LAB SESSION 12

Friend Functions in C++

Objective
The objective of this lab is to understand and apply the concept of friend functions in C++. By the end of this lab, you should be able to define and use friend functions to access private and protected members of a class.

Introduction
In C++, friend functions are functions that are not member functions of a class but have access to the class's private and protected members. Friend functions are useful when you need to allow a non-member function to access private data of a class, which can be helpful in certain scenarios such as interfacing with external functions.

Theory
Friend Functions
A friend function is defined outside the class but has the right to access all private and protected members of the class. A friend function is declared using the keyword friend inside the class.

```
class ClassName {
    friend returnType friendFunctionName(arguments);
    // Other class members
};
```

Example:

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

class Box {
private:
    double width;
public:
    Box(double w) : width(w) {}

    // Friend function declaration
    friend void printWidth(Box box);
};

// Friend function definition
void printWidth(Box box) {
    cout << "Width of box: " << box.width << endl;
}
```

```
int main() {
    Box box(10.5);
    printWidth(box);
    return 0;
}
```

In this example, printWidth is a friend function of the class Box and can access its private member width.

Conclusion

Friend functions provide a way to access the private and protected members of a class without being a member of the class. This can be particularly useful when you need to interface with external functions or perform operations that require access to private data.

Exercise:

1. Write a class Circle with a private member radius. Write a friend function to calculate the area of the circle.

2. Implement a class Rectangle with private members length and width. Write a friend function to calculate the perimeter of the rectangle.