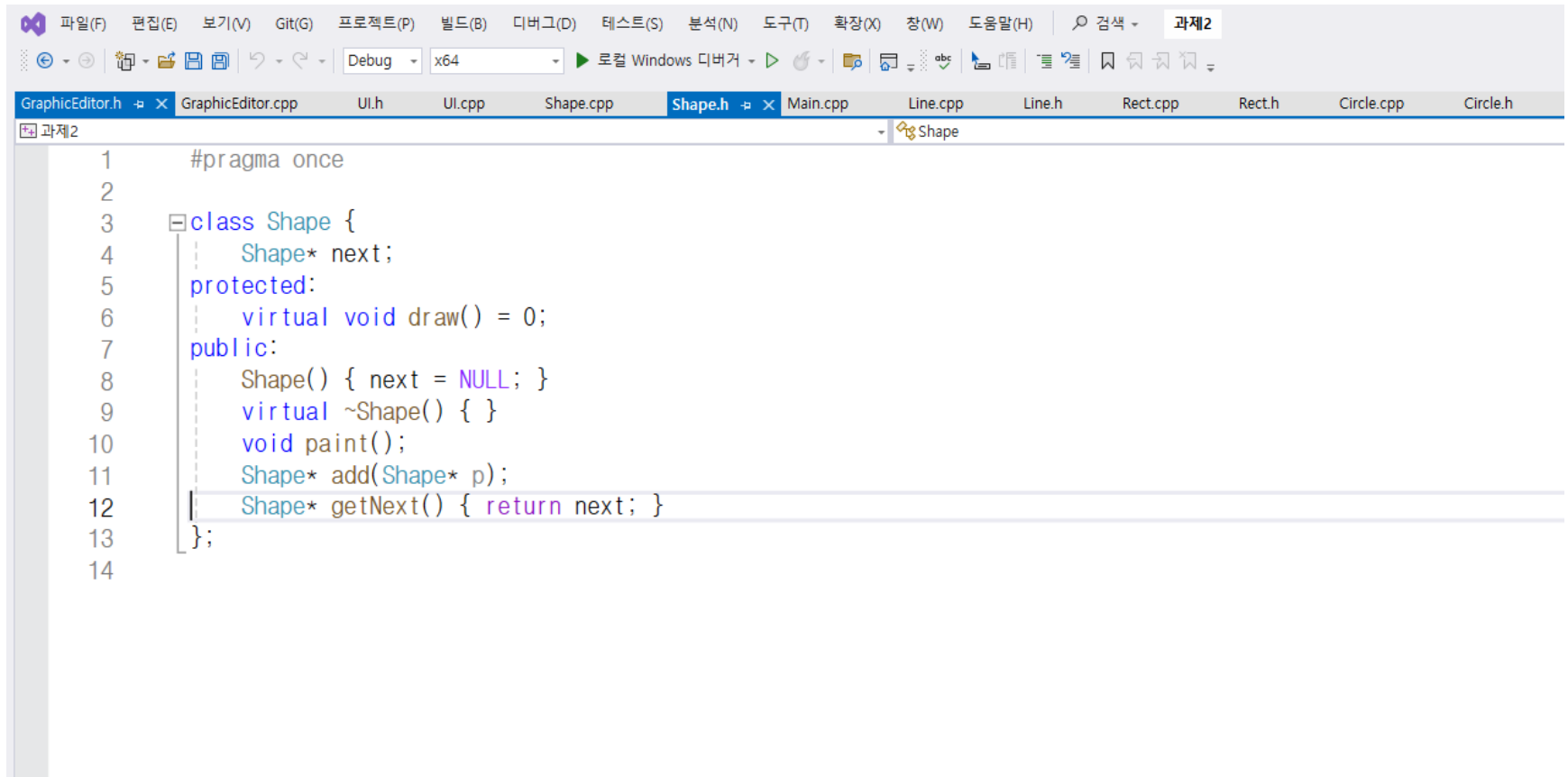


hw#2

과제 #2 P471 10번 문제

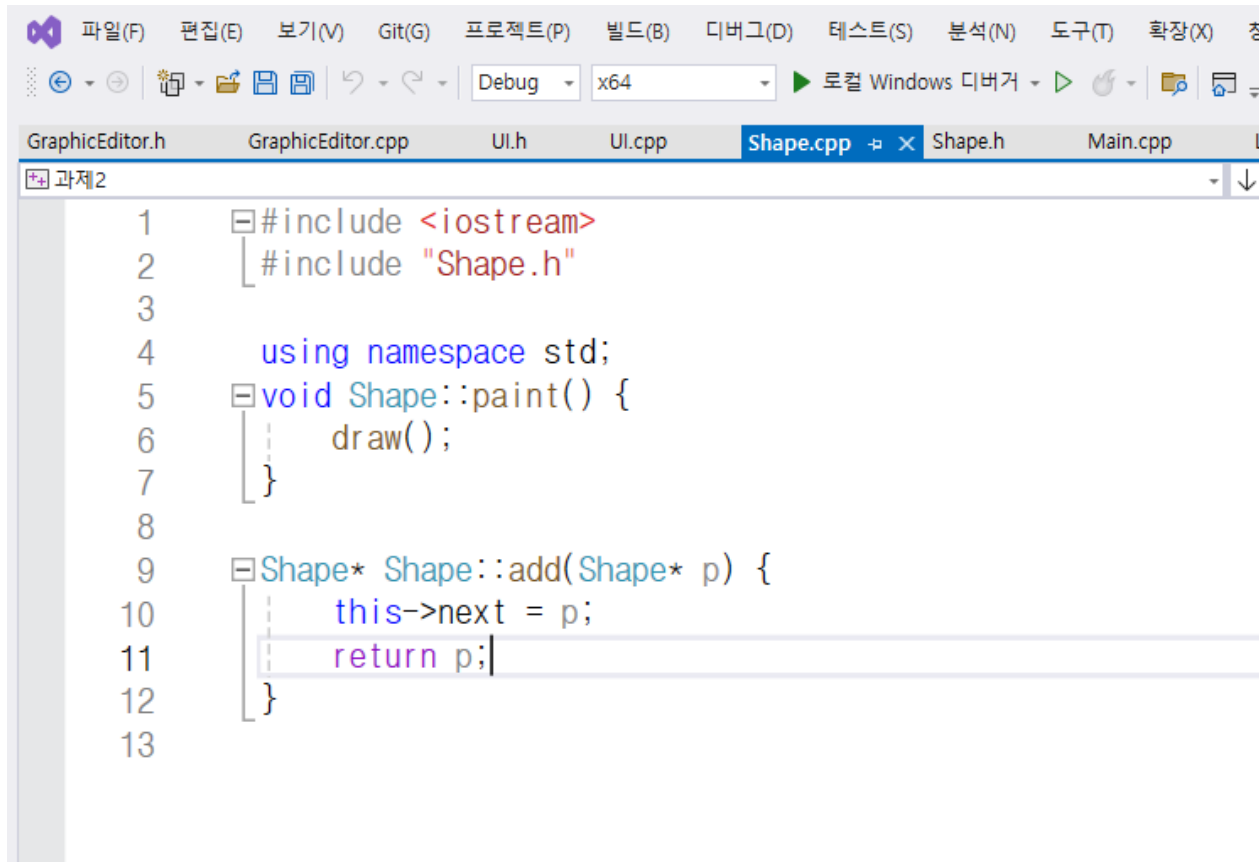
16조 이수영, 원준서

Shape.h



```
1  #pragma once
2
3  class Shape {
4      Shape* next;
5  protected:
6      virtual void draw() = 0;
7  public:
8      Shape() { next = NULL; }
9      virtual ~Shape() { }
10     void paint();
11     Shape* add(Shape* p);
12     Shape* getNext() { return next; }
13 };
14
```

