고급프로그래밍및실습 과제 #5 (10주차)

214823 박종현

1. `++`, `--`
2. `+`, `-`, `*`, `/`
3. `=`, `()`, `[]`
4. `==`, `!=`
5. `<<`, `>>`
6. `*`, `->`

- 각자 최소 1개 씩 오버로딩

답안

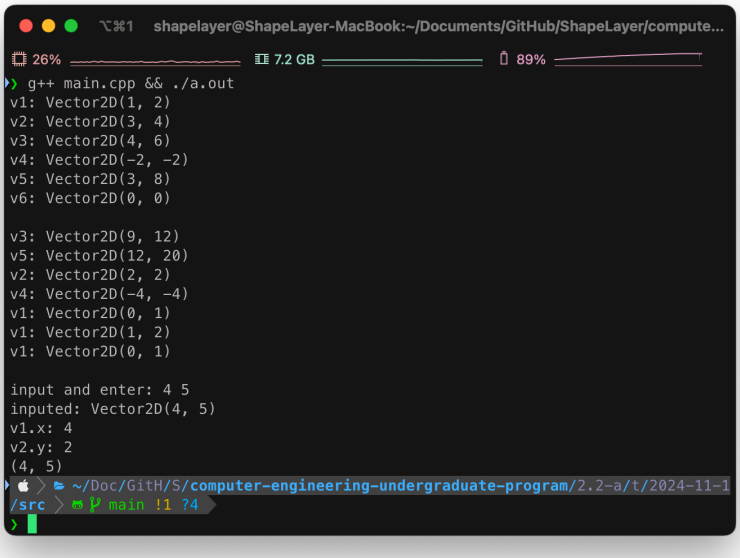
```C++
#include "bits/stdc++.h"
#define ERR_MESSAGE_INDEX_OUT_OF_RANGE "Index out of range"

using namespace std;

class Vector2 {
  friend ostream& operator<<(ostream &os, const Vector2 &v) {
    os << "Vector2D" << v.to_string();
    return os;
  }
  friend istream& operator>>(istream &is, Vector2 &v) {
    is >> v._x >> v._y;
    return is;
  }
private:
  double _x, _y;
public:
  Vector2(double x = 0., double y = 0.) : _x(x), _y(y) {}

  inline int x() const { return _x; }
  inline int y() const { return _y; }
  inline void x(int x) { _x = x; }
  inline void y(int y) { _y = y; }

  string to_string() const {
    return "(" + std::to_string(x()) + ", " + std::to_string(y()) + ")";
  }

  Vector2 operator+(const Vector2 &other) const {
    Vector2 v;
    v.x(this->x() + other.x());
    v.y(this->y() + other.y());
    return v;
  }

  Vector2 operator-(const Vector2 &other) const {
    Vector2 v;
    v.x(this->x() - other.x());
    v.y(this->y() - other.y());
    return v;
  }

  Vector2 operator*(double scalar) const {
```

```
44      Vector2 v;
45      v.x(this->x() * scalar);
46      v.y(this->y() * scalar);
47      return v;
48    }
49
50    Vector2 operator*(const Vector2 &other) const {
51      Vector2 v;
52      v.x(this->x() * other.x());
53      v.y(this->y() * other.y());
54      return v;
55    }
56
57    Vector2 operator/(const Vector2 &other) const {
58      Vector2 v;
59      v.x(this->x() / other.x());
60      v.y(this->y() / other.y());
61      return v;
62    }
63
64    Vector2 operator==(const Vector2 &other) const {
65      return this->x() == other.x() && this->y() == other.y();
66    }
67
68    Vector2 operator!=(const Vector2 &other) const {
69      return this->x() != other.x() || this->y() != other.y();
70    }
71
72    Vector2 operator+=(const Vector2 &other) {
73      this->x(this->x() + other.x());
74      this->y(this->y() + other.y());
75      return *this;
76    }
77
78    Vector2 operator-=(const Vector2 &other) {
79      this->x(this->x() - other.x());
80      this->y(this->y() - other.y());
81      return *this;
82    }
83
84    Vector2 operator*=(const Vector2 &other) {
85      this->x(this->x() * other.x());
86      this->y(this->y() * other.y());
87      return *this;
88    }
89
90    Vector2 operator/=(const Vector2 &other) {
91      this->x(this->x() / other.x());
92      this->y(this->y() / other.y());
93      return *this;
94    }
95
96    Vector2 operator++() {
97      this->x(this->x() + 1);
98      this->y(this->y() + 1);
```

```
