Day 9

15 April 2024 01:02 PM

Operator overloading :-

$c_3 = c_1.sum(c_2)$ ✓

or

$c_3 = c_1 + c_2;$ ✓

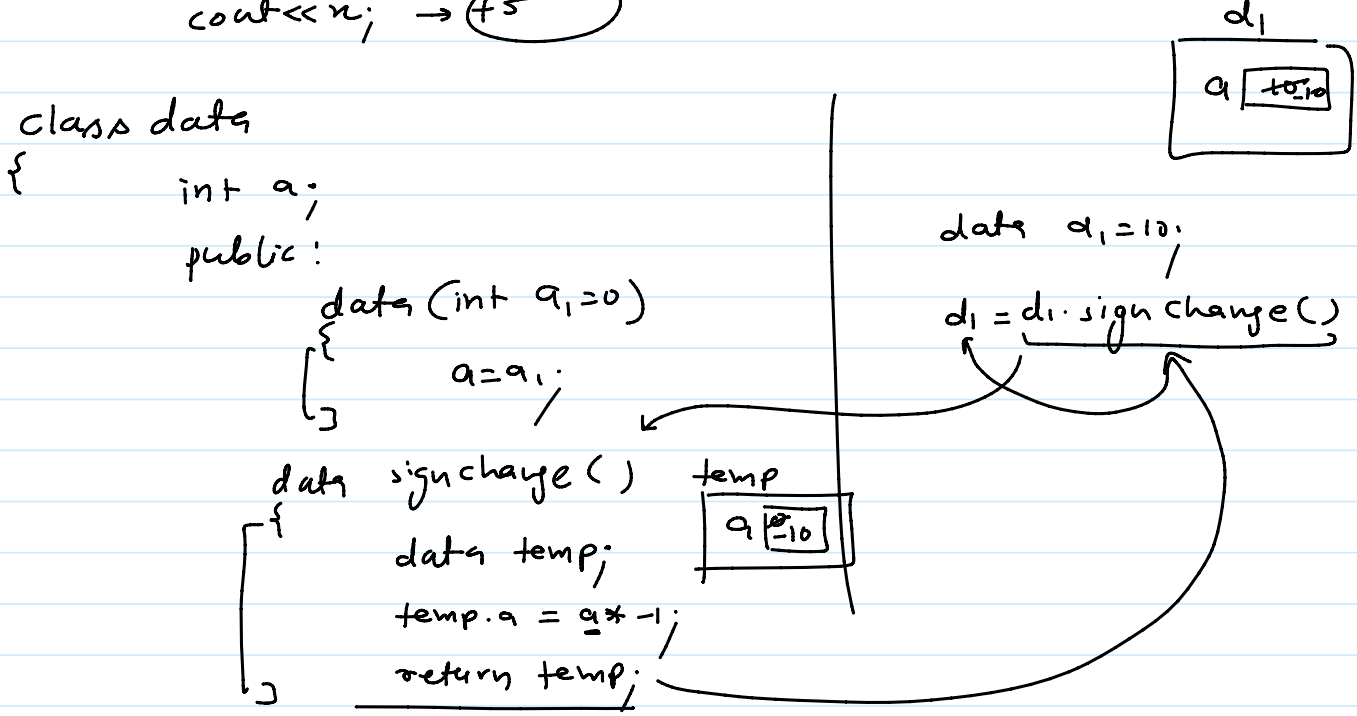
`-` `++` `--`

`-` ↑ sign change

`int x = 5;` x [5] y [-5]

