



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 24, 2025

LAB 07: Using Header Files to keep class in separate files

Task # 1:

1. Create a header file named `currency_converter.h`
 - Declare a class `CurrencyConverter` with private attributes:
 - `float usdRate; // 1 USD to PKR`
 - `float eurRate; // 1 EUR to PKR`
 - `float gbpRate; // 1 GBP to PKR`
 - Provide public member functions:
 - `Constructor(s)` to initialize conversion rates.
 - `void setRates(float usd, float eur, float gbp)` — updates the exchange rates.
 - `float convertToUSD(float pkr)` — converts PKR to USD.
 - `float convertToEUR(float pkr)` — converts PKR to EUR.
 - `float convertToGBP(float pkr)` — converts PKR to GBP.
 - `void displayRates()` — displays current conversion rates.
2. Implement all functions in `currency_converter.cpp`.
3. Write a `main.cpp` program that:
 - Creates an object of the `CurrencyConverter` class.
 - Sets initial rates (e.g., 1 USD = 280 PKR, 1 EUR = 300 PKR, 1 GBP = 350 PKR).
 - Takes an amount in PKR from the user.
 - Displays equivalent values in USD, EUR, and GBP.
 - Optionally allows the user to update exchange rates.

Example Output:

```
--- Currency Converter ---  
Current Rates:  
1 USD = 280 PKR  
1 EUR = 300 PKR  
1 GBP = 350 PKR
```

```
Enter amount in PKR: 5600  
Equivalent in USD: 20.00  
Equivalent in EUR: 18.67  
Equivalent in GBP: 16.00
```

If the user updates rates:

```
Enter new rates (USD, EUR, GBP): 285 310 355  
Rates updated successfully!
```

Screenshot:

The screenshot shows a C++ IDE with a project named 'OOP' and a task named 'TaskNo1'. The code in 'main.cpp' defines a 'CurrencyConverter' class with methods to display rates, convert between currencies, and update rates. The terminal output shows the program running on a MacBook-Air, displaying the current rates, converting an input of 100000 PKR to USD (357.143), EUR (333.333), and GBP (285.714), and then successfully updating the rates to 282 PKR for USD, 327 PKR for EUR, and 380 PKR for GBP.

```
1 >> #include ...  
3 using namespace std;  
4  
5 int main() {  
6     CurrencyConverter converter;  
7     converter.displayRates();  
8  
9     float pkr;  
10    cout << "\nEnter amount in PKR: ";  
11    cin >> pkr;  
12  
13    cout << "Equivalent in USD: " << converter.convertToUSD(pkr) << endl;  
14    cout << "Equivalent in EUR: " << converter.convertToEUR(pkr) << endl;  
15    cout << "Equivalent in GBP: " << converter.convertToGBP(pkr) << endl;  
16  
17    char choice;  
18    cout << "\nDo you want to update rates? (y/n): ";  
19    cin >> choice;  
20  
21    if (choice == 'y' || choice == 'Y') {  
22        float new_usd, new_eur, new_gbp;  
23        cout << "Enter new rates (USD, EUR, GBP): ";  
24        cin >> new_usd >> new_eur >> new_gbp;  
25        converter.updateRates(new_usd, new_eur, new_gbp);  
26        cout << "Rates updated successfully!" << endl;  
27    }  
28    return 0;  
29 }
```

```
alinalawaz@Alis-MacBook-Air TaskNo1 % ./main  
Current Rates:  
1 USD = 280 PKR  
1 EUR = 300 PKR  
1 GBP = 350 PKR  
  
Enter amount in PKR: 100000  
Equivalent in USD: 357.143  
Equivalent in EUR: 333.333  
Equivalent in GBP: 285.714  
  
Do you want to update rates? (y/n): y  
Enter new rates (USD, EUR, GBP): 282  
327  
380  
Rates updated successfully!  
Current Rates:  
1 USD = 282 PKR  
1 EUR = 327 PKR  
1 GBP = 380 PKR  
  
alinalawaz@Alis-MacBook-Air TaskNo1 %
```

Task # 2:

Objective:

Upon completion of this task, students will be able to:

- Design and implement a class structure in C++.
- Use constructors, destructors, and member functions effectively.
- Practice clean class design and modularization using header files.

Task Description:

1. Create a header file named **design_example.h**

Declare a class **Box** representing a 3D box.

Include **private attributes**:

- float length;
- float width;
- float height;

Provide **public member functions**:

- Constructor(s) and Destructor.
- `void setDimensions(float l, float w, float h)` — sets box dimensions.
- `float calculateVolume()` — computes and returns volume.
- `float calculateSurfaceArea()` — computes and returns total surface area.
- `void display()` — displays dimensions, volume, and surface area.

2. Implement all functions in **design_example.cpp**.

3. Write a **main.cpp** program that:

- Creates multiple **Box** objects.
- Sets different dimensions for each box.
- Displays their dimensions, volume, and surface area.
- Demonstrates the effect of constructor and destructor calls (use messages).

