# ऑपरेटर ओवरलोडिंग

---

## पाठ्यपुस्तक के प्रश्न

### वस्तुनिष्ठ प्रश्न

**प्रश्न 1. किस ऑपरेटर को ओवरलोड किया जा सकता है?**
(अ) स्कोप रिजोल्यूशन ऑपरेटर (::)
(ब) क्लास मेंम्बर एक्सेस ऑपरेटर (., .*)
(स) बाइनरी प्लस ऑपरेटर (+)
(द) कंडिशनल ऑपरेटर (?:)

**उत्तर:** (स) बाइनरी प्लस ऑपरेटर (+)

**प्रश्न 2. बाइनरी ऑपरेटर को ओवरलोड करने के लिए ऑपरेटर फंक्शन मेम्बर फंक्शन के रूप में कितने आरग्यूमेन्ट लेता है?**
(अ) दो आरग्यूमेन्ट
(ब) एक आरग्यूमेन्ट
(स) शून्य आरग्यूमेन्ट
(द) इनमें से कोई नहीं

**उत्तर:** (ब) एक आरग्यूमेन्ट

**प्रश्न 3. यूनरी ऑपरेटर को ओवरलोड करने के लिए ऑपरेटर फंक्शन फ्रेंड फंक्शन के रूप में कितने आरग्यूमेन्ट लेता है?**
(अ) दो आरग्यूमेन्ट
(ब) एक आरग्यूमेन्ट
(स) शून्य आरग्यूमेन्ट
(द) इनमें से कोई नहीं

**उत्तर:** (ब) एक आरग्यूमेन्ट

**प्रश्न 4. किस ऑपरेटर को फ्रेंड फंक्शन से ओवरलोड नहीं कर सकते हैं?**
(अ) = असाइनमेंट ऑपरेटर
(ब) () फंक्शन कॉल ऑपरेटर
(स) [ ] सबस्क्रिप्टिंग ऑपरेटर
(द) ये सभी

**उत्तर:** (द) ये सभी

## अतिलघु उत्तरीय प्रश्न

### प्रश्न 1. ऑपरेटर ओवरलोडिंग किसे कहा जाता है?

**उत्तर-** किसी ऑपरेटर को एक डेटा टाइप के लिए विशेष मीनिंग देना ऑपरेटर ऑवरलोडिंग कहा जाता है।

### प्रश्न 2. ऑपरेटर फंक्शन का प्रोटोटाईप लिखो।

**उत्तर-** ऑपरेटर फंक्शन का प्रोटोटाईप इस प्रकार होता है

```
return_typeclass_name :: operator op (arguments list)
{
Function body
}
```
जहाँ Operator एक कीवर्ड है और op एक ऑपरेटर है जिसे ओवरलोड किया जाना है।

### प्रश्न 3. उन ऑपरेटर्स का नाम लिखो जिनको ओवरलोड नहीं किया जा सकता है?

**उत्तर-** C++ में निम्न ऑपरेटर्स को ओवरलोड़ नहीं किया जा सकता है

- क्लास मेबर एक्सेस ऑपरेटर (., . *),
- स्कोप रिजोल्यूशन ऑपरेटर (::),
- साइज ऑपरेटर (Sizeof),
- कंडिशनल ऑपरेटर (?:)

## लघु उत्तरीय प्रश्न

### प्रश्न 1. ऑपरेटर फंक्शन मेबर फंक्शन के रूप में और ऑपरेटर फंक्शन फ्रेंड फंक्शन के रूप में दोनों में अन्तर का वर्णन कीजिए।

**उत्तर-** मेम्बर फंक्शन, कक्षा (class) में परिभाषित होता है और class का member होता है। यह एक public, private या protected फंक्शन हो सकता है। यह यूनरी ऑपरेटर के लिए कोई आरग्यूमेन्ट नहीं लेता और बाइनरी ऑपरेटर के लिए एक आरग्यूमेन्ट लेता है। फ्रेंड फंक्शन हमेशा सभी सम्बद्ध classes के सभी 'मेम्बर्स (members) को access करने में सक्षम (capable) होता है। फ्रेंड फंक्शन को परिभाषित

(define) करने के लिए हम “friend” कीवर्ड का उपयोग करते हैं। यह यूनरी ऑपरेटर के लिए एक आरग्यूमेन्ट लेता है और बाइनरी ऑपरेटर के लिए दो आरग्यूमेन्ट लेता है।