`y = -x;`

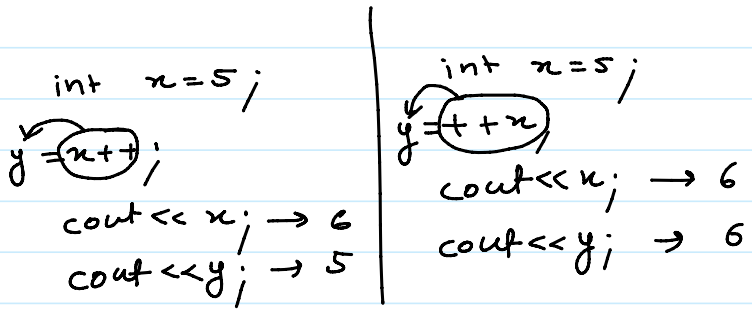
`cout<<x;` → +5



.

```
#include<iostream>
using namespace std;
class data
{
    int a;
```

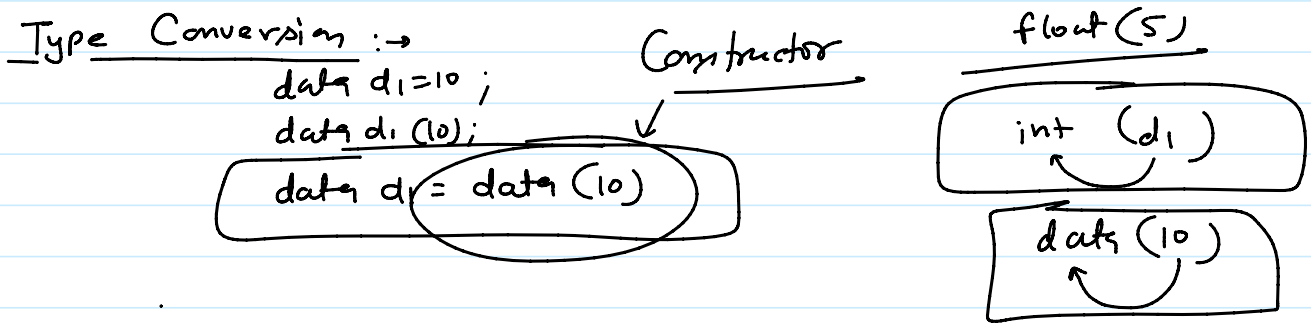
```
    public:
        data(int a1=0)
        {
            a=a1;
        }
        data signchange()
        {
            data temp;
            temp.a=a*-1;
            return temp;
        }
        data operator-()
        {
            data temp;
            temp.a=a*-1;
            return temp;
        }
        void output()
        {
            cout<<a<<endl;
        }
};
int main()
{
    data d1=10,d2;
    //d2=d1.signchange();
    d2=-d1;
    d1.output();    //10
    d2.output();    //-10
}
```



.

```
#include<iostream>
using namespace std;
class data
{
    int a;
    public:
        data(int a1=0)
        {
            a=a1;
        }
        data signchange()
        {
            data temp;
            temp.a=a*-1;
            return temp;
        }
        data operator-()
        {
            data temp;
```

```
            temp.a=a*-1;
            return temp;
        }
        void output()
        {
            cout<<a<<endl;
        }
        data inc()
        {
            a=a+1;
            data temp;
            temp.a=a;
            return temp;
        }
        data operator++()   //pre inc
        {
            a=a+1;
            data temp;
            temp.a=a;
            return temp;
        }
        data operator++(int)    //post inc
        {
            data temp;
            temp.a=a;
            a=a+1;
            return temp;
        }
};
int main()
{
    data d1=10,d2;
    //d2=d1.inc();
    d2=++d1;
    d1.output();    //11
    d2.output();    //11
    d2=d1++;
    d1.output();        //12
    d2.output();        //11
}
```



.

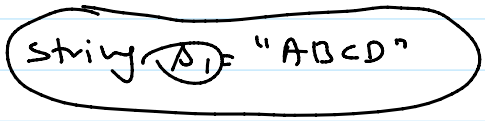
```
#include<iostream>
using namespace std;
class data
{
    int a;
    public:
        data(int a1=0)
        {
            a=a1;
        }
        data signchange()
```

```
        {
            data temp;
            temp.a=a*-1;
            return temp;
        }
        data operator-()
        {
            data temp;
            temp.a=a*-1;
            return temp;
        }
        void output()
        {
            cout<<a<<endl;
        }
        data inc()
        {
            a=a+1;
            data temp;
            temp.a=a;
            return temp;
        }
        data operator++()   //pre inc
        {
            a=a+1;
            data temp;
            temp.a=a;
            return temp;
        }
        data operator++(int)    //post inc
        {
            data temp;
            temp.a=a;
            a=a+1;
            return temp;
        }
        explicit operator int()
        {
            return a;
        }
};
int main()
{
    data d1=10,d2;
    //d2=d1.inc();
    d2=++d1;
    d1.output();    //11
    d2.output();    //11
    d2=d1++;
    d1.output();        //12
    d2.output();        //11
    int x;
    x=int(d1);
    cout<<"X = "<<x;
}
```

