

# Shape.h

12주차 과제

```
1  #ifndef Shape_H
2  #define Shape_H
3
4  class Shape {
5  protected:
6      virtual void draw() = 0;
7  public:
8      void paint() { draw(); }
9  };
10
11 #endif
```

1, 2, 11번 라인의 `#ifndef`, `#define`, `#endif` 명령어를 통해 헤더파일이 중복으로 정의되는 것을 방지합니다.

4~9번 라인에 Shape 클래스를 생성하고 Shape 클래스 내에 가상함수 draw를 protected로 선언하고 paint 함수를 통해 draw 함수를 호출합니다.

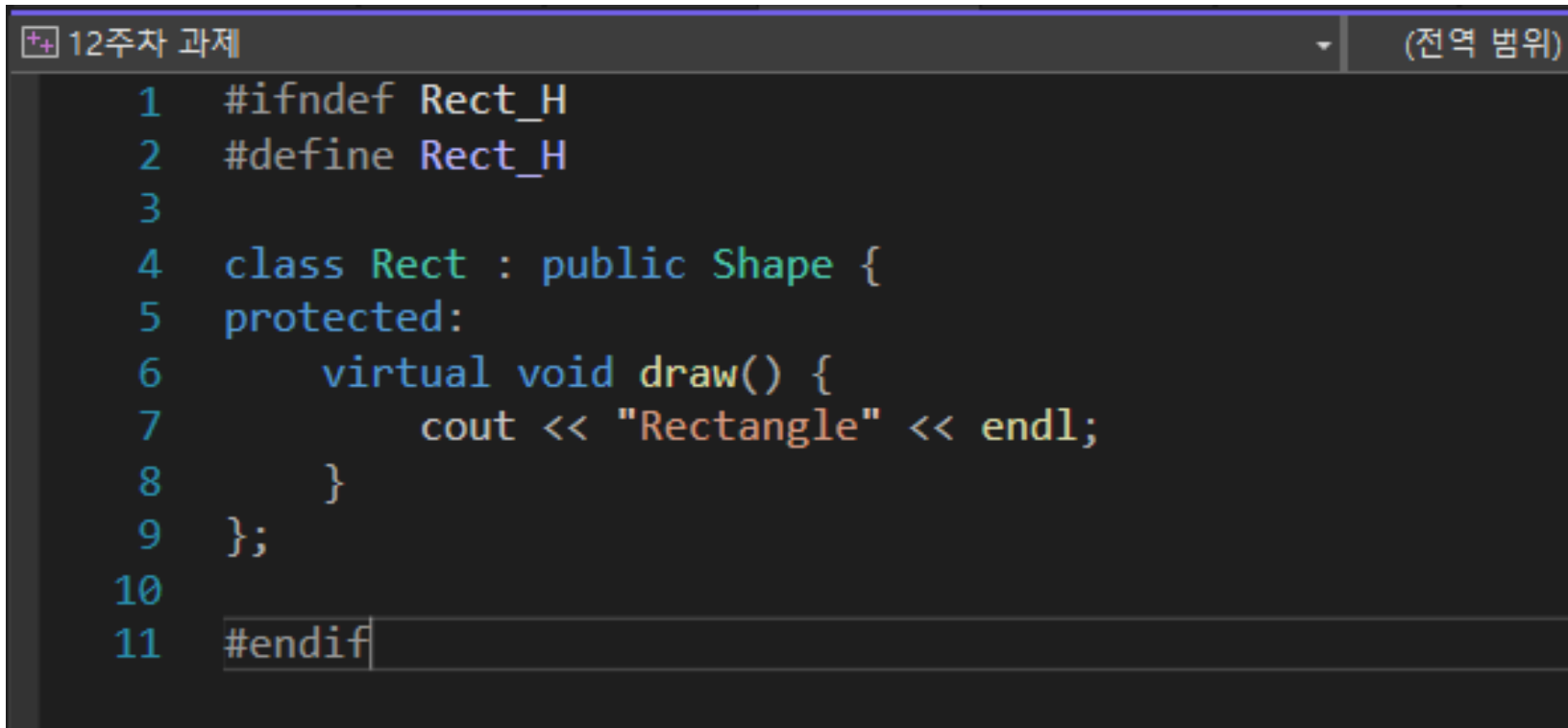
# Circle.h

```
12주차 과제 (전역)
1  #ifndef Circle_H
2  #define Circle_H
3
4  class Circle : public Shape {
5  protected:
6      virtual void draw() {
7          cout << "Circle" << endl;
8      }
9  };
10
11 #endif
```

1, 2, 11번 라인의 `#ifndef`, `#define`, `#endif` 명령어를 통해 헤더파일이 중복으로 정의되는것을 방지합니다.

4~9번 라인에 Shape 클래스를 상속받은 Circle 클래스를 생성해 draw를 수행하고 Circle을 출력한다.

# Rect.h



```
12주차 과제 (전역 범위)
1  #ifndef Rect_H
2  #define Rect_H
3
4  class Rect : public Shape {
5  protected:
6      virtual void draw() {
7          cout << "Rectangle" << endl;
8      }
9  };
10
11 #endif
```

1, 2, 11번 라인의 `#ifndef`, `#define`, `#endif` 명령어를 통해 헤더파일이 중복으로 정의되는것을 방지합니다.

