Data Structures and Object Oriented Programming

Lecture 14

Dr. Naveed Anwar Bhatti

Webpage: naveedanwarbhatti.github.io

Operator Overloading

Some More Binary Operators

Subscript [] Operator

- Subscript operator **[]** is normally used to access elements of an array
- It's a **binary operator**
- The operator function takes the implied **object** and a **subscript index** as parameters
- Usually **returns a reference** to the indexed value to allow assignment, e.g. object[2] = 10;

Overloading Subscript [] Operator

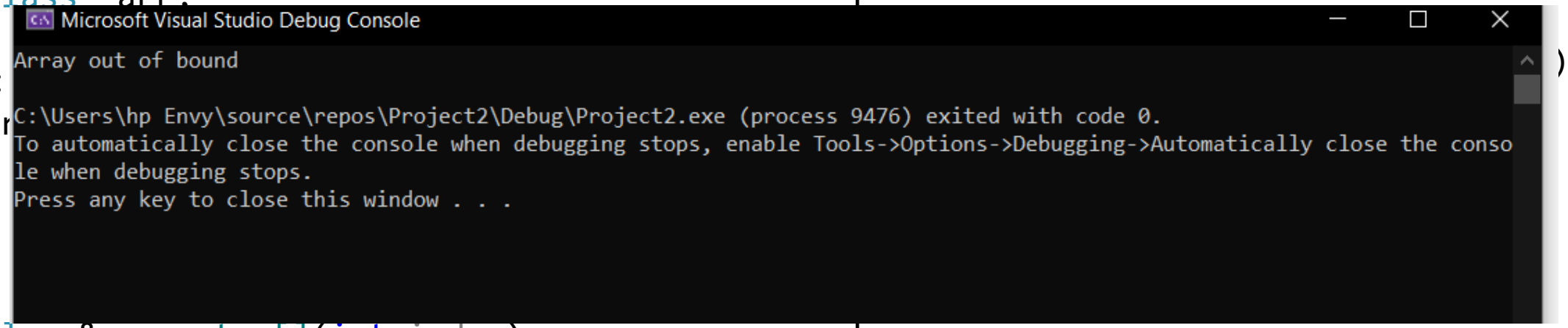
```
class myarray {
    myclass* arr;
    int size;
public:
    myarray(int size)
    {
        this->size = size;
        arr = new myclass[size];
    }









};
```

```
class myclass {
    int x, y;
public:
    myclass(int a = 0, int b = 0)
    {
        x = a;
        y = b;
    }
};


int main() {
    myarray foo(10);
    myclass bar;
    bar=foo[10];
    return 0;

}
```

Overloading Subscript [] Operator

```
class myarray {
    myclass* arr;
    int
public:
    myar
    {


    }
    myclass& operator[](int index)
    {
        if (index >= size)
        {
            cout << "Array out of bound" << endl;
            exit(0);
        }
        return arr[index];
    }
};
```

```
class myclass {
    int x, y;
```




```
int main() {
    myarray foo(10);
    myclass bar;
    bar=foo[10];
    return 0;

}
```

Very similar….Function Call () Operator

- Function call operator **()** is normally used in function calling to pass parameters
- It’s not a **binary operator**
- The operator function takes **one object** and **unlimited inputs** as parameters
- Normally declare as member function

Unary Operator

Class: Operator Overloading

- Unary operators:

  & * + - ++ -- ! ~

- Examples:
  - --x
  - -(x++)
  - !(*ptr ++)

- Unary operators are usually prefix, except for **++** and **--**
- **++** and **--** both act as prefix and postfix

Class: Operator Overloading (++ and - -)

- The operator ++ and –– have two forms : pre and post

```
int x = 6;
++x; // preincrement
x++; // postincrement
--x; // predecrement
x--; // postdecrement
```

- To overload the preincrement and predecrement operator, we use the declaration:

operator++();
operator--();

What about Postfix? Later

Class: Operator Overloading (++ and - -) – (Member Function)