99        return *this;
100     }
101
102     Vector2 operator++(int) {
103       this->x(this->x() + 1);
104       this->y(this->y() + 1);
105       return *this;
106     }
107
108
109     Vector2 operator--() {
110       this->x(this->x() - 1);
111       this->y(this->y() - 1);
112       return *this;
113     }
114
115     Vector2 operator--(int) {
116       this->x(this->x() - 1);
117       this->y(this->y() - 1);
118       return *this;
119     }
120
121     Vector2& operator=(const Vector2 &other) {
122       if (this != &other) {
123         this->x(other.x());
124         this->y(other.y());
125       }
126       return *this;
127     }
128
129     double& operator[](int index) {
130       if (index == 0) {
131         return _x;
132       } else if (index == 1) {
133         return _y;
134       } else {
135         throw std::out_of_range(ERR_MESSAGE_INDEX_OUT_OF_RANGE);
136       }
137     }
138
139     Vector2* operator->() {
140       return this;
141     }
142
143     const Vector2& operator*() const {
144       return *this;
145     }
146 };
147
148
149 // Usage Examples
150 int main() {
151     Vector2 v1(1., 2.), v2(3., 4.);
152     Vector2 v3 = v1 + v2;
153     Vector2 v4 = v1 - v2;
```

```
154    Vector2 v5 = v1 * v2;
155    Vector2 v6 = v1 / v2;
156
157    cout << "v1: " << v1 << endl;
158    cout << "v2: " << v2 << endl;
159    cout << "v3: " << v3 << endl;
160    cout << "v4: " << v4 << endl;
161    cout << "v5: " << v5 << endl;
162    cout << "v6: " << v6 << endl;
163
164    cout << endl;
165
166    v3 = v2 * 3.;
167    cout << "v3: " << v3 << endl;
168    v5 += v3;
169    cout << "v5: " << v5 << endl;
170    v2 -= v1;
171    cout << "v2: " << v2 << endl;
172    v4 *= v2;
173    cout << "v4: " << v4 << endl;
174    v1 /= v2;
175    cout << "v1: " << v1 << endl;
176    ++v1++;
177    cout << "v1: " << v1 << endl;
178    --v1--;
179    cout << "v1: " << v1 << endl;
180
181    cout << endl;
182    cout << "input and enter: ";
183    cin >> v1;
184    cout << "inputed: " << v1 << endl;
185
186    cout << "v1.x: " << v1[0] << endl;
187    cout << "v2.y: " << v2[1] << endl;
188
189    cout << (&v1)->to_string() << endl;
190    return 0;
191 }
192
```

실행 결과


⌥⌘1 shapelayer@ShapeLayer-MacBook:~/Documents/GitHub/ShapeLayer/compute...
26%
7.2 GB
89%
g++ main.cpp && ./a.out
v1: Vector2D(1, 2)
v2: Vector2D(3, 4)
v3: Vector2D(4, 6)
v4: Vector2D(-2, -2)
v5: Vector2D(3, 8)
v6: Vector2D(0, 0)
v3: Vector2D(9, 12)
v5: Vector2D(12, 20)
v2: Vector2D(2, 2)
v4: Vector2D(-4, -4)
v1: Vector2D(0, 1)
v1: Vector2D(1, 2)
v1: Vector2D(0, 1)
input and enter: 4 5
inputed: Vector2D(4, 5)
v1.x: 4
v2.y: 2
(4, 5)
~/Doc/GitH/S/computer-engineering-undergraduate-program/2.2-a/t/2024-11-1
/src main !1 ?4