Ayush Dadali Roll no: 54.
103223227 9

Page No. : ______
Date. : / /

OOC
Assignment No: 02.

Problem Statement:
Develop an object oriented program in C++ to create a database of student information systems containing the following information: Name, Roll number, Class, division, Date of Birth, Aadhar Number, Blood group, Contact address, Telephone number, etc.

Write the code to illustrate the use of default constructor, parameterized constructor, copy constructor and destructor.

Objectives:
1. To learn the concept of constructors and destructors in C++
2. To learn about the invocation & execution of constructors & ~~dstr~~ destructors in C++.

Theory:

Explain

1) Default Constructor:
⇒ It is a constructor that takes no parameters. It is automatically created. It initializes the object's members to default values. If no constructor is defined, C++ provides a default constructor.

Syntax:
```
class ClassName {
public:
    ClassName ( ) {
    }
};
```

2) Parameterized Constructor.
A parameterized ~~constructor~~ constructor accepts one or more parameters to initialize object members with specific values. It allows you to create objects with custom initial states.
Syntax:
```
class ClassName {
public:
    ClassName(int param1, double param2) {
        // Initialize code using parameters
    }
};
```

3) Copy Constructor:
A copy constructor creates a new object as a copy of an existing object. It's called when an object is passed by value, or explicitly created using another object of the same class.
Syntax:
```
class ClassName {
public:
   ClassName(const ClassName& other) {
       // Copy members from 'other' to this object
   }
};
```

4) Destructor:
A destructor is called automatically when an object goes out of scope or is explicitly deleted. It's used to

clean up resources (deallocate memory) that the object may have acquired during it's lifetime.
Syntax:

```
class ClassName {
public:
    ~Class Name () {
            // Cleanup Code
    }
};
```

Algorithm:
1. Start
2. Create a student class.
3. Create a default constructor.
4. Create a parameterized constructor & a copy constructor.
5. Create a destructor
6. Define accept and display member functions of the class.
7. Display the student database.
8. Stop.

Platform: 64-bit Open source Linux.

Input:
Student information: Name, Roll number, Class, division, Date of birth, Adhar Number, Blood group, Contact address, telephone number. Type of constructor being called and the destructor being called.

Page No. :
Date. : / /

Output:
Student Database containing Name, Roll number, Class, division, Date of Birth, Adhar number, Blood group, Contact address, telephone number. Type of constructor being called and the destructor being called and the destructor being called.

Conclusion:
~~Hean~~ Hence, learnt to use constructors and destructors in C++

FAQ's:

1) What is the order of constructor execution in C++?
→ • Base class constructors are executed first, in order of inheritance.
• Then, member object constructors are called in the order of declaration.
• Finally, the derived class constructor body is executed
• For multiple inheritance, constructors are called in left-to-right order as listed in class declaration.

2) How do constructors and destructors manage dynamic memory allocation?
→ Constructor allocate memory for dynamic members using 'new' operator. They initialize pointers to newly allocated memory.

Destructors deallocate this memory using 'delete' operator. This ensures proper resource management throughout the objects lifecycle.
It is crucial for preventing memory leaks and maintaining RAII principle.

3) What is the significance of copy constructor in C++?
→ · It ~~prevents~~ creates a new object as a copy of an existing object. Prevents shallow copying of objects with dynamic memory. Ensures proper duplication of resources owned by the original object. Crucial for pass-by-value semantics and returning objects from functions.
Allows for custom deep copying behaviour when needed.

4) How does C++ hande default initialization if no constructor is provided?
→ C++ provides an implicit default constructor if no constructor is defined.
Built-in types (int, float, etc.) are left initialized
Member objects are initialized using their respective default constructors.

[illegible] 9/8/24

# OOC LABORATORY

**Lab Assignment : 02**

Name : Ayush Kadali
Roll No. : 54
PRN : 1032232229
Panel : B
Batch : B2
SY B.Tech CSE (AI & DS)

## Input Code:

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

class Student {
private:
    char name[50];
    int rollNumber;
    char className[20];
    char division;
    char dateOfBirth[11];
    char aadharNumber[13];
    char bloodGroup[4];
    char contactAddress[100];
    long telephoneNumber;

public:
    // Default constructor
    Student() {
        strcpy(name, "Ayush");
        rollNumber = 102045;
        strcpy(className, "OOP");
        division = 'B';
        strcpy(dateOfBirth, "07-12-2005");
        strcpy(aadharNumber, "9485 3834 8459");
        strcpy(bloodGroup, "B+");
        strcpy(contactAddress, "Pune");
        telephoneNumber = 9920485937;
    }
    // Parameterized constructor
    Student(const char* n, int roll, const char* class_name, char div, const char* dob,
            const char* aadhar, const char* blood, const char* address, long phone) {
        strcpy(name, n);
        rollNumber = roll;
        strcpy(className, class_name);
        division = div;
        strcpy(dateOfBirth, dob);
        strcpy(aadharNumber, aadhar);
        strcpy(bloodGroup, blood);
        strcpy(contactAddress, address);
        telephoneNumber = phone;
    }

    // Copy constructor
    Student(const Student& other) {
        strcpy(name, other.name);
        rollNumber = other.rollNumber;
        strcpy(className, other.className);
        division = other.division;
        strcpy(dateOfBirth, other.dateOfBirth);
        strcpy(aadharNumber, other.aadharNumber);
        strcpy(bloodGroup, other.bloodGroup);
        strcpy(contactAddress, other.contactAddress);
        telephoneNumber = other.telephoneNumber;
    }

    void display() {
        cout << "Name: " << name << endl;
        cout << "Roll Number: " << rollNumber << endl;
        cout << "Class: " << className << endl;
        cout << "Division: " << division << endl;
        cout << "Date of Birth: " << dateOfBirth << endl;
        cout << "Aadhar Number: " << aadharNumber << endl;
        cout << "Blood Group: " << bloodGroup << endl;
        cout << "Contact Address: " << contactAddress << endl;
        cout << "Telephone Number: " << telephoneNumber << endl;
        cout << "------------------------" << endl;
        cout << endl;
    }
};

int main() {
    // Using default constructor
    Student student1;
    cout << "Student 1 (Default Constructor):" << endl;
    student1.display();

    // Using parameterized constructor
    Student student2("Rohan", 1038492390, "SYAIDS", 'A', "20-08-2005", "9850 4850 9490",
"O+", "Mumbai", 9485038596);
    cout << "Student 2 (Parameterized Constructor):" << endl;
    student2.display();

    // Using copy constructor
    Student student3 = student2;
    cout << "Student 3 (Copy Constructor):" << endl;
    student3.display();

    return 0;
}
```

## OUTPUT

```
Student 1 (Default Constructor):
Name: Ayush
Roll Number: 102045
Class: OOP
Division: B
Date of Birth: 07-12-2005
Aadhar Number: 9485 3834 84
Blood Group: B+
Contact Address: Pune
Telephone Number: 9920485937
------------------------

Student 2 (Parameterized Constructor):
Name: Rohan
Roll Number: 1038492390
Class: SYAIDS
Division: A
Date of Birth: 20-08-2005
Aadhar Number: 9850 4850 94
Blood Group: O+
Contact Address: Mumbai
Telephone Number: 9485038596
------------------------

Student 3 (Copy Constructor):
Name: Rohan
Roll Number: 1038492390
Class: SYAIDS
Division: A
Date of Birth: 20-08-2005
Aadhar Number: 9850 4850 94
Blood Group: O+
Contact Address: Mumbai
Telephone Number: 9485038596
------------------------
```