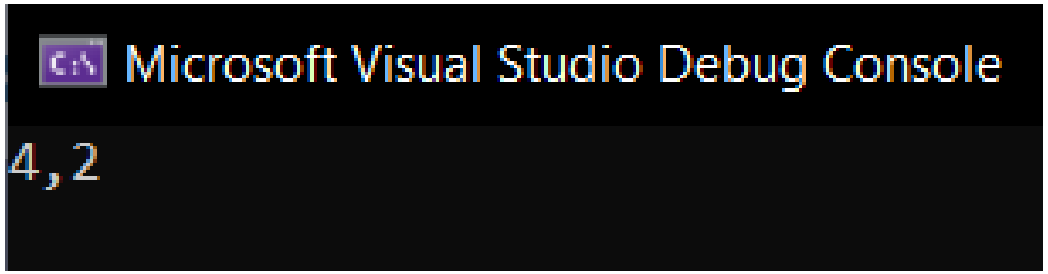
- Example

```
class myclass {
    int x, y;
public:
    myclass() {};
    myclass(int, int);
    void operator++ ();
    void print();
};

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

void myclass::print() {
    cout << x << ',' << y << '\n';
}
```

```
void myclass::operator++ () {
    ++x;
    ++y;
}
```

```
int main() {
    myclass foo(3, 1);
    ++foo;
    foo.print();
    return 0;
}
```

Prefix version



Class: Operator Overloading (++ and - -) – (Member Function)

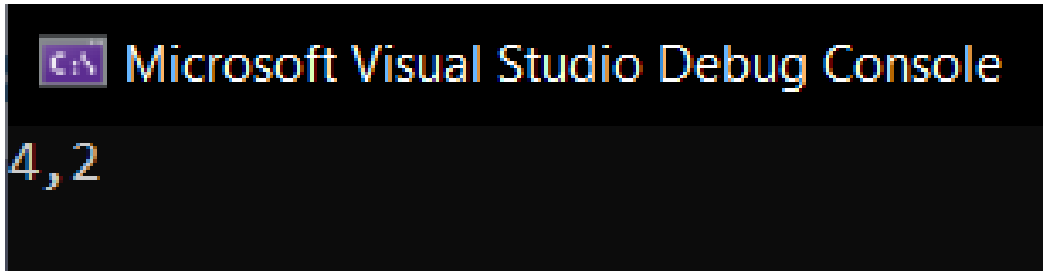
- Example

```
class myclass {
    int x, y;
public:
    myclass() {};
    myclass(int, int);
    void operator++ (int);
    void print();
};

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

void myclass::print() {
    cout << x << ',' << y << '\n';
}
```

Compiler know its Postfix

```
void myclass::operator++ (int) {
    x++;
    y++;
}
```

```
int main() {
    myclass foo(3, 1);
    foo++;
    foo.print();
    return 0;
}
```



Class: Operator Overloading (++ and - -)

- To overload the *preincrement* and *predecrement* operator, we use the declaration:

  operator++();
  operator--();

- To overload the *postincrement* and *postdecrement* operator, we use the declaration:

  operator++(int);
  operator--(int);

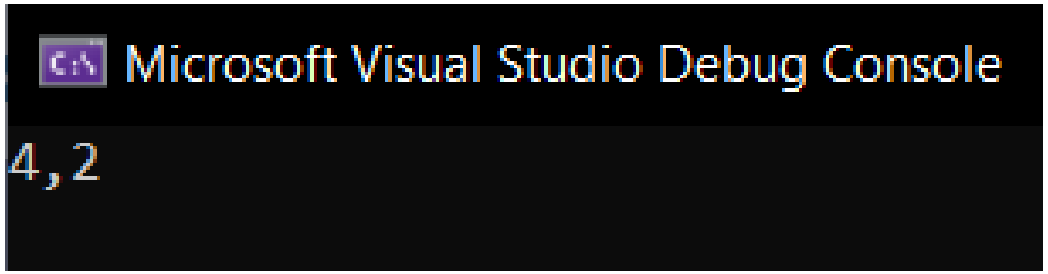
Class: Operator Overloading (++ and - -) – (Friend Function)

- Example

```
class myclass {
    int x, y;
public:
    myclass() {};
    myclass(int, int);
    friend void operator++ (myclass&);
    void print();
};

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

void myclass::print() {
    cout << x << ',' << y << '\n';
}
```

```
void operator++ (myclass& a) {
    ++a.x;
    ++a.y;
}
```

```
int main() {
    myclass foo(3, 1);
    ++foo;
    foo.print();
    return 0;
}
```



