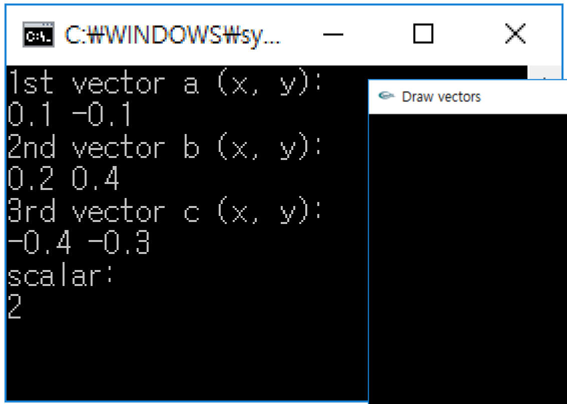
LAB I
Week 11

Seoul National University
Graphics & Media Lab
HyeonSeung Shin

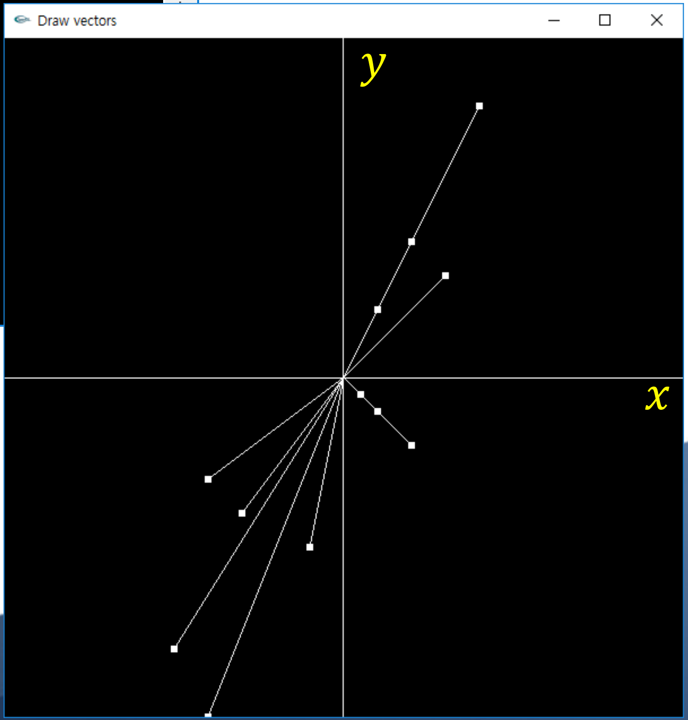
Today's Mission

- Input three 2D vectors **a**, **b,** and **c** in the range [-1, 1] x [-1, 1], and a scalar p.
- Draw the three input vectors in the OpenGL window.
- As the user press the enter key, draw
  - **a** + **b**
  - **a** – **b**
  - **a** * p
  - **a** / p
  - **a** += **c**
  - **a** -= **b**
  - **b** *= p
  - **b** /= p * p
  - (**a**[0] + **b**[0], **a**[1] - **b**[1])

vector



```
1st vector a (x, y):
0.1 -0.1
2nd vector b (x, y):
0.2 0.4
3rd vector c (x, y):
-0.4 -0.3
scalar:
2
```

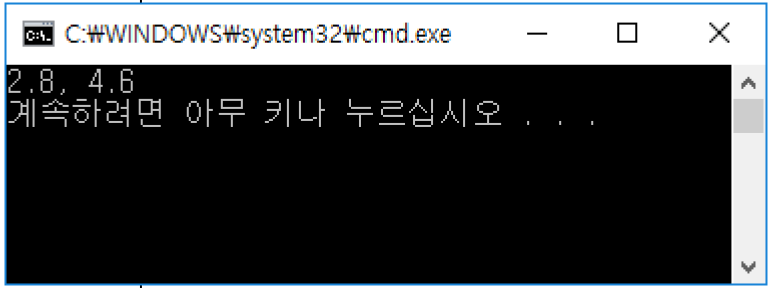


Non-member Operator Overloading

```
class Vec2 {
public :
   Vec2() : x(0), y(0) {}
   Vec2(float a, float b) : x(a), y(b) {}

   float x,y;
};

inline Vec2 operator+(const Vec2& a, const Vec2& b)
{ return Vec2(a.x+b.x, a.y+b.y); }

void main() {
   Vec2 a(1.1f,0), b(1.3f,2.5f);
   Vec2 x = a + b;
}
```

Non-member Operator Overloading

```
class Vec2 {
public:
	Vec2() : x(0), y(0) {}
	Vec2(float a, float b) : x(a), y(b) {}

	float x, y;
};

Vec2 operator*(Vec2& v, float s) {
	return Vec2(v.x * s, v.y * s);
}

void main() {
	Vec2 v1(1.4, 2.3);
	float s = 2;

	Vec2 v_ans = v1 * s;
	cout << v_ans.x << ", " << v_ans.y << endl;
}
```


