

2/22/2025

OOPS LAB

ASSIGNMENT#2



NAME: AHTISHAM KHAN
ENROLLMENT: 09-131241-014
DEPARTMENT: BSE-2B

Classes

Introduction to Classes and Objects

Objective:

The objective of this lab is to introduce the concept of classes and objects in C++. By the end of this lab, you should be able to:

1. Define a class with data members and member functions.
 2. Create objects of a class.
 3. Understand the difference between classes and objects.
 4. Understand the concept of access specifiers (public, private, protected).
-

1. Classes in C++

- A class is a user-defined data type that encapsulates data members (attributes) and member functions (methods).
- It acts as a blueprint for creating objects.

Syntax:

```
cpp                                                                    Copy
class ClassName {
private:
    // Data members (attributes)
    // Member functions (methods)
public:
    // Data members (attributes)
    // Member functions (methods)
};
```

2. Objects in C++

- An object is an instance of a class. It is a real-world entity that has state (data members) and behavior (member functions).

Syntax:

```
cpp                                                                    Copy
ClassName objectName;
```

3. Access Specifiers

- **Public:** Members are accessible from outside the class.
- **Private:** Members are not accessible from outside the class (default for classes).
- **Protected:** Similar to private but accessible in derived classes.

Lab Activities:

Activity 1: Define a Class and Create Objects

Define a class named Student with the following:

- Data members: name, rollNumber, age
- Member functions: display() to print the student details.

Code:

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

class Student {
private:
    string name;
    int rollNumber;
    int age;

public:
    // Member function to set data
    void setData(string n, int r, int a) {
        name = n;
        rollNumber = r;
        age = a;
    }

    // Member function to display data
    void display() {
        cout << "Name: " << name << endl;
        cout << "Roll Number: " << rollNumber << endl;
        cout << "Age: " << age << endl;
    }
};
```

```

int main() {
    // Create objects of the class Student
    Student student1;
    Student student2;

    // Set data for student1
    student1.setData("John Doe", 101, 20);

    // Set data for student2
    student2.setData("Jane Smith", 102, 21);

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

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

    return 0;
}

```

Output:

Copy

```

Student 1 Details:
Name: John Doe
Roll Number: 101
Age: 20

Student 2 Details:
Name: Jane Smith
Roll Number: 102
Age: 21

```

Your Output Here....

In case of output snippet please make sure output snippet contains student name and id. `AliAhmed_123_Lab02_A1.exe`

```

Ahtisham khan (014) OOP LAB 02
Student 1 Detalis:
NAME:Maha Tabinda Quershi
ENROLLMENT:09-131242-014
AGE:18

Student 2 Detalis:
NAME:Ahtisham khan
ENROLLMENT:09-131242-016
AGE:19

Press any key to continue . . . _

```

Activity 2: Access Specifiers

Modify the Student class to demonstrate the use of access specifiers (public, private).

Code:

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

class Student {
private:
    string name;
    int rollNumber;

public:
    int age; // Public data member

    // Public member function to set private data
    void setData(string n, int r, int a) {
        name = n;
        rollNumber = r;
        age = a;
    }

    // Public member function to display data
    void display() {
        cout << "Name: " << name << endl;
        cout << "Roll Number: " << rollNumber << endl;
        cout << "Age: " << age << endl;
    }
};

int main() {
    Student student1;

    // Set data for student1
    student1.setData("Alice Johnson", 103, 22);

    // Access public data member directly
    cout << "Accessing public data member (age): " << student1.age << endl;

    // Display student details
    cout << "Student 1 Details:" << endl;
    student1.display();
    return 0;}
```

Output:

```
Accessing public data member (age): 22
Student 1 Details:
Name: Alice Johnson
Roll Number: 103
Age: 22
```

[Copy](#)

Your Output Here....

In case of output snippet please make sure output snippet contains student name and id. `AliAhmed_123_Lab02_A2.exe`

```
Ahtisham khan (014) OOP LAB 02
Accessing Public data member(age): 18

Student 1 Detalis:
NAME: Maha Tabinda Quershi
ENROLLMENT: 09-131242-014
AGE: 18

Press any key to continue . . .
```

Activity 3: Class with Member Functions Defined Outside

Define a class Rectangle with data members length and width.

Define member functions setData() and calculateArea() outside the class.

Code:

```

#include <iostream>
using namespace std;

class Rectangle {
private:
    float length;
    float width;

public:
    void setData(float l, float w); // Function declaration
    float calculateArea();          // Function declaration
};

// Function definitions outside the class
void Rectangle::setData(float l, float w) {

    length = l;
    width = w;
}

float Rectangle::calculateArea() {
    return length * width;
}

int main() {
    Rectangle rect;

    // Set data for rectangle
    rect.setData(5.5, 4.0);

    // Calculate and display area
    cout << "Area of Rectangle: " << rect.calculateArea() << endl;

    return 0;
}

```


Output:

Area of Rectangle: 22

Copy

Your Output Here....

In case of output snippet please make sure output snippet contains student name and id. `AliAhmed_123_Lab02_A3.exe`

 Ahtisham khan (014) OOP LAB 2

Enter the first float number:

4.4

Enter the second float number:

5.6

The area of rectangle: 24.64

Press any key to continue . . . _

Conclusion:

In this lab, we learned:

1. How to define a class with data members and member functions.
2. How to create objects of a class and access their members.
3. The importance of access specifiers in encapsulation.
4. How to define member functions outside the class.