Shape.cpp

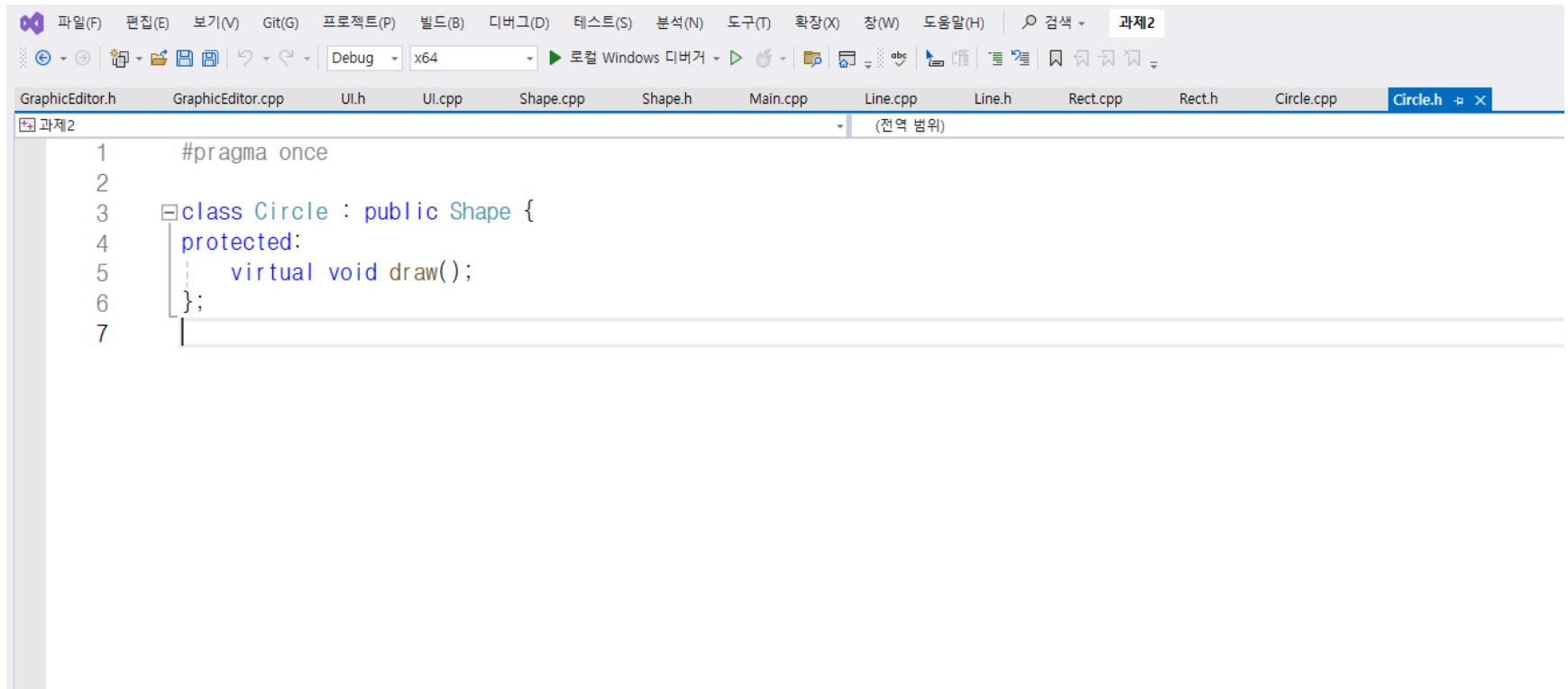


The image shows a screenshot of a C++ IDE with the following elements:

- Menu Bar:** 파일(F), 편집(E), 보기(V), Git(G), 프로젝트(P), 빌드(B), 디버그(D), 테스트(S), 분석(N), 도구(T), 확장(X), 창
- Toolbar:** Includes icons for file operations, a dropdown menu, a 'Debug' button, a 'x64' architecture selector, and a '로컬 Windows 디버거' (Local Windows Debugger) button.
- Tab Bar:** Contains tabs for GraphicEditor.h, GraphicEditor.cpp, UI.h, UI.cpp, Shape.cpp (active), Shape.h, and Main.cpp.
- Editor Window:** Titled '과제2' (Assignment 2), it displays the following C++ code:

```
1  #include <iostream>
2  #include "Shape.h"
3
4  using namespace std;
5  void Shape::paint() {
6      draw();
7  }
8
9  Shape* Shape::add(Shape* p) {
10     this->next = p;
11     return p;
12 }
13
```