+ — * / %

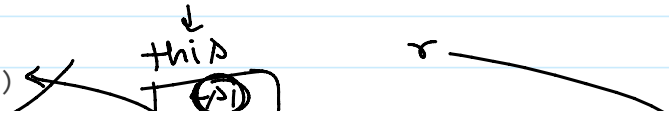
Complex :-

```cpp
#include<iostream>
using namespace std;
class complex
{
    int real, imag;
    public:
        complex(int r=0, int i=0)
        {
            real=r;
            imag=i;
        }
        void output()
        {
            if(imag<0)
                cout<<real<<imag<<"i\n";
            else
                cout<<real<<"+"<<imag<<"i\n";
        }
        complex sum(complex &r)
        {
            complex temp;
            temp.real=real+r.real;
            temp.imag = imag + r.imag;
            return temp;
        }
        complex operator+(complex &r)
        {
            complex temp;
            temp.real=real+r.real;
            temp.imag = imag + r.imag;
            return temp;
        }
        complex operator*(complex &r)
        {
            complex temp;
            temp.real=real*r.real - imag * r.imag;
            temp.imag = imag * r.real + real * r.imag;
            return temp;
        }
};
int main()
{
    complex c1(5,4),c2(7,3);
    complex c3=c1.sum(c2);
    c1.output();
    c2.output();
    c3.output();
    c3=c1*c2;
    c3.output();
}
```