C:₩WINDOWS₩system32₩cmd.exe

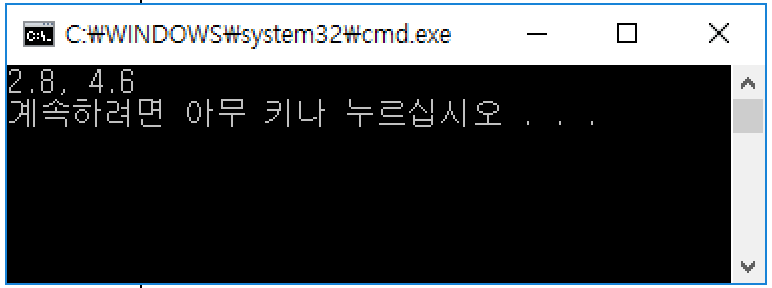
```
2.8, 4.6
계속하려면 아무 키나 누르십시오 . . .
```

Member Operator Overloading

```
class Vec2 {
public :
   Vec2() : x(0), y(0) {}
   Vec2(float a, float b) : x(a), y(b) {}

   Vec2 operator+(const Vec2& a)
   { return Vec2(x+a.x, y+a.y); }

   float x,y;
};

void main() {
   Vec2 a(1.1f,0), b(1.3f,2.5f);
   Vec2 x = a + b;
}
```

Member Operator Overloading

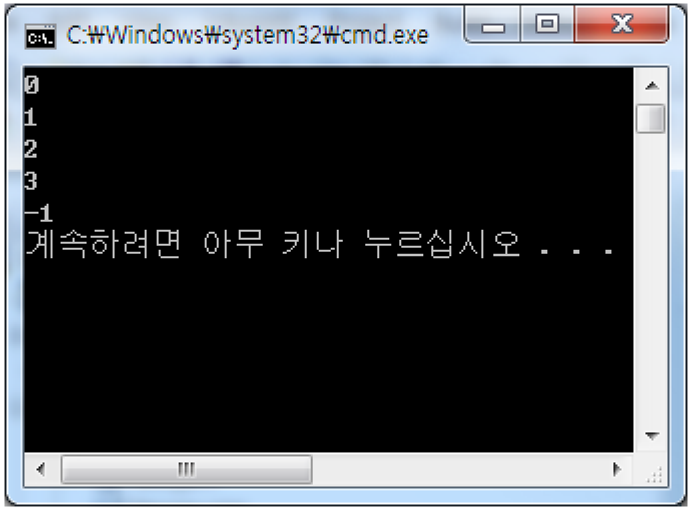
```
class Vec2 {
public:
	Vec2() : x(0), y(0) {}
	Vec2(float a, float b) : x(a), y(b) {}

	Vec2& operator*=(float s) {
		x *= s;    y *= s;    return (*this);
	}

	float x, y;
};

Vec2 operator*(Vec2& v, float s) {
	return Vec2(v.x * s, v.y * s);
}

void main() {
	Vec2 v1(1.4, 2.3);
	float s = 2;

	Vec2 v_ans = v1 * s;
	cout << v_ans.x << ", " << v_ans.y << endl;
}
```



```
2.8, 4.6
계속하려면 아무 키나 누르십시오 . . .
```

Subscript Operator Overloading

```
class Array {
public :
   Array(std::size_t num) : _size(num) { ptr = new int[num]; }
   Array(const Array& arr) : _size(arr._size) {
      ptr = new int[_size];
      for(std::size_t i=0;i<_size;++i)
         ptr[i] = arr.ptr[i];
   }
   ~Array() { if(ptr != NULL) delete [] ptr; }
   Array& operator=(const Array& arr) {
      if(ptr != NULL) delete [] ptr;
      _size = arr._size;
      ptr = new int[arr._size];
      for(std::size_t i=0;i<_size;++i)
         ptr[i] = arr.ptr[i];
      return (*this);
   }
   const std::size_t size() const { return _size; }

   int& operator[](const std::size_t i) { return ptr[i]; }
   // int operator[](const std::size_t i) const { return ptr[i]; }

public :
   int *          ptr;
   std::size_t    _size;
};

void main() {
   Array a(5);
   a[0] = 0; a[1] = 1; a[2] = 2; a[3] = 3; a[4] = -1;
   for(std::size_t i=0;i<a.size();++i)
      std::cout << a[i] << std::endl;
}
```

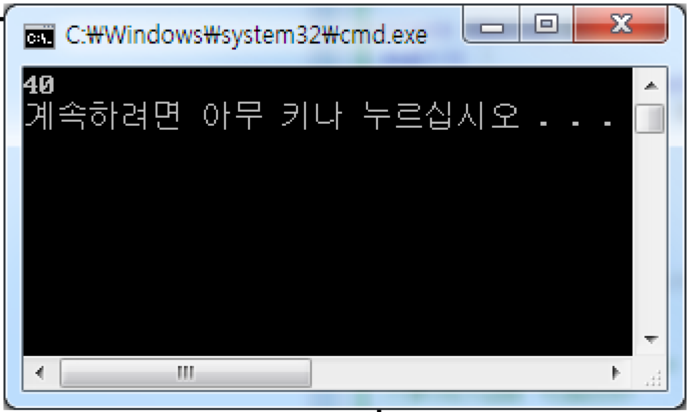


```
0
1
2
3
-1
계속하려면 아무 키나 누르십시오 . . .
```

