



Institute of Geographical Information Systems

CS-212 - Object Oriented Programming LAB

Semester: Fall 2025

Class: SCEE-IGIS - 2024

Name: Ali Nawaz

CMS ID : 00000526123

Submitted to: Ma'am Alvina Anjum

Due Date: Oct 01, 2025

LAB 04: Constructors & Destructors

Task # 1:

Define a Point class with x and y coordinates. Write a function **pDistance(Point, Point)** that computes the distance between two points. Write a function **pathLength(Point[], int)** that takes an array of Point (representing a path) and its size as input parameters and compute the path length by using the **pDistance** function. Write a main program to test your class and functions.

Screenshot:

```
OOP > Week-04 > Problem1.cpp > main()
1  #include <iostream>
2  #include <cmath>
3  using namespace std;
4
5  class Point
6  {
7  public:
8      float x, y;
9      Point()
10     {
11         x = 0;
12         y = 0;
13     }
14     Point(float a, float b)
15     {
16         x = a;
17         y = b;
18     }
19 };
20
21 float pDistance(Point p1, Point p2)
22 {
23     return sqrt(pow(p2.x - p1.x, 2) + pow(p2.y - p1.y, 2));
24 }
25
26 float pathLength(Point path[], int size)
27 {
28     float total = 0;
29     for (int i = 0; i < size - 1; i++)
30         total += pDistance(path[i], path[i + 1]);
31     return total;
32 }
33
34 int main()
35 {
36     Point path[4] = {Point(0, 0), Point(3, 4), Point(6, 8), Point(9, 12)};
37     cout << "Total Path Length: " << pathLength(path, 4);
38     return 0;
39 }
```

EXPLORER

- OPEN EDITORS
- DEVELOPMENT
 - .env
 - vscode
 - CwA
 - Jargon
 - NUSTVerse
 - OOP
 - idea
 - cmake-build-debug
 - Week-01
 - Week-02
 - Week-03
 - OOP LAB03 - 526123.pdf
 - Problem1
 - Problem1.cpp
 - Problem2
 - Problem2.cpp
 - Problem3
 - Problem3.cpp
 - Week03-526123.zip
 - Week-04
 - Problem1
 - Problem1.cpp
 - Problem2.cpp
 - Problem3.cpp
 - Week-05
 - CMakeLists.txt
 - index.cpp
 - jargon.cpp
 - main.cpp
 - TFUGIslamabad
 - ok
 - project.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER

```
• (.venv) alinawaz@Alis-MacBook-Air Week-05 % cd "/Users/alinawaz/Developer/Development/OOP/Week-04/" && g++ Problem1.cpp -o Problem1 && ./Problem1
Total Path Length: 15
• (.venv) alinawaz@Alis-MacBook-Air Week-04 %
```

Ln 39, Col 2 Spaces: 4 UTF-8 (LF) C++ Go Live Mac Prettier

Task # 2:

Create a class Employee that manages employee details for a company. Each employee has:

- Name
- Age
- Salary
- Department

The company should be able to:

1. Hire employees by creating separate objects.
2. Display employee details.
3. Keep track of the total number of employees (using a counter in main, not static).
4. Identify the employee with the highest salary.
5. Search for employees in a given department and display them.

In the main() function:

- Hire at least 5–6 employees as separate objects.
- Display their details.
- Show the total number of employees hired.
- Find and display the employee with the highest salary.
- Allow the user to enter a department name and display employees of that department.

Screenshot:

```
OOP > Week-04 > Problem2.cpp > main()
1 #include <iostream>
2 #include <string>
3 using namespace std;
4
5 class Employee
6 {
7 public:
8     string name;
9     int age;
10    float salary;
11    string department;
12    Employee() {}
13    Employee(string n, int a, float s, string d)
14    {
15        name = n;
16        age = a;
17        salary = s;
18        department = d;
19    }
20    void display()
21    {
22        cout << "Name: " << name << ", Age: " << age << ", Salary: " << salary << ", Department: " << department << endl;
23    }
24 };
25
26 int main()
27 {
28     Employee e1("Ali", 22, 60000, "IT");
29     Employee e2("Ahsan", 24, 75000, "HR");
30     Employee e3("Fatima", 26, 90000, "Finance");
31 }
```

```

e2
All Employees:
Name: Ali, Age: 22, Salary: 60000, Department: IT
Name: Ahsan, Age: 24, Salary: 75000, Department: HR
Name: Fatima, Age: 26, Salary: 90000, Department: Finance
Name: Sara, Age: 23, Salary: 65000, Department: IT
Name: Bilal, Age: 27, Salary: 85000, Department: Finance
Name: Zain, Age: 25, Salary: 70000, Department: Marketing
Total Employees: 6
Employee with Highest Salary:
Name: Fatima, Age: 26, Salary: 90000, Department: Finance
Enter Department to Search: IT
Employees in IT Department:
Name: Ali, Age: 22, Salary: 60000, Department: IT
Name: Sara, Age: 23, Salary: 65000, Department: IT
(.venv) alinawaz@Alis-MacBook-Air Week-04 %
```