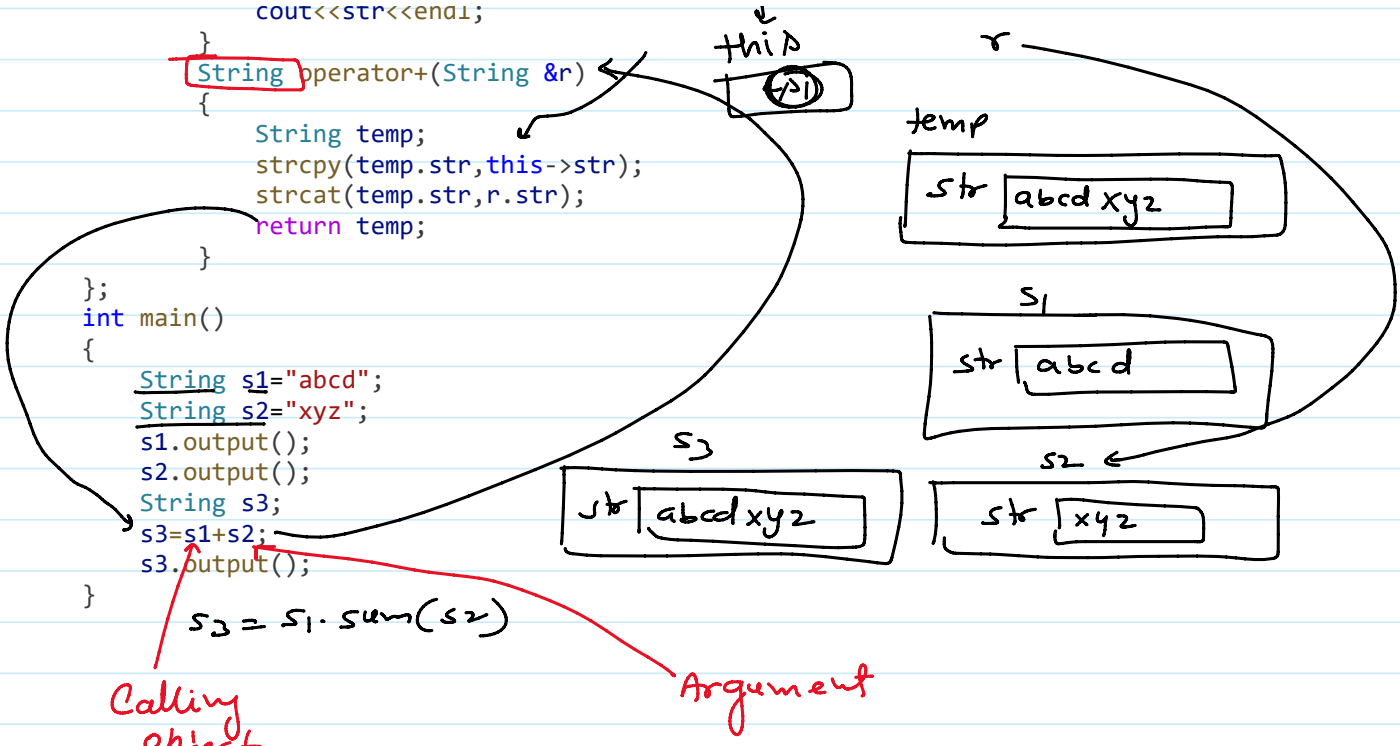


.

```cpp
#include<iostream>
#include<string.h>
using namespace std;
class String
{
    char str[100];
    public:
        String(const char s[]="")
        {
            strcpy(str,s);
        }
        void output()
        {
            cout<<str<<endl;
        }
        String operator+(String &r)
```



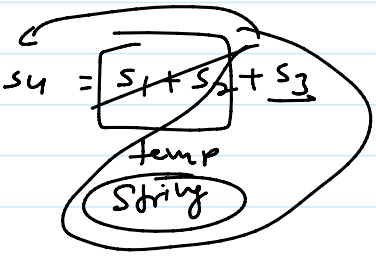
```
        cout<<str<<endl;
    }
    String operator+(String &r)
    {
        String temp;
        strcpy(temp.str,this->str);
        strcat(temp.str,r.str);
        return temp;
    }
};
int main()
{
    String s1="abcd";
    String s2="xyz";
    s1.output();
    s2.output();
    String s3;
    s3=s1+s2;
    s3.output();
}
```