निम्नलिखित उदाहरणों से इन दोनों में अन्तर को स्पष्टतया समझा जा सकता है
यूनरी ऑपरेटर को मेम्बर फंक्शन से ओवरलोड करना
हम पोस्टफिक्स इंक्रिमेंट ऑपरेटर (++) को उदाहरण के लिए लेते हैं। यह केवल एक ऑपरेंड लेता है और इसकी वेल्यू को एक बढ़ा देता है जब बेसिक डेटा टाईप के साथ प्रयोग किया जाता है। हम इस ऑपरेटर को ओवरलोड करेंगे ताकि इसका प्रयोग एक ऑब्जेक्ट के साथ किया जा सके और इस ऑब्जेक्ट के हर एक डेटा आइटम की वेल्यू को एक बढ़ा सके।

प्रोग्राम-पोस्टफिक्स इंक्रिमेंट ऑपरेटर को ओवरलोड करना

```
#include<iostream>
using namespace std;
class point
{
int x, y;
public:
void getdata (int a, int b)
{
x=a;
y=b;
}
void show (void)
{
cout<<"x="<<x;
cout<<"y="<<y<<"\n";
}
void operator++ (int)
x++;
y++;
}
};
int main()
{
point p;
p.getdata (5,8);
cout<<"p:";
p.show();
```

```
p++; // invoke operator function
cout<<"p++";
p. show ();
return 0;
{
```
प्रोग्राम का आउटपुट होगा
```
p:x=5 y=8
p++:x=6 y=9
```

ऑपरेटर फंक्शन में int का प्रयोग यह दर्शाता है कि हम पोस्टफिक्स इंक्रिमेंट ऑपरेटर को ओवरलोड कर रहे हैं न कि प्रिफिक्स इंक्रिमेंट ऑपरेटर को।

यूनरी ऑपरेटर को फ्रेंड फंक्शन से ओवरलोड करना।
हम प्रिफिक्स डिक्रिमेंट ऑपरेटर (–) को उदाहरण के लिए लेते हैं जो एक ऑपरेंड लेता है और उसकी वेल्यू को एक कम कर देता है जब बेसिक डेटा टाईप के साथ प्रयोग किया जाता है। हम इस ऑपरेटर को ओवरलोड करेंगे ताकि इसका प्रयोग ऑब्जेक्ट के साथ किया जा सके और इस ऑब्जेक्ट के हर एक डेटा आइटम की वेल्यू को एक कम कर देगा।

प्रोग्राम-प्रिफिक्स डिक्रिमेंट ऑपरेटर को ओवरलोड करना

```
#include<iostream>
using namespace std;
class point
{
int x,y;
public
void getdata (int a, int b).
{
x = a;
y = b;
}
void show (void)
{
cout<<"x = " <<x;
cout<<="y= <<y<<"/n";
}
friend void operator-- (point &s)
{
s.x =s.x-1;
```

```
s.y=s.y-1;
}
};
int main()
{
point p;
p.getdata (7, 10);
cout<<"p:";
p. show ();.
--p;
cout<<"--p:";
p.show ();
return 0;
}
```

प्रोग्राम का आउटपुट होगा

```
p:x=7 y=10
p++:x=6 y=9
```

ध्यान दें कि ऑपरेटर फंक्शन में आरग्यूमेंन्ट रेफरेंस के द्वारा भेजा गया है। अगर हम वेल्यू के द्वारा भेजते तो यह काम नहीं करता क्योंकि जो बदलाव ऑपरेटर फंक्शन में किये गये हैं वो main () फंक्शन में प्रतिबिम्बित नहीं होंगे।

## निबंधात्मक प्रश्न

**प्रश्न 1. क्लास के ऑब्जेक्ट को ऋणात्मक बनाने के लिए यूनरी माइनस ऑपरेटर को ओवरलोड करने का प्रोग्राम फ्रेंड फंक्शन का उपयोग करके लिखो।**

**उत्तर-** प्रोग्राम

```
#include<iostream. h>
using namespace std;
class point
{
int x, y;
public:
void getdata (int a, int b)
{
x = a;
y = b;
```

```
}
void show (void)
{
cout<<"x = "<<x;
cout << "y = " << y «< "\n";
}
friend void operator -- (point &s)
{
s.x = s.x-1;
s.y = s.y-1;
}
};
int main()
}
point p;
p.getdata (7,10);
cout<<"p:" ;
p.show();
--P;
cout<<"--p:";
p.show();
return 0;
}
```

