Problem Statement: Define a class Test where overload a method Sum() to sum numbers sent from main() function.

Source Code :

```
#include<bits/stdc++.h>
using namespace std;

class Test{
public :
    int Sum (int x){
        return x;
    }
    int Sum (int x, int y){
        return x + y;
    }
    double Sum (double x, int y){
        return x + y;
    }
    double Sum (int x, double y){
        return x + y;
    }
    double Sum (double x, double y){
        return x + y;
    }
};
int main(void){
    Test t;
    cout << t.Sum(10) << '\n';
    cout << t.Sum(10,20) << '\n';
    cout << t.Sum(5.7,20) << '\n';
    cout << t.Sum(10,2.6) << '\n';
    cout << t.Sum(10.5,20.7) << '\n';
}
```

Output :

```
10
30
25.7
12.6
31.2
```

Problem Statement: Suppose in a AC circuit, there are 3 impedances $z1 = 3 + j4, z2 = 4 - j3$ and $z3 = j6$ are connected in parallel. Now find the current in the circuit if input voltage is 100+j50. Implement **operator overloading** concept for your calculation. Use class **Circuit** and initialize the impedance values (real & img) by a constructor.

Source Code :

```cpp
#include <bits/stdc++.h>
using namespace std;

class Circuit {
private:
    double real, img;
public:
    Circuit(double x, double y) {
        real = x ,img = y;
    }
    Circuit inverse() const {
        double denom = real * real + img * img;
        return Circuit(real / denom, -img / denom);
    }
    Circuit operator+(const Circuit &other) const {
        return Circuit(real + other.real, img + other.img);
    }
    Circuit operator/(const Circuit &other) const {
        double denom = other.real * other.real + other.img * other.img;
        double r = (real * other.real + img * other.img) / denom;
        double i = (img * other.real - real * other.img) / denom;
        return Circuit(r, i);
    }
    void show(){
        if (img >= 0) cout << real << " + j" << img << '\n';
        else cout << real << " - j" << -img << '\n';
    }
};

int main() {
    Circuit z1(3.0, 4.0), z2(4.0, -3.0), z3(0.0, 6.0), v(100.0, 50.0);
    Circuit z_eq = z1.inverse() + z2.inverse() + z3.inverse();
    Circuit z_total = z_eq.inverse();
    Circuit current = v / z_total;

    cout << "The current is "; current.show();
}
```

Output:

```
The current is 38.3333 - j6.66667
```

Problem Statement: Write a program using unary operator overloading to control the sound of a TV remote. If ‘+’ button is pressed then sound increases and if ‘-‘ button pressed then sound decreases.

Source Code:

```cpp
#include<bits/stdc++.h>
using namespace std;

class remote{
    int sound;
public :
```

```
    remote(int x){
        sound = x;
    }
    remote operator ++(){
        sound += 5;
        return 0;
    }
    remote operator --(){
        sound -= 5;
        return 0;
    }
    void Show(){
        cout << "The Current Sound is " << sound << ".\n";
    }
};
int main(void){
    remote r(50);
    ++r; ++r; --r; ++r; --r;
    r.Show();
}
```

Output:

```
The Current Sound is 55.
```

Problem Statement: Suppose your money is deposited in two banks through their private data members **money1** and **money2** respectively. Now based on the statements in **main()** function, write a program to calculate your total money using a friend function **Sum().**

Source Code:

```
#include<bits/stdc++.h>
using namespace std;

class Bank1{
private :
    double money1;
public :
    Bank1(double x){
        money1 = x;
    }
    double GetMoney(){
        return money1;
    }
    friend void Sum(double tk1, double tk2);
};
class Bank2{
private :
    double money2;
public :
    Bank2(double x){
        money2 = x;
    }
```

```
    double GetMoney(){
        return money2;
    }
    friend void Sum(double tk1, double tk2);
};
void Sum(double tk1, double tk2){
    cout << "SUM = " << tk1 +tk2 << " BDT.\n";
}

int main(void){
    Bank1 b1(500.43);
    Bank2 b2(3450.32);
    Sum(b1.GetMoney(), b2.GetMoney());
}
```

Output:

```
SUM = 3950.75 BDT.
```