your Output here…



```
Enter the value length:
19.3
enter the value of width:
20.4
Invlid input
The value of length is: 19.3
The value of width is: 1
The value of perimeter of rectangle is: 40.6
The area of reactangle is: 19.3
Press any key to continue . . .
```

`In case of output snippet please make sure output snippet contains student name and id. AliAhmed_123_Lab03_T2.exe`