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

```
#include <iostream>
using namespace std;

class Point{
private:
    int xpos, ypos;
public:
    Point(int x, int y):xpos(x),ypos(y){}
    void show() const{
        cout<<"("<<xpos<<","<<ypos<<")"<<endl;
    }
    const Point operator+(const Point &ref){
        Point pos(xpos+ref.xpos,ypos+ref.ypos);
        return pos;
    }
};
int main(){
    Point pos1(3,4);
    Point pos2(10,20);
    Point pos3=pos1.operator+(pos2);
    Point pos4=pos1+pos2;

    pos1.show();
    pos2.show();
    pos3.show();
    pos4.show();
    return 0;
}
```

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

Global function Operator overloading 1

```cpp
#include <iostream>
using namespace std;
class Point{
private:
    int xpos, ypos;
public:
    Point(int x, int y):xpos(x),ypos(y){}
    void show() const{
        cout<<"("<<xpos<<","<<ypos<<")"<<endl;
    }
    int getX() const{
        return xpos;
    }
    int getY() const{
        return ypos;
    }
};
const Point operator+(const Point &ref1, const Point &ref2){
    const Point pos(ref1.getX()+ref2.getX(),ref1.getY()+ref2.getY());
    return pos;
}
int main(){
    Point pos1(3,4);
    Point pos2(10,20);
    Point pos3=pos1+pos2;
    Point pos4=operator+(pos1, pos2);

    pos1.show();
    pos2.show();
    pos3.show();
    pos4.show();
    return 0;
}
```

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

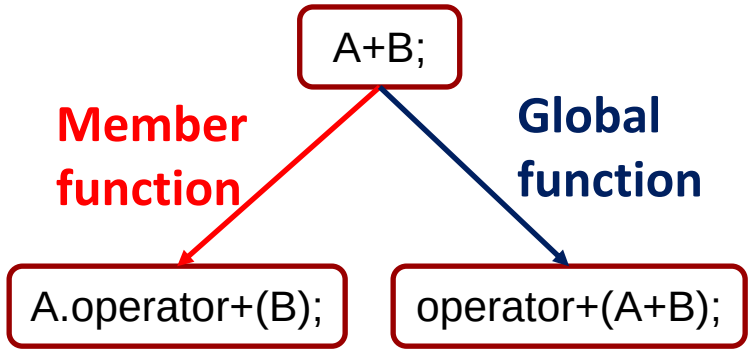
Global function Operator overloading 2

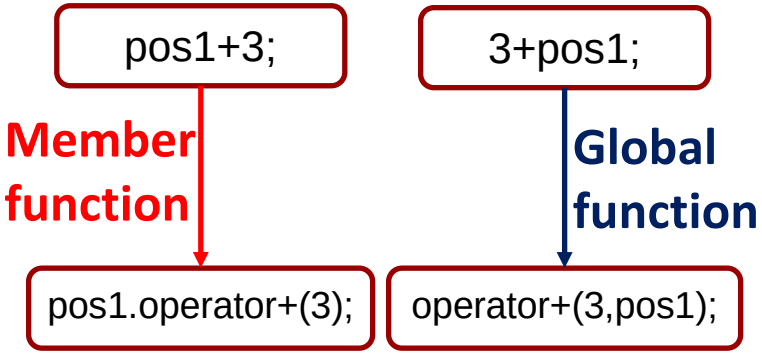
```cpp
#include <iostream>
using namespace std;
class Point{
private:
    int xpos, ypos;
public:
    Point(int x, int y):xpos(x),ypos(y){}
    void show() const{
        cout<<"("<<xpos<<","<<ypos<<")"<<endl;
    }
    friend const Point operator+(const Point &ref1, const Point &ref2);
};
const Point operator+(const Point &ref1, const Point &ref2){
    Point pos(ref1.xpos+ref2.xpos,ref1.ypos+ref2.ypos);
    return pos;
}
int main(){
    Point pos1(3,4);
    Point pos2(10,20);
    Point pos3=pos1+pos2;
    Point pos4=operator+(pos1, pos2);

    pos1.show();
    pos2.show();
    pos3.show();
    pos4.show();
    return 0;
}
```

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

Member and Global function example

```cpp
#include <iostream>
using namespace std;
class Point{
private:
    int xpos, ypos;
public:
    Point(int x, int y):xpos(x),ypos(y){}
    void show() const{
        cout<<"("<<xpos<<","<<ypos<<")"<<endl;
    }
    const Point operator+(int num){
        Point pos(xpos+num,ypos+num);
        return pos;
    }
    friend const Point operator+(int num, const Point &ref);
};
const Point operator+(int num, const Point &ref){
    const Point pos(num+ref.xpos,num+ref.ypos);
    return pos;
}
int main(){
    Point pos1(3,4);
    int num = 3;
    Point pos2=pos1+3;
    Point pos3=3+pos1;

    pos2.show();   (6,7)
    pos3.show();   (6,7)
    return 0;
}
```