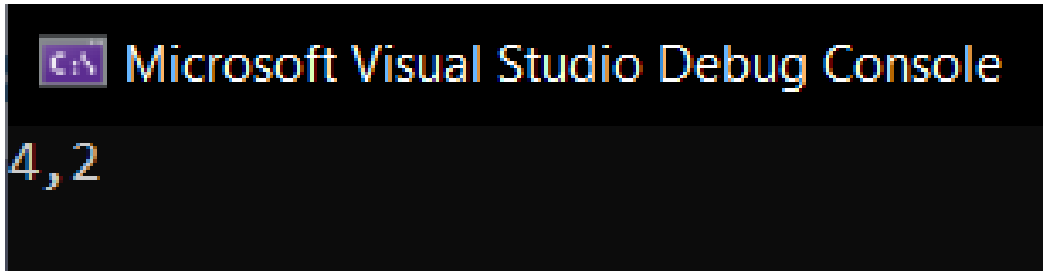
Class: Operator Overloading (++ and - -) – (Friend Function)

- What will happen if I will write main() like this?

```
int main() {
    myclass foo(3, 1);
    myclass bar(0,0);
    bar = ++foo;
    bar.print();
    return 0;
}
```

Error

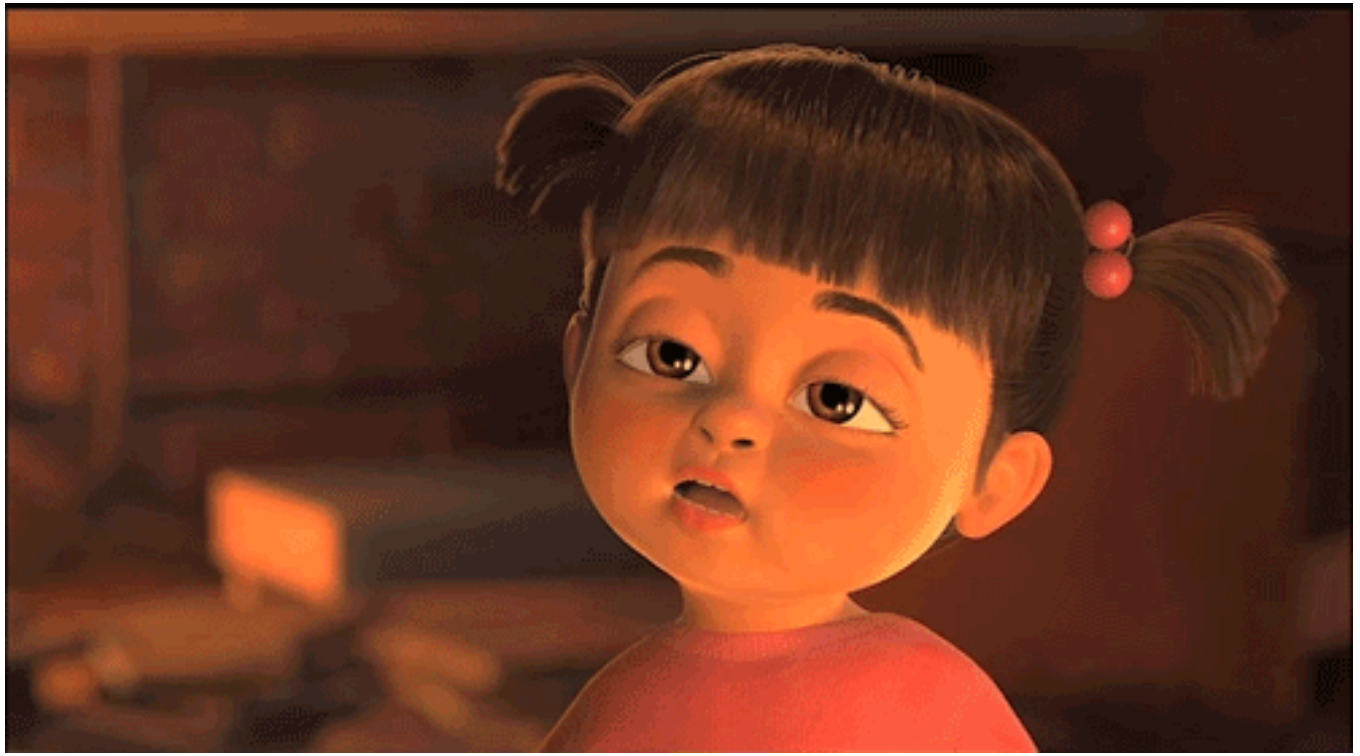
```
myclass operator++ (myclass& a) {
    myclass temp;
    temp.x=++a.x;
    temp.y=++a.y;
    return temp;
}
```



Class: Operator Overloading (++ and - -)

Exercise: Overload ‘ -- ’ operator for same class using both methods, i.e., Friend Function and Member Function.

Thanks a lot



If you are taking a Nap, **wake up**.....Lecture Over