Tutorial 27 - Friend Classes & Member Friend Functions in C++

1. Member Friend Functions

- **Definition**: Friend functions allow access to the private/protected members of the class in which they are declared.
- Only the specific functions declared as friends can access the private members.

Code Example: Member Friend Functions

```
class Complex {
    int a, b;
    // Declaring specific member functions of another class as friends
    friend int Calculator::sumRealComplex(Complex, Complex);
    friend int Calculator::sumCompComplex(Complex, Complex);
public:
    void setNumber(int n1, int n2) {
        a = n1;
        b = n2;
    }
    void printNumber() {
        cout << "Your number is " << a << " + " << b << "i" << endl;
    }
};
class Calculator {
public:
    int sumRealComplex(Complex o1, Complex o2) {
        return (o1.a + o2.a); // Access private members
    }
    int sumCompComplex(Complex o1, Complex o2) {
        return (o1.b + o2.b); // Access private members
    }
};

```

---

2. Friend Classes

- **Definition**: A friend class has access to all private and protected members of another class.
- Declared using the `friend class` keyword in the target class.

Code Example: Friend Classes

```
// Forward declaration
class Complex;
class Calculator {
public:
    int add(int a, int b) {
        return (a + b);
    }
    int sumRealComplex(Complex, Complex);
    int sumCompComplex(Complex, Complex);
};
class Complex {
    int a, b;
    // Declaring entire Calculator class as a friend
    friend class Calculator;
```

```
public:
    void setNumber(int n1, int n2) {
        a = n1;
        b = n2;
    }
    void printNumber() {
        cout << "Your number is " << a << " + " << b << "i" << endl;
    }
};
int Calculator ::sumRealComplex(Complex o1, Complex o2)
{
    return (o1.a + o2.a);
}
int Calculator ::sumCompComplex(Complex o1, Complex o2)
{
    return (o1.b + o2.b);
}
```

---

3. Main Program

```
int main() {
    Complex o1, o2;
    o1.setNumber(1, 4);
    o2.setNumber(5, 7);
    Calculator calc;
    int res = calc.sumRealComplex(o1, o2);
    cout << "The sum of real part of o1 and o2 is " << res << endl;
    int resc = calc.sumCompComplex(o1, o2);
    cout << "The sum of complex part of o1 and o2 is " << resc << endl;
    return 0;
}

```

---

4. Output

```
The sum of real part of o1 and o2 is 6
The sum of complex part of o1 and o2 is 11

```

---

5. Key Points

Member Friend Functions

1. Declared with `friend` keyword in a class.
2. Can access private/protected members of the class.
3. Must specify the exact function in the target class.

Friend Classes

1. Declared as `friend class ClassName` in a class.
2. Gives access to all private/protected members.
3. Can simplify multiple friend function declarations.

Usage Example

- Member friend functions: Fine-grained control, specific functions only.
- Friend classes: Broad access for all member functions.

---

6. Short Notes

Member Friend Functions

- Declared with `friend` in the target class.
- Allows specific functions of another class access to private members.
- Example:

```
friend int ClassName::FunctionName(Args);

```

Friend Classes

- Declared using `friend class ClassName`.
- Grants access to all private/protected members for the friend class.

Properties

1. Both enable private member access across classes.
2. Member functions: Precise control.
3. Friend classes: Full access for a class.