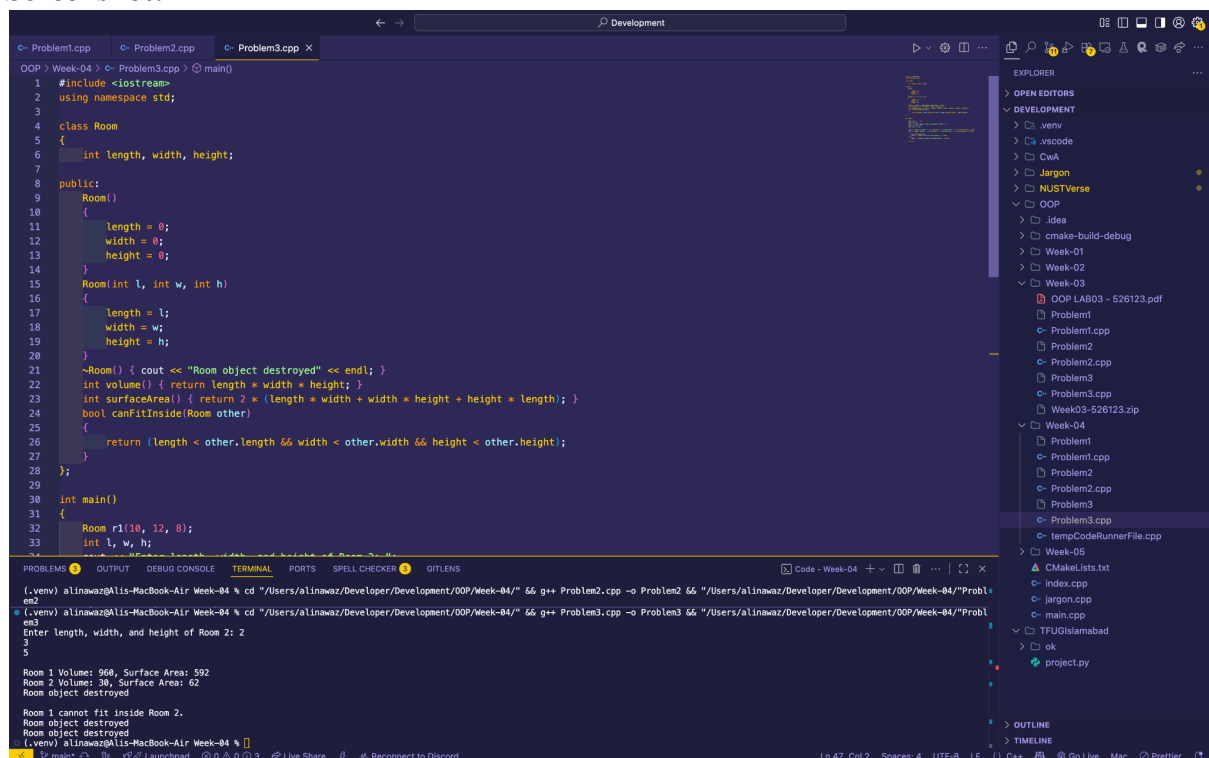
Task # 3:

Create a class Room that represents a 3D rectangular room.

Requirements:

1. Data members:
 - int length
 - int width
 - int height
2. Constructors & Destructor:
 - A default constructor that sets all dimensions
 - A parameterized constructor that sets dimensions from arguments.
 - A destructor that prints a message like "Room object destroyed".
3. Member functions:
 - volume() → returns the volume of the room.
 - surfaceArea() → returns the surface area of the room
 - canFitInside(Room other) → returns true if the current room can completely fit inside the other room (all three dimensions must be smaller).
4. In main():
 - Create a Room r1 with length 10, width 12 and height 8 and for Room r2 take input from the user.
 - Print the volume and surface area of both rooms.
 - Check if r1 can fit inside r2, and display the result.
 - Observe destructor messages at the end.

Screenshot:



```
1 #include <iostream>
2 using namespace std;
3
4 class Room
5 {
6     int length, width, height;
7
8 public:
9     Room()
10    {
11        length = 0;
12        width = 0;
13        height = 0;
14    }
15    Room(int l, int w, int h)
16    {
17        length = l;
18        width = w;
19        height = h;
20    }
21    ~Room() { cout << "Room object destroyed" << endl; }
22    int volume() { return length * width * height; }
23    int surfaceArea() { return 2 * (length * width + width * height + height * length); }
24    bool canFitInside(Room other)
25    {
26        return (length < other.length && width < other.width && height < other.height);
27    }
28 };
29
30 int main()
31 {
32     Room r1(10, 12, 8);
33     int l, w, h;
34     cin >> l >> w >> h;
35     Room r2(l, w, h);
36
37     cout << "Room 1 Volume: " << r1.volume() << ", Surface Area: " << r1.surfaceArea() << endl;
38     cout << "Room 2 Volume: " << r2.volume() << ", Surface Area: " << r2.surfaceArea() << endl;
39     cout << "Room object destroyed" << endl;
40     cout << "Room 1 cannot fit inside Room 2." << endl;
41     cout << "Room object destroyed" << endl;
42     cout << "Room object destroyed" << endl;
43 }
```

Room 1 Volume: 960, Surface Area: 592
Room 2 Volume: 384, Surface Area: 62
Room object destroyed
Room 1 cannot fit inside Room 2.
Room object destroyed
Room object destroyed