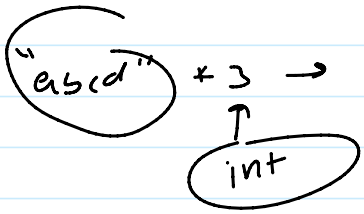


$s_3 = s_1.sum(s_2)$

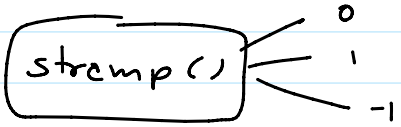
Calling object

Argument





$>$, $<$, $>=$, $<=$, $==$, $!=$



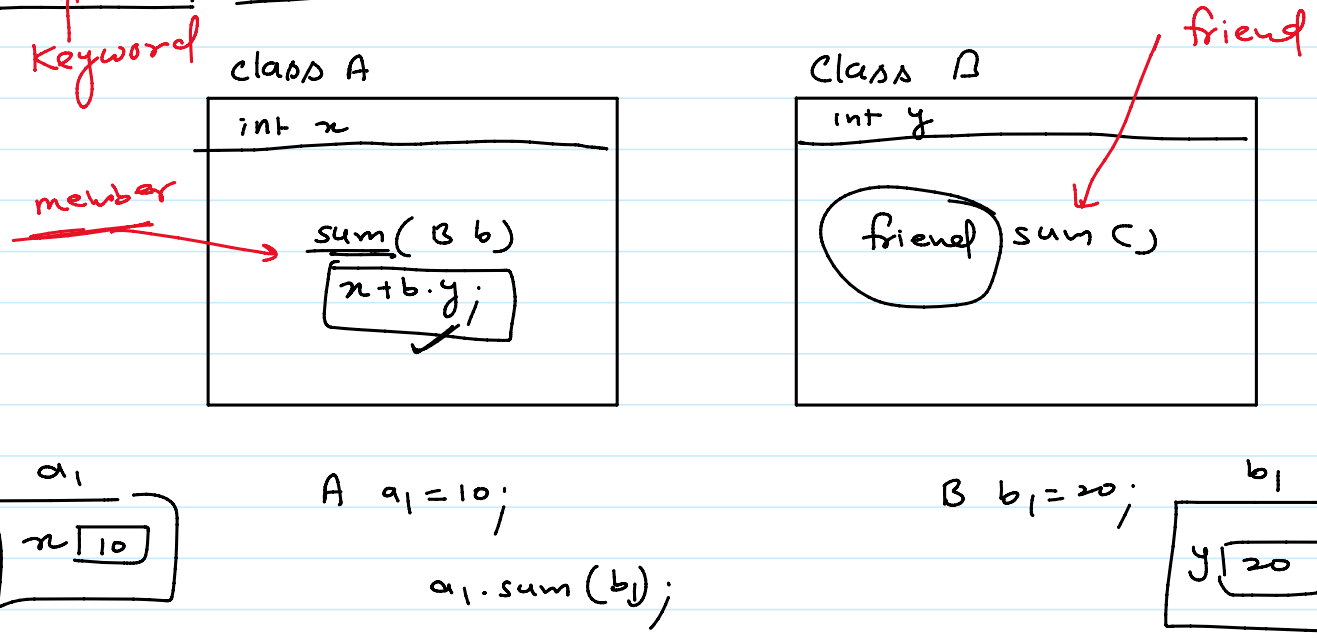

```
#include<iostream>
#include<string.h>
using namespace std;
class String
{
    char str[100];
    public:
        String(const char s[]="")
        {
            strcpy(str,s);
        }
        void output()
        {
            cout<<str<<endl;
```

```
        }
        String operator+(String &r)
        {
            String temp;
            strcpy(temp.str,this->str);
            strcat(temp.str,r.str);
            return temp;
        }
        String operator*(int repeat)
        {
            String temp;
            while(repeat--)
            {
                strcat(temp.str,str);
            }
            return temp;
        }
        bool operator >(String &r)
        {
            if(strcmp(str,r.str) == 1)
                return true;
            else
                return false;
        }
        bool operator <(String &r)
        {
            if(strcmp(str,r.str) == -1)
                return true;
            else
                return false;
        }
        bool operator >=(String &r)
        {
            if(strcmp(str,r.str) != -1)
                return true;
            else
                return false;
        }
        bool operator <=(String &r)
        {
            if(strcmp(str,r.str) != 1)
                return true;
            else
                return false;
        }
        bool operator == (String &r)
        {
            if(strcmp(str, r.str) == 0)
                return true;
            return false;
        }
        bool operator != (String &r)
        {
            if(strcmp(str, r.str) != 0)
                return true;
            return false;
        }
};
int main()
{
    String s1="abcd";
    String s2="xyz";
```

```
        s1.output();
        s2.output();
        String s3;
        s3=s1+s2;
        s3.output();
        s3=s1*3;
        s3.output();
        if(s1>s2)
            cout<<"True";
        else
            cout<<"False";
    }
```

Friend function :-

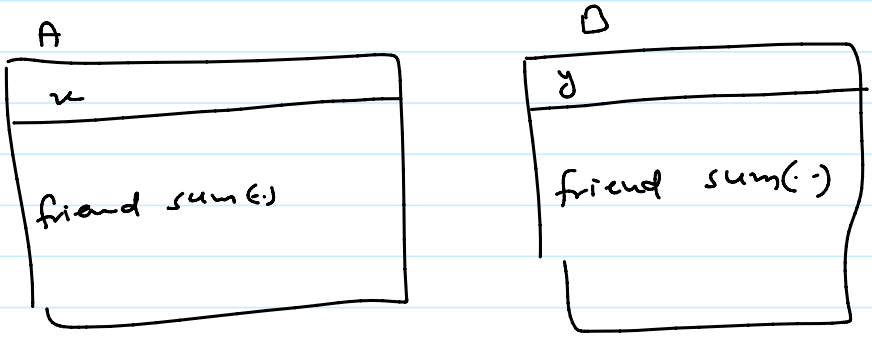
Keyword



1. Member class ← member
2. Friend class ← friend
3. function definition

```
#include<iostream>
using namespace std;
class B;
class A
{
    int x;
    public:
        A(int x1=0)
        {
            x=x1;
        }
        int sum(B);     //member func declaration
```

```
};
class B
{
    int y;
    public:
        B(int y1=0)
        {
            y=y1;
        }
        friend int A::sum(B);   //friend func declaration
};
int A::sum(B b1)    //definition
{
    return x+b1.y;
}
int main()
{
    A a1=10;
    B b1=20;
    cout<<a1.sum(b1);
}d
```