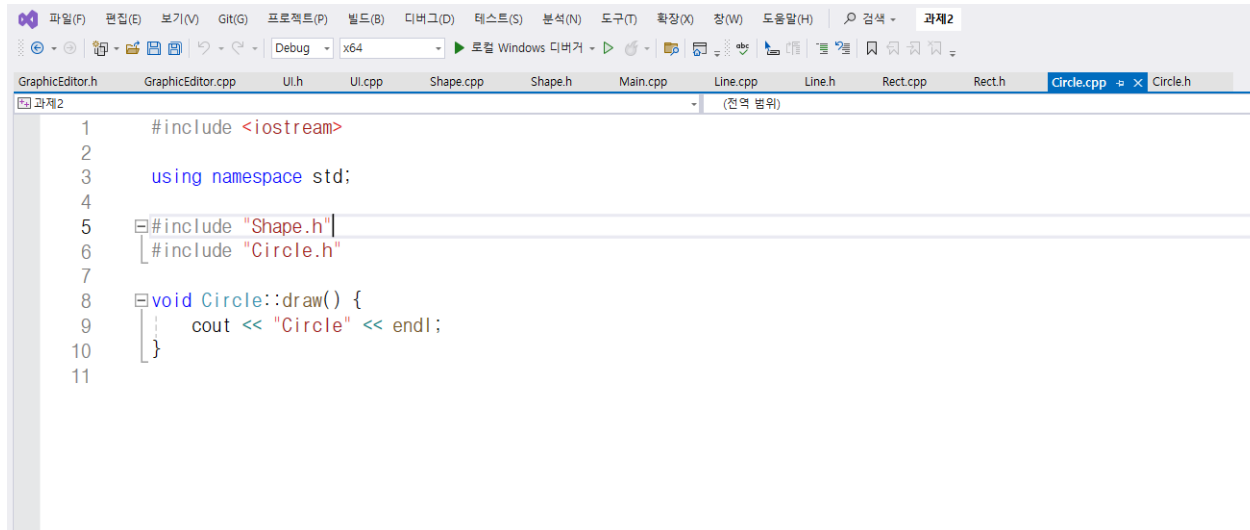
Circle.h



The screenshot shows a C++ IDE with a project named '과제2'. The file explorer on the left lists several files: GraphicEditor.h, GraphicEditor.cpp, UI.h, UI.cpp, Shape.cpp, Shape.h, Main.cpp, Line.cpp, Line.h, Rect.cpp, Rect.h, Circle.cpp, and Circle.h (which is the active file). The code editor displays the following C++ code for Circle.h:

```
1  #pragma once
2
3  class Circle : public Shape {
4  protected:
5      virtual void draw();
6  };
7
```