Example Output

Box 1:

Length = 4, Width = 3, Height = 2

Volume = 24

Surface Area = 52

Box 2:

Length = 6, Width = 4, Height = 3

```
Volume = 72
Surface Area = 108
```

General Requirements

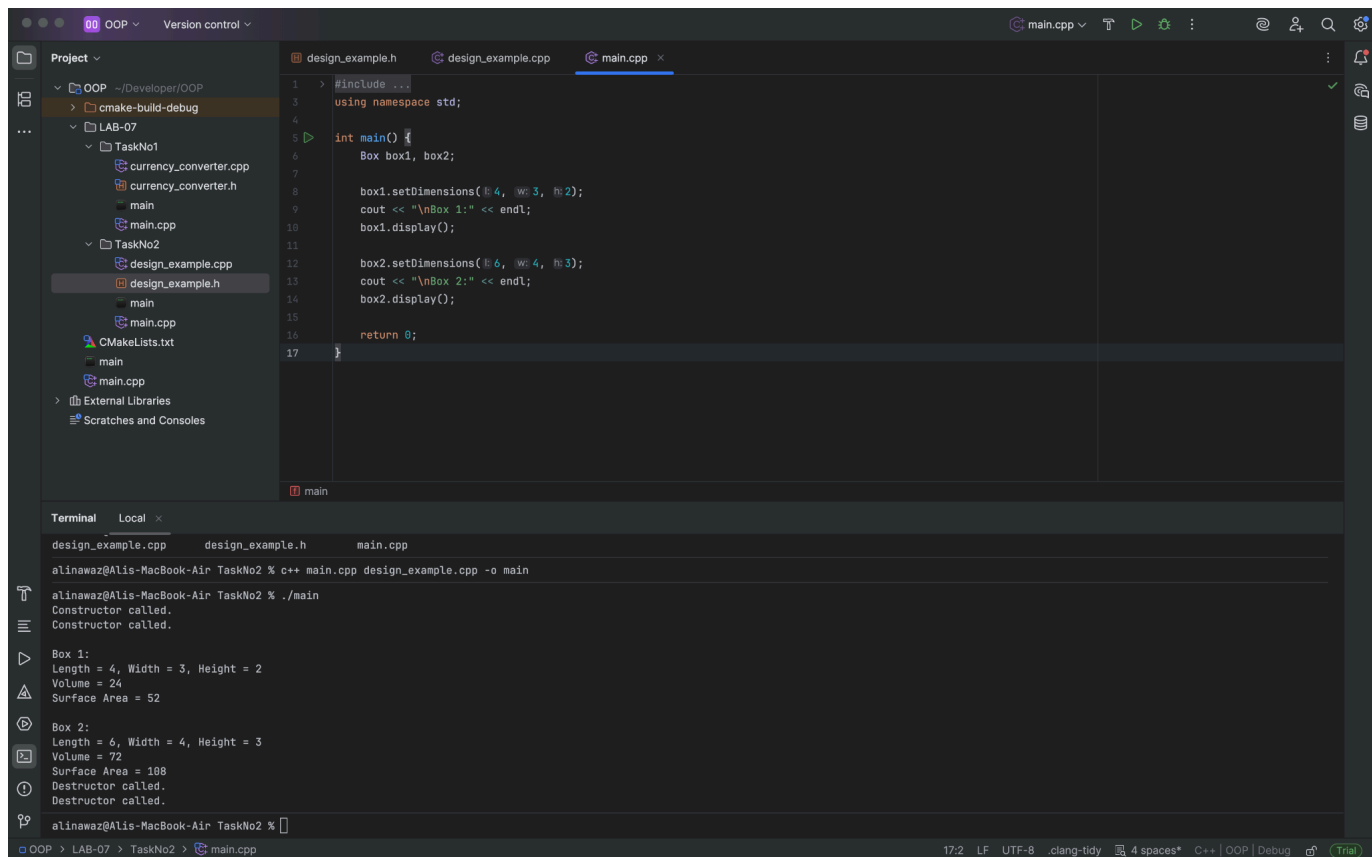
- All data members in classes must be private.
- No global variables are allowed.
- Each module must have a .h and a .cpp file for separation of declaration and implementation.
- The main.cpp file must demonstrate the usage of each module.
- Use appropriate comments and indentation.
- Test your program with multiple sets of inputs.

Deliverables

Submit the following files for each task:

- **Header (.h) file**
- **Source (.cpp) file**
- **main.cpp**

Screenshot:



```
1 > #include <iostream>
2 using namespace std;
3
4
5 int main() {
6     Box box1, box2;
7
8     box1.setDimensions(4, 3, 2);
9     cout << "Box 1:" << endl;
10    box1.display();
11
12    box2.setDimensions(6, 4, 3);
13    cout << "Box 2:" << endl;
14    box2.display();
15
16    return 0;
17 }
```

```
alinalawaz@Alis-MacBook-Air TaskNo2 % c++ main.cpp design_example.cpp -o main
alinalawaz@Alis-MacBook-Air TaskNo2 % ./main
Constructor called.
Constructor called.
Box 1:
Length = 4, Width = 3, Height = 2
Volume = 24
Surface Area = 52
Box 2:
Length = 6, Width = 4, Height = 3
Volume = 72
Surface Area = 108
Destructor called.
Destructor called.
alinalawaz@Alis-MacBook-Air TaskNo2 %
```