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;
}
```

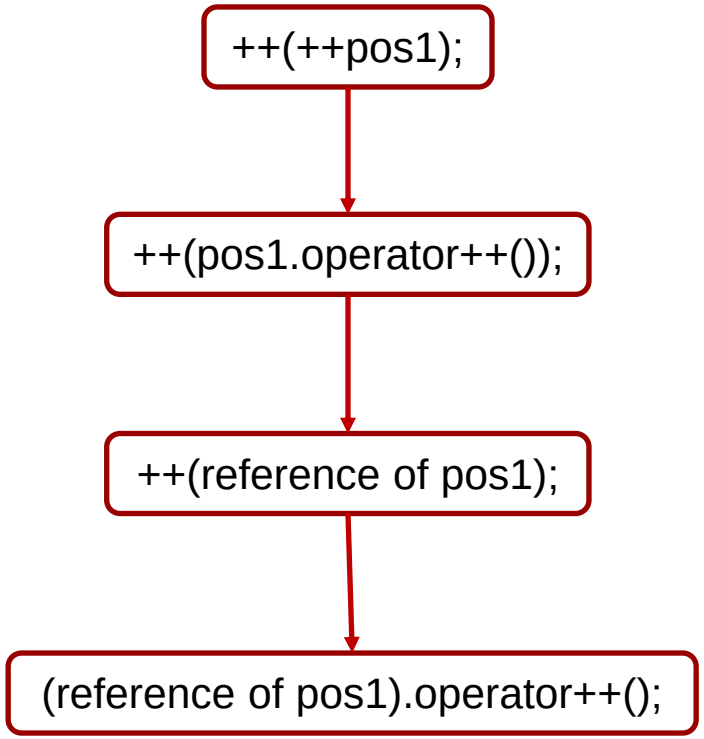
```
(7,16)
(13,4)
(-13,-4)
```

++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  }
```

(4,5)
(6,7)

(4,5)
(5,6)



Postfix++ operator overloading

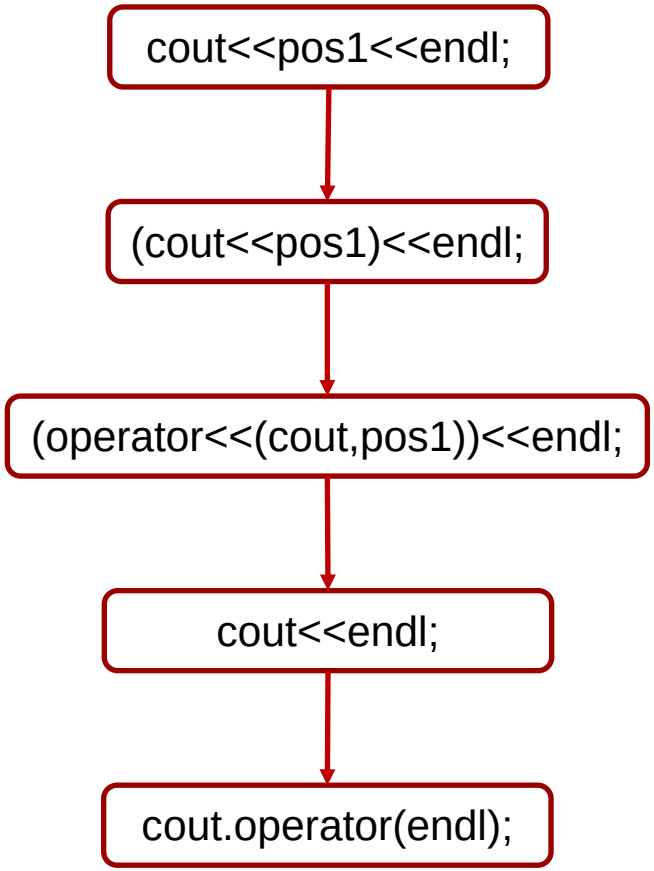
```
1   #include <iostream>
2   using namespace std;
3
4   class Point{
5   private:
6       int xpos, ypos;
7   public:
8       Point(int x, int y):xpos(x),ypos(y){}
9       void show() const{
10          cout<<"("<<xpos<<","<<ypos<<")"<<endl;
11      }
12      const Point operator++(int){
13      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

```
(3,4)
(4,5)
```

<< operator overloading

```cpp
#include <iostream>
using namespace std;

class Point{
private:
    int xpos, ypos;
public:
    Point(int x, int y):xpos(x),ypos(y){}
    const Point operator+(const Point &ref){
        Point pos(xpos+ref.xpos,ypos+ref.ypos);
        return pos;
    }
    friend ostream& operator<<(ostream& os, const Point& pos);
};
ostream& operator<<(ostream& os, const Point& pos)
{
    os << "(" << pos.xpos << "," << pos.ypos << ")";
    return os;
}
int main(){
    Point pos1(3,4);
    Point pos2(10,20);
    Point pos3=pos1+pos2;
    Point pos4=pos1.operator+(pos2);

    cout << pos1 << endl;
    cout << pos2 << endl;
    cout << pos3 << pos4 << endl;;
    return 0;
}
```

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



>> operator overloading

```
#include <iostream>
using namespace std;
class Point{
private:
    int xpos, ypos;
public:
    Point(int x, int y):xpos(x),ypos(y){}
    const Point operator+(const Point &ref){
        Point pos(xpos+ref.xpos,ypos+ref.ypos);
        return pos;
    }
    friend istream& operator>>(istream& is, Point& pos);
    friend ostream& operator<<(ostream& os, const Point& pos);
};
istream& operator>>(istream& is, Point& pos)
{
    is >> pos.xpos >> pos.ypos;
    return is;
}
ostream& operator<<(ostream& os, const Point& pos)
{
    os << "(" << pos.xpos << "," << pos.ypos << ")";
    return os;
}
int main(){
    Point pos1(0,0);
    Point pos2(0,0);
    cin >> pos1 >> pos2;
    Point pos3=pos1+pos2;
    Point pos4=pos1.operator+(pos2);

    cout << pos1 << endl;
    cout << pos2 << endl;
    cout << pos3 << pos4 << endl;;
    return 0;
}
```

```
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
```