4~9번 라인에 Shape 클래스를 상속받은 Rect 클래스를 생성해 draw를 수행하고 Rect을 출력한다.

# Line.h

```
12주차 과제 (전역 범위)
1  #ifndef Line_H
2  #define Line_H
3
4  class Line : public Shape {
5  protected:
6      virtual void draw() {
7          cout << "Line" << endl;
8      }
9  };
10
11 #endif
12
```

1, 2, 11번 라인의 `#ifndef`, `#define`, `#endif` 명령어를 통해 헤더파일이 중복으로 정의되는것을 방지합니다.

4~9번 라인에 Shape 클래스를 상속받은 Line 클래스를 생성해 draw를 수행하고 Line을 출력한다.

# UI.h

```
12주차 과제 (전역 범위)

1  #ifndef UI_H
2  #define UI_H
3
4  class UI {
5  public:
6      static int main_memu() {
7          int n;
8          cout << "삽입:1, 삭제:2, 모두보기:3, 종료:4 >> ";
9          cin >> n;
10         return n;
11     }
12     static int shape_memu() {
13         int n;
14         cout << "선:1, 원:2, 사각형:3 >> ";
15         cin >> n;
16         return n;
17     }
18     static int delete_menu() {
19         int n;
20         cout << "삭제하고자 하는 도형의 인덱스 >> ";
21         cin >> n;
22         return n;
23     }
24     static void showAll(vector<Shape*>& v, vector<Shape*>::iterator& it) {
25         int i = 0;
26         for (it = v.begin(); it != v.end(); it++, i++) {
27             cout << i << ": ";
28             v.at(i)->paint();
29         }
30     }
31 };
32
33 #endif
```

# UI.h

1, 2, 33번 라인의 `#ifndef`, `#define`, `#endif` 명령어를 통해 헤더파일이 중복으로 정의되는것을 방지합니다.

4~31번 라인에 UI클래스를 생성하고 각각의 `main_memu`, `shape_memu`, `delete_menu` 등을 생성하고 각 메뉴에서 `n값`을 입력받아 `n값`을 리턴합니다. 그리고 `showAll` 함수를 통하여 `vector`에 저장된 모든 도형을 출력하는 함수를 구현합니다.

# GraphicEditor.h

```
12주차 과제
GraphicEditor

1 #ifndef GraphicEditor_H
2 #define GraphicEditor_H
3
4 class GraphicEditor {
5     vector<Shape*> v;
6     vector<Shape*>::iterator it;
7 public:
8     GraphicEditor() {
9         cout << "그래픽 에디터입니다.\n";
10        start();
11    }
12    void start() {
13        while (true) {
14            int n;
15            n = UI::main_menu();
16            switch (n) {
17            case 1:
18                n = UI::shape_menu();
19                switch (n) {
20                case 1:
21                    v.push_back(new Line());
22                    break;
23                case 2:
24                    v.push_back(new Circle());
25                    break;
26                case 3:
27                    v.push_back(new Rect());
28                    break;
29                default:
30                    cout << "잘못 선택하셨습니다.\n";
31                    break;
32                }
33                break;
34            case 2: {
35                n = UI::delete_menu();
36                if (n >= v.size() || n < 0) {
37                    cout << "없는 인덱스 입니다.\n";
38                    break;
39                }

```

```

39            }
40            it = v.begin();
41            Shape* tmp = *(it + n);
42            v.erase(it + n);
43            delete tmp;
44            break;
45        }
46        case 3:
47            UI::showAll(v, it);
48            break;
49        case 4:
50            return;
51        default:
52            cout << "잘못 입력하셨습니다.\n";
53            break;
54        }
55    }
56 }
57
58 };
59
60 #endif
```

# GraphicEditor.h

1, 2, 60번 라인의 #ifndef, #define, #endif 명령어를 통해 헤더파일이 중복으로 정의되는것을 방지합니다.

4~60번 라인에 GraphicEditor클래스를 생성하고 `vector<Shape*> v;`를 통해 도형 객체들의 포인터를 저장하는 동적 배열 `v`를 생성하고 `vector`를 순회할 때 사용하는 반복자 `vector<Shape*>::iterator it;`을 선언합니다. 그리고 `switch`문을 이용하여 입력받은 `n` 값을 통해 각 UI클래스를 상속받아 각 메뉴에서 입력한 `n` 값에 따라서 해당하는 기능이 수행되도록 함수를 구현합니다.



# main.cpp

```
12주차 과제 (전역 범위)

1  #include <iostream>
2  #include <vector>
3  using namespace std;
4
5  #include "Shape.h"
6  #include "Circle.h"
7  #include "Rect.h"
8  #include "Line.h"
9  #include "UI.h"
10 #include "GraphicEditor.h"
11
12 int main() {
13     new GraphicEditor();
14 }
```

5~10번 라인에 각각의 헤더 파일들을 include 하고, 12~14번 라인의 메인 함수에서 GraphicEditor 객체를 생성하고 프로그램을 실행합니다.