Call Operator

- A **call operator** can be overloaded for the class.

```
class AbsInt {
public :
   int operator()(int val) {
      return val < 0 ? - val : val;
   }
};

void main() {
   AbsInt absint;
   std::cout << absint(-40) << std::endl;
}
```


C:\Windows\system32\cmd.exe

```
40
계속하려면 아무 키나 누르십시오 . . .
```

Graphics & Media Lab Seoul National Univ

Function Object (Functor)

- Even though `AbsInt` is a class and not a function, we can make a "call" on an object of `AbsInt`.

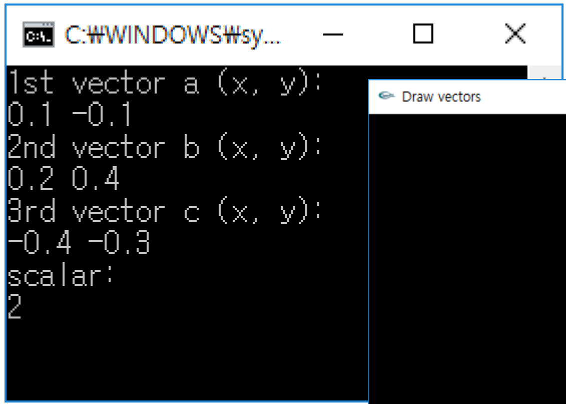
- Class objects which can be used with the call operator are referred to as **function objects** or **functors**.
  - They are objects that act like functions

```
class AbsInt {
public :
   int operator()(int val) {
      return val < 0 ? - val : val;
   }
};

void main() {
   AbsInt absint;
   std::cout << absint(-40) << std::endl;
}
```

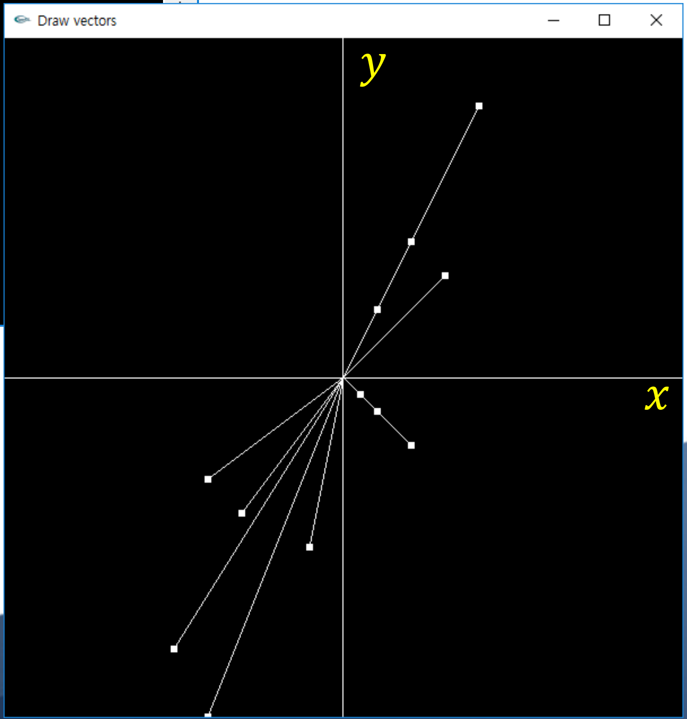
Today's Mission

- Input three 2D vectors **a**, **b,** and **c** in the range [-1, 1] x [-1, 1], and a scalar p.
- Draw the three input vectors in the OpenGL window.
- As the user press the enter key, draw
  - **a** + **b**
  - **a** – **b**
  - **a** * p
  - **a** / p
  - **a** += **c**
  - **a** -= **b**
  - **b** *= p
  - **b** /= p * p
  - (**a**[0] + **b**[0], **a**[1] - **b**[1])

vector



```
1st vector a (x, y):
0.1 -0.1
2nd vector b (x, y):
0.2 0.4
3rd vector c (x, y):
-0.4 -0.3
scalar:
2
```



Class Diagram

```
class Vec2 {
public:
        Vec2();
        Vec2(float x, float y);
        Vec2(const Vec2& v);

        void setPos(float x, float y);
        void draw() const;

        float& operator[](const unsigned int i);

        Vec2& operator+=(Vec2& v);
        Vec2& operator-=(Vec2& v);
        Vec2& operator*=(float s);
        Vec2& operator/=(float s);

private:
        float pos[2];
};

Vec2 operator+(Vec2& v1, Vec2& v2);
Vec2 operator-(Vec2& v1, Vec2& v2);
Vec2 operator*(Vec2& v, float s);
Vec2 operator/(Vec2& v, float s);
```

member

Non-member