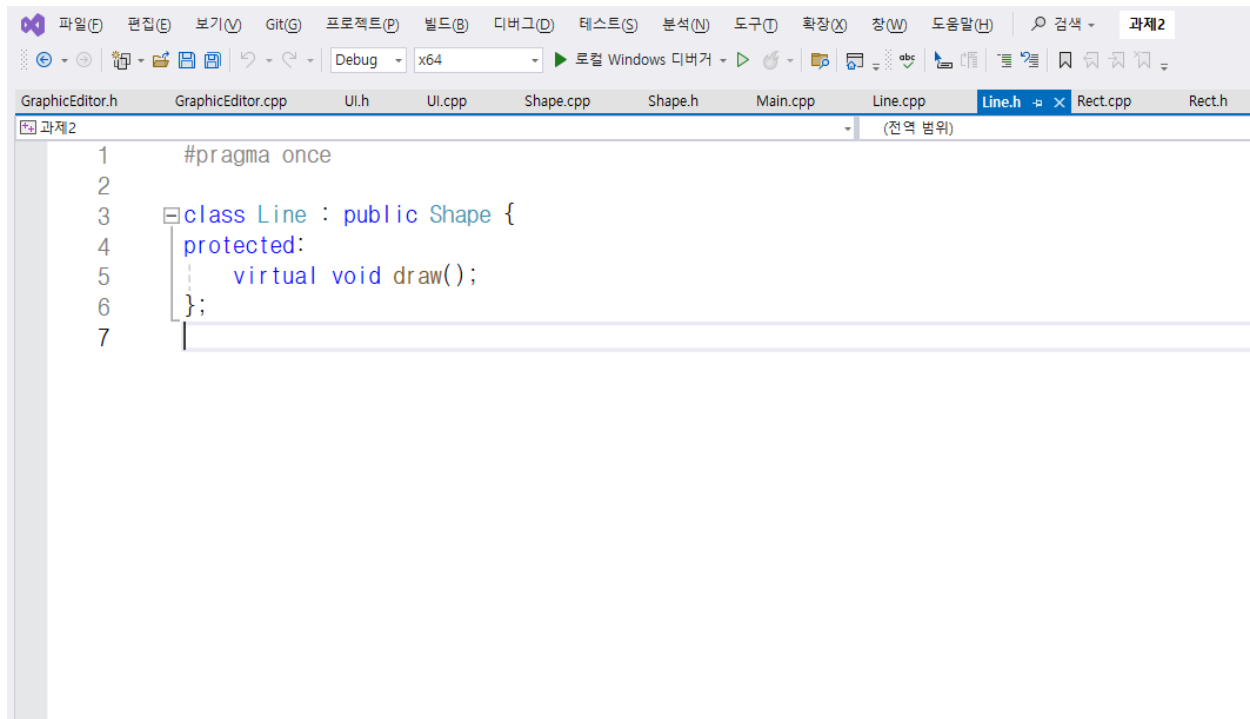
Circle.cpp



The screenshot shows a C++ IDE with a menu bar at the top containing options like '파일(F)', '편집(E)', '보기(V)', 'Git(G)', '프로젝트(P)', '빌드(B)', '디버그(D)', '테스트(S)', '분석(N)', '도구(T)', '확장(X)', '장(W)', '도움말(H)', and '검색'. Below the menu bar is a toolbar with icons for file operations, a 'Debug' dropdown, a 'x64' architecture dropdown, and a '로컬 Windows 디버거' button. The tab bar shows several open files: 'GraphicEditor.h', 'GraphicEditor.cpp', 'UI.h', 'UI.cpp', 'Shape.cpp', 'Shape.h', 'Main.cpp', 'Line.cpp', 'Line.h', 'Rect.cpp', 'Rect.h', and the active file 'Circle.cpp'. The code editor displays the following C++ code:

