

# Basics: From C to C++

**Computer Programming for Engineers (DSAF003-42)**

Fall, 2021

## **Practice 10 : Operator overloading**

**Instructor:**

Youngjoong Ko (nlp.skku.edu)

# Member function Operator overloading

```
1  #include <iostream>
2  using namespace std;
3
4  class Point{
5  private:
6      int xpos, ypos;
7  public:
8      Point(int x, int y):xpos(x),ypos(y){}
9      void show() const{
10         cout<<"("<<xpos<<","<<ypos<<")"<<endl;
11     }
12     const Point operator+(const Point &ref){
13         Point pos(xpos+ref.xpos,ypos+ref.ypos);
14         return pos;
15     }
16 };
17 int main(){
18     Point pos1(3,4);
19     Point pos2(10,20);
20     Point pos3=pos1.operator+(pos2);
21     Point pos4=pos1+pos2;
22
23     pos1.show();    (3,4)
24     pos2.show();    (10,20)
25     pos3.show();    (13,24)
26     pos4.show();    (13,24)
27     return 0;
28 }
```

# Global function Operator overloading 1

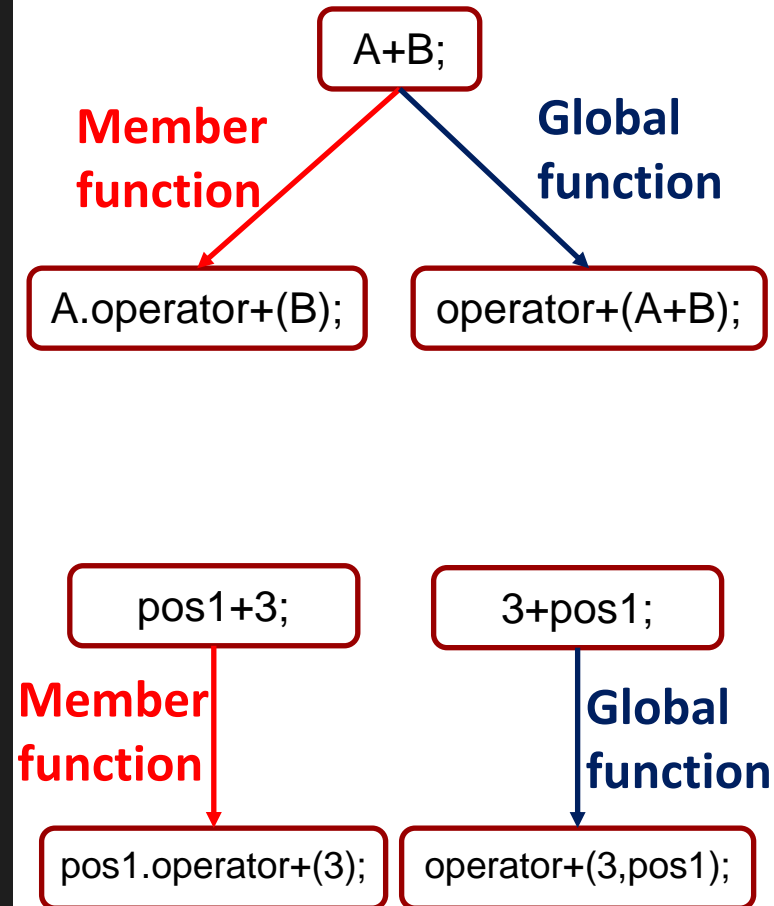
```
1  #include <iostream>
2  using namespace std;
3  class Point{
4  private:
5      int xpos, ypos;
6  public:
7      Point(int x, int y):xpos(x),ypos(y){}
8      void show() const{
9          cout<<"("<<xpos<<"", "<<ypos<<"")"<<endl;
10     }
11     int getX() const{
12         return xpos;
13     }
14     int getY() const{
15         return ypos;
16     }
17 };
18 const Point operator+(const Point &ref1, const Point &ref2){
19     const Point pos(ref1.getX()+ref2.getX(),ref1.getY()+ref2.getY());
20     return pos;
21 }
22 int main(){
23     Point pos1(3,4);
24     Point pos2(10,20);
25     Point pos3=pos1+pos2;
26     Point pos4=operator+(pos1, pos2);
27
28     pos1.show();    (3,4)
29     pos2.show();    (10,20)
30     pos3.show();    (13,24)
31     pos4.show();    (13,24)
32     return 0;
33 }
```