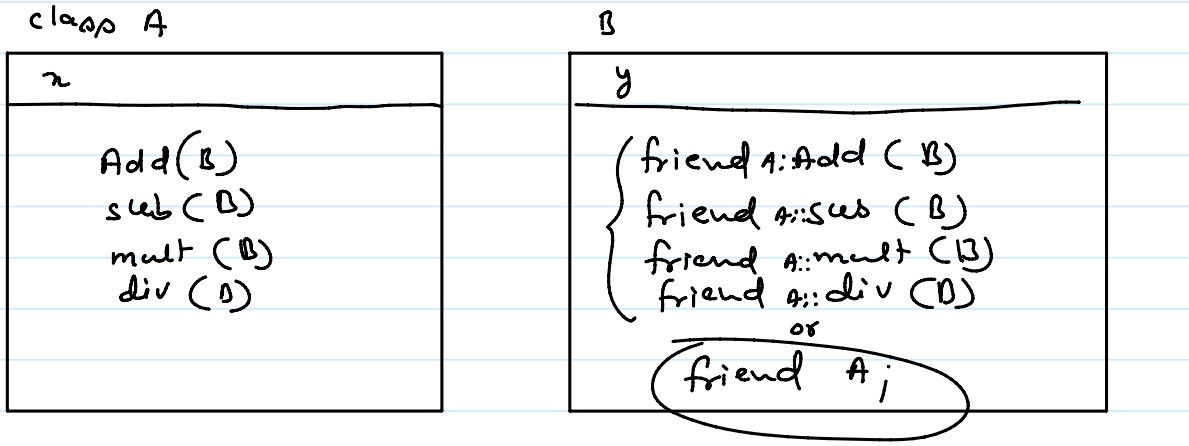

```
int sum(A a1, B b1)
{
    return a1.x + b1.y;
}
```

.

```
#include<iostream>
using namespace std;
class B;
class A
{
    int x;
    public:
        A(int x1=0)
        {
            x=x1;
        }
        friend int sum(A,B);     //friend func declaration
};
class B
{
    int y;
    public:
        B(int y1=0)
        {
            y=y1;
```

