
OVERLOADING

```
                         |
        ----------------------------------------
        |                                      |
FUNCTION  OVERLOADING             OPERATOR   OVERLOADING
```

fun. name -> same
argument -> different

+ , - , * , / , ...

sum (int , int); -------------> t = a + b ; // integer

sum (float , float)

t . sum (p,q); -------------> t = p + q; // complex no.

+ -> int , float , user-defined data type

OPERATOR OVERLOADING

A. OPERATOR MEMBER FUNCTION

B. OPERATOR FRIEND FUNCTION

OVERLOADING ASSIGNMENT (=) OPERATOR

```
        //    COPY  COMPLEX  NO.

#include<iostream>
using namespace std;
class     complex
 {
          private :        int  a , b;

          public    :      complex()
                           {
                           }
                            complex( int  x , int  y)
                            {
                                 a = x ;  b = y ;
                            }

                            void    out()
                            {
                                    cout<< a << " +i " << b << endl;
                            }

                            void    operator = ( complex  x )
                            {
                                    a =  x . a;        // a = 3
                                    b =  x . b;       //  b = 2
                           }
```

```
              ~complex() {     }
};
int   main()
{
        complex  p(3,2);
        complex  q;

        q = p ;     // q . operator = (p);

        p . out();  // 3 + i2
        q . out(); //  3 + i2
}
```

----

1. passing   object into function

```
copy  constructor    // compiler

complex  x = p;  // complex  x(p);
```

2. returning object

```
void  opertor= ( complex x )     // compiler

t  =  sum (p,q);
t  =   w ;  //  t.operator = (w)
```

OVERLOADING UNARY(-) OPERATOR

```
void      operator- ()
{
          a = -a ;  // a = -3

          b = -b ;  // b = -2
}

int    main()
{
       complex  p (3,2);

       p . out();  // 3 , 2

      -p ; // -p  or  p-   // p . operator - () ;

       p . out();  // -3, -2

}
```

OVERLOADING INCREMENT ++ AND DECREMENT -- OPERATOR

```
//  p ++ ;  // p . operator++ ()

        void    operator ++()
        {
                a = a + 1 ;  // a++;

                b = b + 1 ;
        }
        void    operator --()
        {
                a = a - 1;

                b = b - 1;
        }
```

OVERLOADING AIRTHMATIC ASSIGNMENT += AND   -= OPERATOR

```
//    p+= 5;  // p . operator += (5);

        void   operator += ( int  t )
        {
                a =  a + t ;

                b =  b + t ;
        }
```

```
void   operator -= ( int  t )
{
          a =  a - t;

          b =  b - t;
}

int    main()
{
        complex  p(3,2);

         p += 5;     //  p . operator += (5)    // 8, 7

        p -= 2;     //   p . operator -= (2)   // 6, 5

        p++ ;      //    p . operator ++ ()   // 5~~, 4~~  7.6

        p-- ;        //    p . operator --()   // 4~~ , 3~~ ;   6.5

        p. out();  //  ~~4 , 3~~    6.5

}
```

OVERLOADING BINARY(+) OPERATOR

ADDITION OF TWO COMPLEX NOS USING OVERLOADING '+' OPERATOR

```
complex   operator+  ( complex  x )
{
              complex  w;

              w . a = a + x . a;  // w . a = 3 + 2 = 5

              w . b = b + x . b;  // w . b = 2 + 4 = 6

              return(w);
}

int       main()
{
          complex  p(3,2);
          complex  q(2,4);
          complex  t;

          t = p + q ;  // t = p . operator+ (q);
                                     ⇓
                                     w

          p . out();
          q . out();
          t . out(); // 5 + i 6
}
```

```
//addition of two complex nos using overloading '+' operator

#include<iostream>
using namespace std;
class complex
{
private: int a,b;
public :
   complex(int x=0,int y=0)
   {
         a=x;b=y;
   }
   void out()
   {
         cout<<a<<"+i"<<b<<endl;
   }
  complex   operator + ( complex  x )
   {
                         complex w;

                         w.a = a + x.a ;  // w.a = 3 + 2 = 5

                         w.b = b + x.b ; //  w.b = 2 + 4 = 6

                         return(w);
    }
};
```

```
    int   main()
    {
                complex  p(3,2);
                complex  q(2,4);
                complex t ;

                t = p + q ;  // t = p . operator + ( q );

                p . out();   // 3 +i 2
                q . out(); //  2 +i 4
                t . out(); //   5 + i 6
    }
 /*
 -----------------------------------------------------------------
                a + i b        // first complex
                x.a + i x.b    //  second complex
-----------------------------------------------------------------


            a + ib        // first complex

        *   x.a + ix.b    //  second complex
_________________________________________________________________

a * x.a + ib * x.a + ix.b * a - b * x.b
_________________________________________________________________

( a * x.a - b * x.b ) +i  ( b * x.a + a * x.b)
  real                      imag
```

OVERLOADING * OPERATOR

```
complex   operator * ( complex  x )
{
                complex w;

                w.a = ( a * x.a - b * x.b ) ;

                w.b = ( b * x.a + a * x.b)

                return( w );
}
```

$$\frac{\text{first complex}}{\text{second complex}} \quad \frac{(a + ib)}{(x.a + ix.b)} * \frac{(x.a - i*x.b)}{(x.a - i*x.b)}$$

$$\frac{(a * x.a + b * x.b)}{(x.a * x.a + x.b * x.b)} + i \frac{(b * x.a - a * x.b)}{(x.a * x.a + x.b * x.b)}$$

real imag.

OVERLOADING / OPERATOR

```
complex  operator / ( complex  x )
{
                complex w;

                w.a = ( a * x.a + b * x.b ) /(x.a * x.a + x.b * x.b);

                w.b = ( b * x.a - a * x.b) / (x.a * x.a + x.b * x.b);


                return(w);
}
```

ADDITON   OF TWO STRING USING OPERATOR (+) MEMBER FUNCTION

```
class  string
{
               private :    char  a[40];

               public  :
                                .
                                .
```

```
            string   operator + ( string  x )
            {
                    string  w;

                    strcpy( w.a , a ) ;

                     strcat( w.a , x.a);

                    return(w);

            }

    };

*/
```