प्रोग्राम का आउटपुट होगा

p:x=7 y=10

p++:x=6 y=9

**प्रश्न 2. बाइनरी प्लस ऑपरेटर द्वारा दो स्ट्रिंग को जोड़ने के लिए ओवरलोड करने का प्रोग्राम मेम्बर फंक्शन का उपयोग करके लिखो।**

**उत्तर:**

```
# include<conio.h>
# include<string.h>
# include<iostream.h>
class string
{
public :
char *s;
```

```
int size;
void getstring (char *str)
}
Size = strlen(str);
S = new char[size];
strcpy (s, str);
{
void operator+ (string);
};
void string :: operator+(string ob)
{
size = size + ob. size;
s = new char [size];
strcat(s, ob.s);
cout <<"/n Concatnated string is : "<<s;
}
void main()
{
string ob1, ob2;
char *string1, *string2;
clrscr ();
cout<<"\n Enter First String :" ;
cin>>string1;
ob1 .getstring (string1);
cout<<"\n Enter Second String :";
cin>>string 2;
ob 2.getstring (string 2);
ob1+ob 2;
getch ();
}
```

प्रोग्राम का आउटपुट होगा

```
Enter First String : Computer
Enter Second String : Programming
Concatenated String is : Computer programming
```

# अन्य महत्त्वपूर्ण प्रश्न

## अतिलघु उत्तरीय प्रश्न

**प्रश्न 1. जब हम किसी ऑपरेटर को ओवरलोड करते हैं तो क्या इसका मूलरूप बदलता है?**

**उत्तर-** जब हम किसी ऑपरेटर को ओवरलोड करते हैं, तो इसका मूलरूप नहीं बदलता है, वह बरकरार रहता है।

**प्रश्न 2. ऑपरेटर फंक्शन क्या होता है?**

**उत्तर-** किसी ऑपरेटर को अतिरिक्त मिनिंग देने के लिए हम एक विशेष फंक्शन काम में लेते हैं जिसे ऑपरेटर फंक्शन कहा जाता है।

**प्रश्न 3. C++ में किसी एक ऑपरेटर का नाम बताइये जिसे ओवरलोड नहीं किया जा सकता।**

**उत्तर-** C++ में कंडीशनल ऑपरेटर (?:) को ओवरलोड नहीं किया जा सकता है।

**प्रश्न 4. C++ में किस ऑपरेटर को फ्रेंड फंक्शन से ओवरलोड नहीं कर सकते?**

**उत्तर-** C++ में असाइनमेंट ऑपरेटर (=) को फ्रेंड फंक्शन से ओवरलोड नहीं कर सकते हैं।

**प्रश्न 5. बाइनरी ऑपरेटर के लिए, बायीं तरफ का ऑपरेंड क्या करता है?**

**उत्तर-** बाइनरी ऑपरेटर के लिए, बायीं तरफ का ऑपरेंड ऑपरेटर फंक्शन को कॉल करने का काम करता है।

**प्रश्न 6. यूनरी ऑपरेटर को ओवरलोड करने के लिए ऑपरेटर फंक्शन मेम्बर फंक्शन के रूप में कितने आरग्यूमेंट लेता है?**

**उत्तर-** यूनरी ऑपरेटर को ओवरलोड करने के लिए ऑपरेटर फंक्शन मेम्बर फंक्शन के रूप में एक आरग्यूमेंट लेता

## लघु उत्तरीय प्रश्न

**प्रश्न 1. C++ में किन ऑपरेटर को ओवरलोड़ नहीं कर सकते?**

**उत्तर-** C++ में निम्न ऑपरेटर को ओवरलोड नहीं कर सकते

- क्लास मेम्बर एक्सेस ऑपरेटर (., .*)
- स्कोप रिजोल्यूशन ऑपरेटर (::)
- साइज ऑपरेटर (sizeof)
- कंडीशनल ऑपरेटर (?:)

**प्रश्न 2. ऑपरेटर फंक्शन एक क्लास का मेम्बर फंक्शन या फ्रेंड फंक्शन होना चाहिए। इस कथन में क्या फर्क है? समझाइए।**

**उत्तर-** ऑपरेटर फंक्शन एक क्लास का मेम्बर फंक्शन या फ्रेंड फंक्शन होना चाहिए। उनमें फर्क है कि मेम्बर फंक्शन यूनरी ऑपरेटर के लिए आरग्यूमेन्ट नहीं लेता है और बाइनरी ऑपरेटर के लिए एक आरग्यूमेन्ट लेता है जबकि फ्रेंड फंक्शन यनरी ऑपरेटर के लिए एक आरग्यूमेन्ट लेता है और बाइनरी ऑपरेटर के लिए दो ऑपरेटर लेता है।