# Global function Operator overloading 2

```
1  #include <iostream>
2  using namespace std;
3  class Point{
4  private:
5      int xpos, ypos;
6  public:
7      Point(int x, int y):xpos(x),ypos(y){}
8      void show() const{
9          cout<<"("<<xpos<<" "<<ypos<<"")<<endl;
10     }
11     friend const Point operator+(const Point &ref1, const Point &ref2);
12 };
13 const Point operator+(const Point &ref1, const Point &ref2){
14     Point pos(ref1.xpos+ref2.xpos,ref1.ypos+ref2.ypos);
15     return pos;
16 }
17 int main(){
18     Point pos1(3,4);
19     Point pos2(10,20);
20     Point pos3=pos1+pos2;
21     Point pos4=operator+(pos1, pos2);
22
23     pos1.show();    (3,4)
24     pos2.show();    (10,20)
25     pos3.show();    (13,24)
26     pos4.show();    (13,24)
27     return 0;
28 }
```

# Member and Global function example

```
1  #include <iostream>
2  using namespace std;
3  class Point{
4  private:
5      int xpos, ypos;
6  public:
7      Point(int x, int y):xpos(x),ypos(y){}
8      void show() const{
9          cout<<"("<<xpos<<" "<<ypos<<"")<<endl;
10     }
11     const Point operator+(int num){
12         Point pos(xpos+num,ypos+num);
13         return pos;
14     }
15     friend const Point operator+(int num, const Point &ref);
16 };
17 const Point operator+(int num, const Point &ref){
18     const Point pos(num+ref.xpos,num+ref.ypos);
19     return pos;
20 }
21 int main(){
22     Point pos1(3,4);
23     int num = 3;
24     Point pos2=pos1+3;
25     Point pos3=3+pos1;
26
27     pos2.show(); (6,7)
28     pos3.show(); (6,7)
29     return 0;
30 }
```



# Exercise 1

- Define Point class and global function

```
int main(){
    Point pos1(3,4);
    Point pos2(10,20);
    Point pos3=pos2-pos1;
    Point pos4=20-pos3;
    Point pos5=-pos4;

    pos3.show();
    pos4.show();
    pos5.show();
    return 0;
}
```

(7,16)  
(13,4)  
(-13,-4)

# ++Prefix operator overloading

```
1  #include <iostream>
2  using namespace std;
3
4  class Point{
5  private:
6      int xpos, ypos;
7  public:
8      Point(int x, int y):xpos(x),ypos(y){}
9      void show() const{
10         cout<<"("<<xpos<<","<<ypos<<")"<<endl;
11     }
12     Point& operator++(){
13         xpos+=1;
14         ypos+=1;
15         return *this;
16     }
17     // Point operator++(){
18     //     xpos+=1;
19     //     ypos+=1;
20     //     return *this;
21     // }
22 };
23 int main(){
24     Point pos1(3,4);
25     ++pos1;
26     pos1.show();
27     ++(++pos1);
28     pos1.show();
29     return 0;
30 }
```

++(++pos1);

++(pos1.operator++());

++(reference of pos1);

(reference of pos1).operator++();

# Postfix++ operator overloading

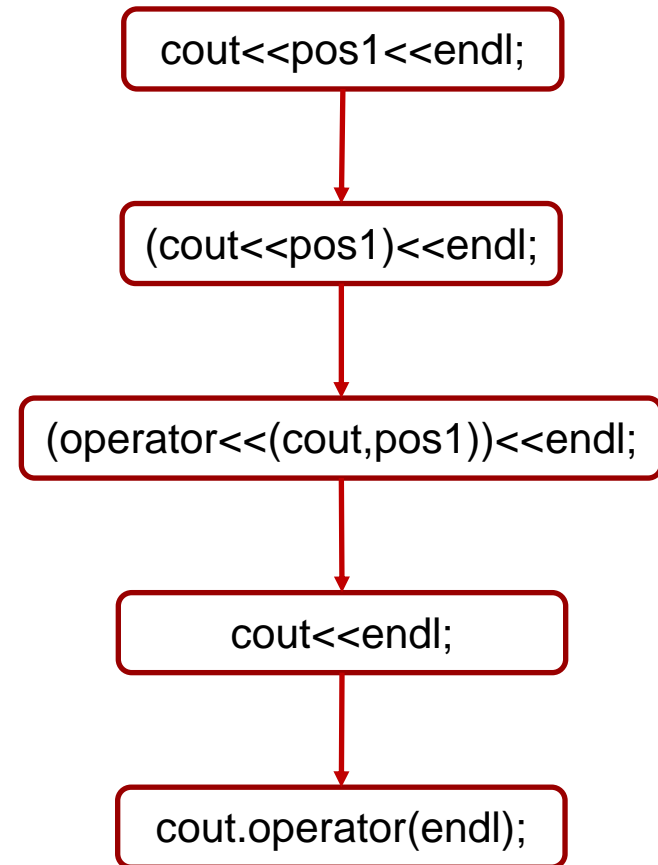
```
1  #include <iostream>
2  using namespace std;
3
4  class Point{
5  private:
6      int xpos, ypos;
7  public:
8      Point(int x, int y):xpos(x),ypos(y){}
9      void show() const{
10         cout<<"("<<xpos<<","<<ypos<<")"<<endl;
11     }
12     const Point operator++(int){
13         Point ref(xpos,ypos);
14         xpos+=1;
15         ypos+=1;
16         return ref;
17     }
18 };
19 int main(){
20     Point pos1(3,4);
21     Point cpy(0,0);
22     cpy = pos1++;
23     cpy.show();    (3,4)
24     pos1.show();   (4,5)
25
26     // cpy=(pos1++)++;
27     // pos1.show();
28     // cpy.show();
29     return 0;
30 }
```



