

```

/*
 * advanced programming
 * alireza akhavan Pour
 * akhavan@alirezaWeb.com
 * date: 1394/12/15
 */
int main()
{
    cout<< "In the name of God";
    Lecture_11();
    return 0;
}

string Lecture_11()
{
    cout << "1.operator overloading" << endl;
    cout << "2.Composition" << endl;
    cout << "3.association" << endl;
    return ":");
}

```

1



Shahid Rajaee Teacher
Training University

```

#include <iostream>
using namespace std;

class CVector {
public:
    int x,y;
    CVector () {};
    CVector (int a,int b) : x(a), y(b) {}
    CVector sum (const CVector& );
};

CVector CVector::sum(const CVector& param) {
    CVector temp;
    temp.x = x + param.x;
    temp.y = y + param.y;
    return temp;
}

int main () {
    CVector foo (3,1);
    CVector bar (1,2);
    CVector result;
    result = foo.sum(bar);
    cout << result.x << ',' << result.y << '\n';
    return 0;
}

```

میتوان کاری کرد که به جای این
 !? Result=foo+sum; عبارت نوشت

2

– به عنوان تابع عضو کلاس Operator overloading

```

1 // overloading operators example
2 #include <iostream>
3 using namespace std;
4
5 class CVector {
6 public:
7     int x,y;
8     CVector () {};
9     CVector (int a,int b) : x(a), y(b) {}
10    CVector operator + (const CVector&);
11 };
12
13 CVector CVector::operator+ (const CVector& param) {
14     CVector temp;
15     temp.x = x + param.x;
16     temp.y = y + param.y;
17     return temp;
18 }
19
20 int main () {
21     CVector foo (3,1);
22     CVector bar (1,2);
23     CVector result;
24     result = foo + bar;
25     cout << result.x << ',' << result.y << '\n';
26     return 0;
27 }
```

[Try Online](#)

رو عبارت زیر برآورند:

c = a + b;
c = a.operator+ (b);

1-overloading-member.cpp

3

Overloadable operators

+	-	*	/	=	<	>	+=	-=	*=	/=	<<	>>
<=	>=	==	!=	<=	>=	++	--	%	&	^	!	
~	&=	^=	=	&&		%=	[]	()	,	->*	->	new
delete	new []	delete []										

Expression	Operator	Member function	Non-member function
@a	+ - * & ! ~ ++ --	a::operator@()	operator@(a)
a@	++ --	a::operator@(int)	operator@(a,int)
a@b	+ - * / % ^ & < > == != <= >= << >> && ,	a::operator@(b)	operator@(a,b)
a@b	= += -= *= /= %= ^= &= = <=>= []	a::operator@(b)	-
a(b,c...)	()	a::operator()(b,C...)	-
a->b	->	a::operator->()	-
(TYPE) a	TYPE	a::operator TYPE()	-

4

– به عنوان تابع غیرعضو Operator overloading

```

1 // non-member operator overloads
2 #include <iostream>
3 using namespace std;
4
5 class CVector {
6 public:
7     int x,y;
8     CVector () {}
9     CVector (int a, int b) : x(a), y(b) {}
10};
11
12
13 CVector operator+ (const CVector& lhs, const CVector& rhs) {
14     CVector temp;
15     temp.x = lhs.x + rhs.x;
16     temp.y = lhs.y + rhs.y;
17     return temp;
18}
19
20 int main () {
21     CVector foo (3,1);
22     CVector bar (1,2);
23     CVector result;
24     result = foo + bar;
25     cout << result.x << ',' << result.y << '\n';
26     return 0;
27}

```

[Try Online](#)

5

2-overloading-nonmember.cpp

یاداوری: کاربرد **this** – اشاره گر به شئ جاري

```

1 // example on this
2 #include <iostream>
3 using namespace std;
4
5 class Dummy {
6 public:
7     bool isitme (Dummy& param);
8 };
9
10 bool Dummy::isitme (Dummy& param)
11 {
12     if (&param == this) return true;
13     else return false;
14 }
15
16 int main () {
17     Dummy a;
18     Dummy* b = &a;
19     if ( b->isitme(a) )
20         cout << "yes, &a is b\n";
21     return 0;
22 }

```

yes, &a is b

[Try Online](#)

3-this.cpp

6

کاربرد **this** در سربارگذاری عملگر =

```

1 CVector& CVector::operator= (const CVector& param)
2 {
3     x=param.x;
4     y=param.y;
5     return *this;
6 }
```

7

```
Rectangle& Rectangle::operator= (const Rectangle& param)
```

```
{
    width=param.width;
    height=param.height;
    return *this;
}
```

چرا عملگر = نیاز به خروجی دارد؟!

```

int main () {
    Rectangle rect (3,4);
    Rectangle rectb;
    Rectangle rectc;
    rectb.set(4,2);
    cout << "rect area: " << rect.area() << endl;
    cout << "rectb area: " << rectb.area() << endl;
    rectc=rect=rectb;
    cout << "rect area: " << rect.area() << endl;
    cout << "rectb area: " << rectb.area() << endl;

    system("pause");
    return 0;
}
```

دلیل:

زیرا خروجی حاصل از
انتساب **rect=rectb** به
عنوان ورودی
Rectc.operator=()
نیاز است

4-equal-rectangle.cpp

8

منابع این جلسه

✓ <http://www.cplusplus.com/doc/tutorial/templates/>