Additional Notes:

- Classes are the foundation of Object-Oriented Programming (OOP) in C++.
- Objects are instances of classes and hold separate copies of data members.
- Access specifiers control the visibility of class members.

Lab Tasks/ Homework

Task 01: Write a program to add two different times (hours, minutes and seconds) using simple class and display total time and convert time into number of days.

Write functions:

```
getTime();  
displayTime(Time obj);  
void addTime(Time obj1,Time obj2);  
int numberOfDays(Time obj);
```

Solution:

Enter your code here...

```
#include<iostream>  
#include<cstdlib>  
using namespace std;  
class CalculateTime {  
private:  
    int hours;  
    int minutes;  
    int seconds;  
public:  
    void calTime(int T,int M,int S) {  
        hours = T;  
        minutes = M;  
        seconds = S;  
    }  
    // getting values  
    void Gettime() {  
        cout << "Enter the hour: " << endl;  
        cin >> hours;  
        cout << "Enter the minutes: " << endl;  
        cin >> minutes;  
        cout << "Enter the seconds: " << endl;  
        cin >> seconds;  
    }  
    // displaying values  
    void Displaytime() {  
        cout << "Time in hours: " << hours << endl;  
        cout << "Time in minutes: " << minutes << endl;  
        cout << "Time in seconds: " << seconds << endl;  
    }  
}
```

```

// adding time

void addtime(CalculateTime t1, CalculateTime t2) {
    cout << "Total hours, minutes and seconds of both objects:"
<< endl;
    cout << endl;
    hours=t1.hours + t2.hours;
    minutes = t1.minutes + t2.minutes;
    seconds=t1.seconds + t2.seconds;

}
// for number of days
int Numberofdays() {
    return hours / 24;
}
};
int main() {
    system("title Ahtisham khan (014) OOPS LAB 2");

    CalculateTime t1;
    t1.Gettime();
    t1.Displaytime();
    cout << endl;
    CalculateTime t2;
    t2.Gettime();
    t2.Displaytime();
    cout << endl;
    CalculateTime pp;
    pp.addtime(t1, t2);
    pp.Displaytime();
    cout << endl;
    cout << "The numbers of Days: " << pp.Numberofdays() << endl;
    cout << endl;
    cout << endl;
    system("pause");
}

```

Output:

Your Output here...

```
Ahtisham khan (014) OOPS LAB 2
Enter the hour:
12
Enter the minutes:
23
Enter the seconds:
25
Time in hours: 12
Time in minutes: 23
Time in seconds: 25

Enter the hour:
13
Enter the minutes:
23
Enter the seconds:
16
Time in hours: 13
Time in minutes: 23
Time in seconds: 16

Total hours, minutes and seconds of both objects:

Time in hours: 25
Time in minutes: 46
Time in seconds: 41

The numbers of Days: 1

Press any key to continue . . . _
```

In case of output snippet please make sure output snippet contains student name and id. AliAhmed_123_Lab02_T1.exe

Task 02: C++ program to create a class to read and add two distance, Distance should be input in the form of feet and inch and check if inch is greater than 12 and increment it into feet and display remaining inch and display using the given function.

Write functions:

void getDist ();

void showDist ();

Distance addDist(Distance);

Distance subDist(Distance);

Solution:

Enter your code here...

```
#include<iostream>
#include<cstdlib>
using namespace std;
class Distance {
private:
    int feet;
    int inches;
public:
    void caldistance(int F, int I) {
        feet = F;
        inches = I;
    }
    // function for getting values
    void getdistance() {
        cout << "Enter the distance in feet: " << endl;
        cin >> feet;
        cout << "Enter the distance in inches: " << endl;
        cin >> inches;
    }
    // function for displaying values
    void displaydistance() {
        cout << "Feet " << feet << " inches " << inches << endl;
    }
    // function for adding both distance
    void adddistance(Distance d1, Distance d2) {

        feet = d1.feet + d2.feet;
        inches = d1.inches + d2.inches;
        if (inches >= 12) {
            feet += inches / 12;
            inches = inches % 12;
        }

    }
    // function for subtracting distances
    void subdistance(Distance d1, Distance d2) {
        feet = d1.feet - d2.feet;
        inches = d1.inches - d2.inches;
    }
};
```

```

int main() {
    // object 1
    Distance d1;
    cout << "Enter first distance: " << endl;
    cout << endl;
    d1.getdistance();
    d1.displaydistance();
    cout<< endl;
    // object 2
    Distance d2;
    cout << "Enter Second distance: " << endl;
    cout << endl;
    d2.getdistance();
    d2.displaydistance();
    cout << endl;
    Distance dp;
    // calling the addition functions
    cout << "Addition of both distances: " << endl;
    dp.adddistance(d1, d2);
    dp.displaydistance();
    cout << endl;
    // calling the subtraction function
    cout << "Substraction of both distances: " << endl;
    dp.subdistance(d1, d2);
    dp.displaydistance();

}

```

Output:

Your Output here...

```
Ahtisham khan (014) OOPS LAB 2
Enter first distance:

Enter the distance in feet:
13
Enter the distance in inches:
14
Feet 13 inches 14

Enter Second distance:

Enter the distance in feet:
41
Enter the distance in inches:
144
Feet 41 inches 144

Addition of both distances:
Feet 67 inches 2

Substraction of both distances:
Feet -28 inches -130

Press any key to continue . . .
```

In case of output snippet please make sure output snippet contains student name and id. AliAhmed_123_Lab02_T2.exe

#####