# << operator overloading

```
1  #include <iostream>
2  using namespace std;
3
4  class Point{
5  private:
6      int xpos, ypos;
7  public:
8      Point(int x, int y):xpos(x),ypos(y){}
9      const Point operator+(const Point &ref){
10         Point pos(xpos+ref.xpos,ypos+ref.ypos);
11         return pos;
12     }
13     friend ostream& operator<<(ostream& os, const Point& pos);
14 };
15 ostream& operator<<(ostream& os, const Point& pos)
16 {
17     os << "(" << pos.xpos << "," << pos.ypos << ")";
18     return os;
19 }
20 int main(){
21     Point pos1(3,4);
22     Point pos2(10,20);
23     Point pos3=pos1+pos2;
24     Point pos4=pos1.operator+(pos2);
25
26     cout << pos1 << endl;
27     cout << pos2 << endl;
28     cout << pos3 << pos4 << endl;;
29     return 0;
30 }
```

(3,4)  
(10,20)  
(13,24)(13,24)



# >> operator overloading

```
1  #include <iostream>
2  using namespace std;
3  class Point{
4  private:
5      int xpos, ypos;
6  public:
7      Point(int x, int y):xpos(x),ypos(y){}
8      const Point operator+(const Point &ref){
9          Point pos(xpos+ref.xpos,ypos+ref.ypos);
10         return pos;
11     }
12     friend istream& operator>>(istream& is, Point& pos);
13     friend ostream& operator<<(ostream& os, const Point& pos);
14 };
15 istream& operator>>(istream& is, Point& pos)
16 {
17     is >> pos.xpos >> pos.ypos;
18     return is;
19 }
20 ostream& operator<<(ostream& os, const Point& pos)
21 {
22     os << "(" << pos.xpos << "," << pos.ypos << ")";
23     return os;
24 }
25 int main(){
26     Point pos1(0,0);
27     Point pos2(0,0);
28     cin >> pos1 >> pos2;
29     Point pos3=pos1+pos2;
30     Point pos4=pos1.operator+(pos2);
31
32     cout << pos1 << endl;
33     cout << pos2 << endl;
34     cout << pos3 << pos4 << endl;;
35     return 0;
36 }
```

```
3
4
10
20
(3,4)
(10,20)
(13,24)(13,24)
```

# Exercise 2

## ■ Define Time class and global functions

- Time class has 3 int variables(hour, minute, second)
- hour(0~23), minute(0~59), second(0~59)
- If exceed maximum value of variable, increase hour or minute and store the rest.  
(e.g. 12h 130m 55s = 14h 10m 55s, 3h 58m 150s = 4h 0m 30s,  
15h 35m 20s + 10h 30m 30s = 2h 5m 40s)
- You don't need to consider negative value
- There are four operator overloading functions(+,>>,<<,++)
- postfix++ operator overloading adds one to all variables

```
int main(){
    Time t1;
    Time t2;
    cin >> t1 >> t2;
    Time t3 = t1+t2;

    cout << t1 << endl;
    cout << t2 << endl;
    cout << t3++ << endl;
    cout << t3 << endl;
    return 0;
}
```

```
10
20
30
10
20
30
10h 20m 30s
10h 20m 30s
20h 41m 0s
21h 42m 1s
```

```
12
130
55
3
58
150
14h 10m 55s
4h 0m 30s
18h 11m 25s
19h 12m 26s
```

```
15
35
20
10
30
30
15h 35m 20s
10h 30m 30s
2h 5m 50s
3h 6m 51s
```