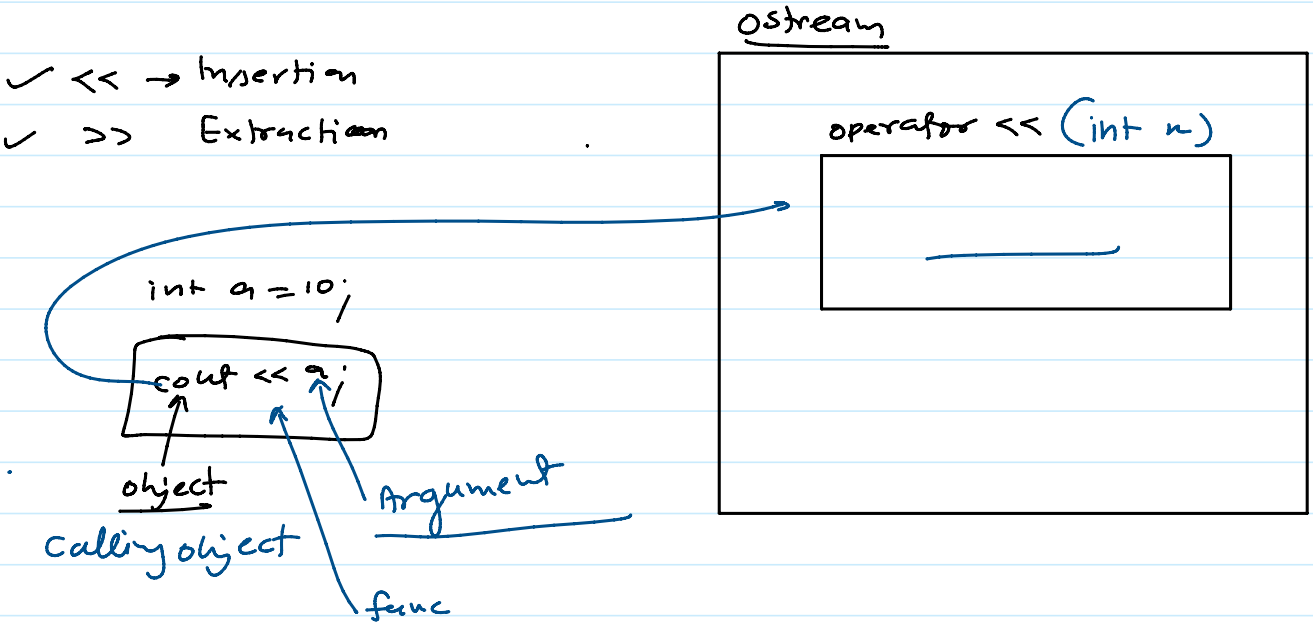
```
        }
        friend int sum(A,B);   //friend func declaration
};
int sum(A a1,B b1)    //definition
{
    return a1.x + b1.y;
}
int main()
{
    A a1=10;
    B b1=20;
    cout<<sum(a1,b1);
}
```