**प्रश्न 3. किन ऑपरेटर्स को फ्रेंड फंक्शन से ओवरलोड नहीं किया जा सकता है?**

**उत्तर-** निम्नलिखित ऑपरेटर्स को फ्रेंड फंक्शन से ओवरलोड नहीं किया जा सकता है

- असाइनमेंट ऑपरेटर =
- फंक्शन कॉल ऑपरेटर ()
- सब्सक्रिप्टींग ऑपरेटर []
- क्लास मेंबर एक्सेस ऑपरेटर →

## निबंधात्मक प्रश्न

**प्रश्न 1. बाइनरी प्लस ऑपरेटर (+) को ओवरलोड करने का प्रोग्राम लिखिए।**

**अथवा**

**बाइनरी प्लस ऑपरेटर (+) द्वारा दों मैट्रिक्स को जोड़ने का प्रोग्राम लिखिए।**

**उत्तर-** बाइनरी प्लस ऑपरेटर (+) द्वारा दो मैट्रिक्स को जोड़ने का प्रोग्राम।

```
#include<iostream>
using namespace std;
class matrix
{
```

```
int mat [2] [2] ;
public:
void getmatrix (void);
matrix operator + (matrix);
void showmatrix (void);
};
void matrix :: getmatrix (void)
{
for (int i=0;i<2;i++)
for (int j=0;j<2;j++)
{
cout<<"Enter the number:";
cin>>mat[i] [j]:
}
}
matrix matrix::operator+ (matrix m)
{
matrix temp;
for (int i = 0;i<2;i++)
for (int j = 0:j<2;j++)
temp. mat [i] [j] =mat [i] [j] +m. mat [i] [j];
return temp;
}
void matrix:: showmatrix (void)
{
for(int i=0; i<2; i++)
{
for (int j=0; j<2; j++)
cout<<mat [i] [j] <<"\t";
cout<<"\n";
}
}
int main()
}
matrix m1, m2, m3;
m1. getmatrix ();
m2.getmatrix();
m3=m1+m2;
cout<<"matrix m1:\n";
m1.showmatrix();
```

```
cout<<"matrix m2:\n";
m2.showmatrix ();
cout<<"Resultant matrix:\n";
m3.showmatrix();
return 0;
}
```

प्रोग्राम का आउटपुट होगा

```
Enter the number: 2
Enter the number: 3
Enter the number: 1
Enter the number: 4
Enter the number: 6
Enter the number: 7
Enter the number: 8
Enter the number: 9
matrix m1:
2 3
1 4
matrix m2:
6 7
8 9
Resultant matrix:
8 10
9 13
```

**प्रश्न 2. बाइनरी प्लस ऑपरेटर को फ्रेंड फंक्शन से ओवरलोड करने का प्रोग्राम लिखिए।**

**अथवा**

**बाइनरी प्लस ऑपरेटर से दो कॉम्पलेक्स संख्याओं को जोड़ने का प्रोग्राम लिखिए।**

**उत्तर-** बाइनरी प्लस ऑपरेटर से दो कॉम्पलेक्स संख्याओं को जोड़ने का प्रोग्राम।

```
#include<iostream>
using namespace std;
class complex
{
float real;
float imag;
```

```
public:
void input (float x,float y)
{
real=x;
imag=y;
}
friend complex operator+ (complex a, complex b)
{
complex c;
c.real=a.real+b.real;
c.imag=a.imag+b.imag;
return c;
}
void show(void)
{
cout<<real<<"+i"<<imag<<"\n";
}
};
int main()
{
complex c1,c2,c3;
c1.input (1.6,6.2);
c2.input (2.3,3.4);
c3=c1+c2; //invoke operator function
cout<<"C1=";
c1.show();
cout<<"C2=";
c2.show();
cout<<"C3=";
c3.show();
return 0;
}
```

प्रोग्राम को आउटपुट होगा

```
C1 = 1.6+i6.2;
C2 = 2.3+i3.4;
C3 = 3.9+i9.6;
```