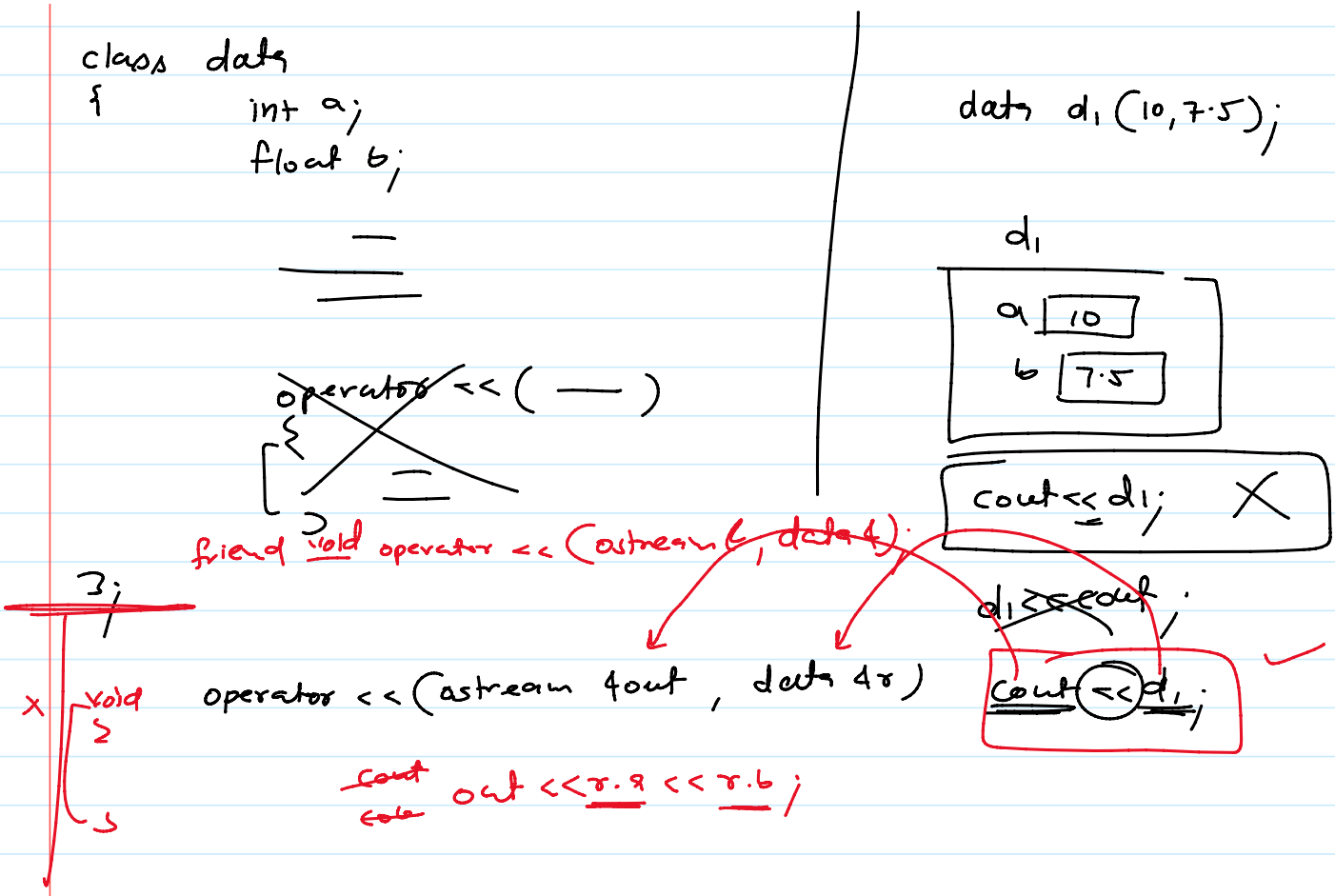
Friend class :-



✓ << → Insertion

✓ >> Extraction






```
#include<iostream>
using namespace std;
class data
{
    int a;
    float b;
    public:
        data(int a1=0, float b1=0.0)
        {
            a=a1;
            b=b1;
        }
        friend ostream& operator<<(ostream&, data&);
        friend istream& operator>>(istream &, data &);

};
ostream& operator<<(ostream &out,data &r)
{
    out<<r.a<<"\t"<<r.b<<endl;
    return out;
}
istream& operator>>(istream &in, data &r)
{
    cout<<"Enter value of A:";
    cin>>r.a;
    cout<<"Enter value of B:";
    in>>r.b;
    return in;
}
```

```
int main()
{
    data d1(10,7.5);
    data d2(20,65.25);
    cout<<d1<<d2;
    cin>>d1>>d2;
    cout<<d1<<d2;
}
```

Template :→

```
int sum (int a, int b)
{
    return a+b;
}

double sum( double a, double)
{
    return a+b;
}

float sum ( float a, float b)
{
    return a+b;
}
```

sum(10, 70) ;

sum (7.5, 8.3);

sum (2.7f, 9.5f);

```
template <typename T>
T sum (T a, T b)
{
    return a+b;
}
```

T → int : sum(10, 20)

T → double : sum (5.8, 7.7);

T → float : sum (5.7f, 2.8f);

```
#include<iostream>
using namespace std;
template<typename T>
T sum(T a, T b)
{
    return a+b;
}
int main()
{
    cout<<sum(10,20)<<endl;
    cout<<sum(8.5,7.3)<<endl;
    cout<<sum(4.5f,9.3f)<<endl;
    return 0;
}
```

.

```cpp
#include<iostream>
using namespace std;
template<typename T1, typename T2>
double sum(T1 a, T2 b)
{
    return a+b;
}
int main()
{
    cout<<sum(10,20.5)<<endl;
    cout<<sum(8.5f,7.3)<<endl;
    cout<<sum(4.5f,9)<<endl;
    return 0;
}
```

.

```cpp
#include<iostream>
using namespace std;
template<typename T>
class Array
{
    T arr[100];
    int n;
    public:
        void input()
        {
            cout<<"Enter no of elements:";
            cin>>n;
            for(int i=0;i<n;i++)
            {
                cout<<"Enter value of "<<i+1<<" element:";
                cin>>arr[i];
            }
        }
        void output()
        {
            for(int i=0;i<n;i++)
            {
                cout<<arr[i]<<" ";
            }
            cout<<endl;
        }
};
int main()
{
    Array<int> a1;
    a1.input();
    a1.output();
    Array<float> a2;
    a2.input();
    a2.output();

}
```

Inheritance :-