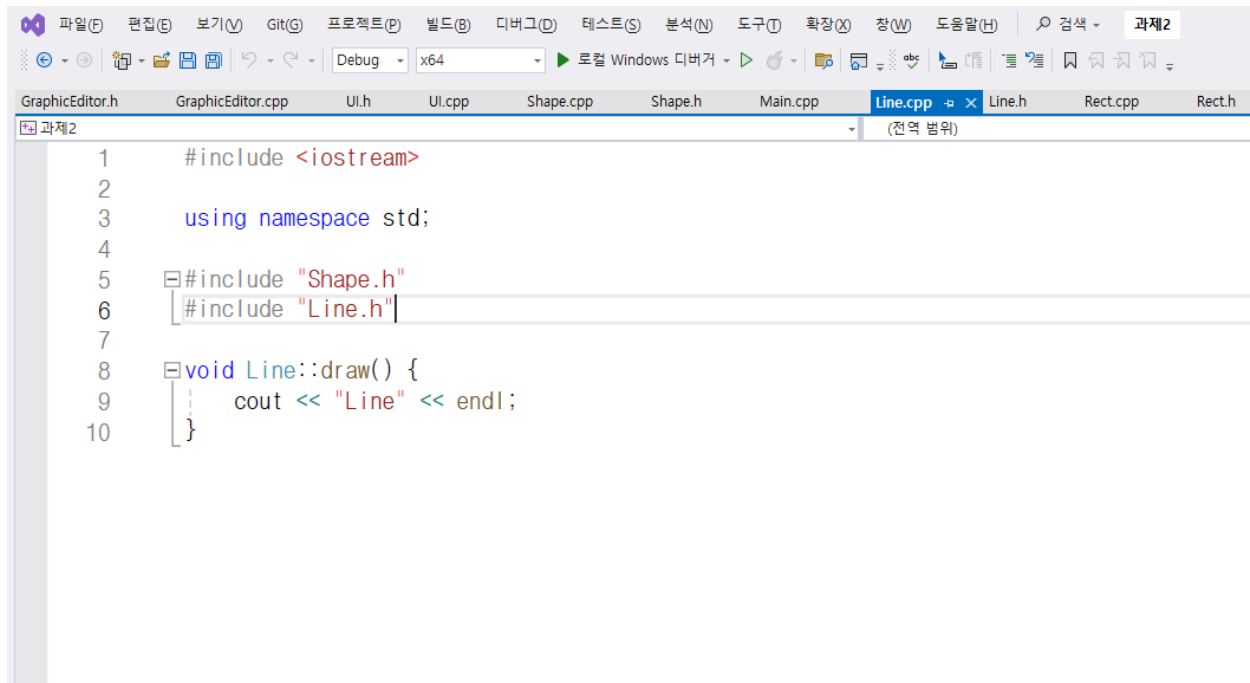
```
1  #include <iostream>
2
3  using namespace std;
4
5  #include "Shape.h"
6  #include "Circle.h"
7
8  void Circle::draw() {
9      cout << "Circle" << endl;
10 }
11
```

Line.h



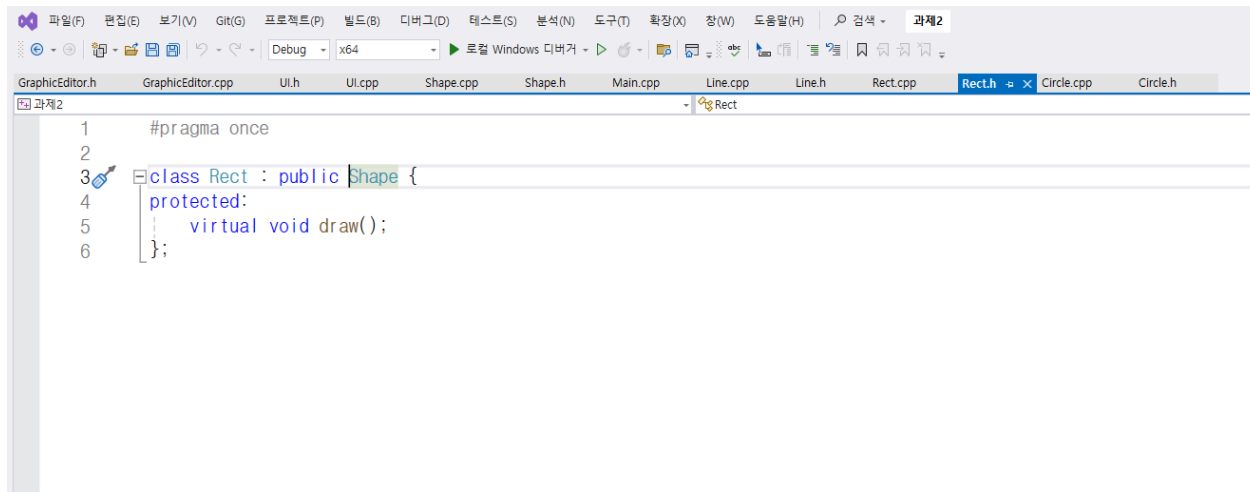
```
1  #pragma once
2
3  class Line : public Shape {
4  protected:
5      virtual void draw();
6  };
7
```

Line.cpp



```
1  #include <iostream>
2
3  using namespace std;
4
5  #include "Shape.h"
6  #include "Line.h"
7
8  void Line::draw() {
9      cout << "Line" << endl;
10 }
```

Rect.h

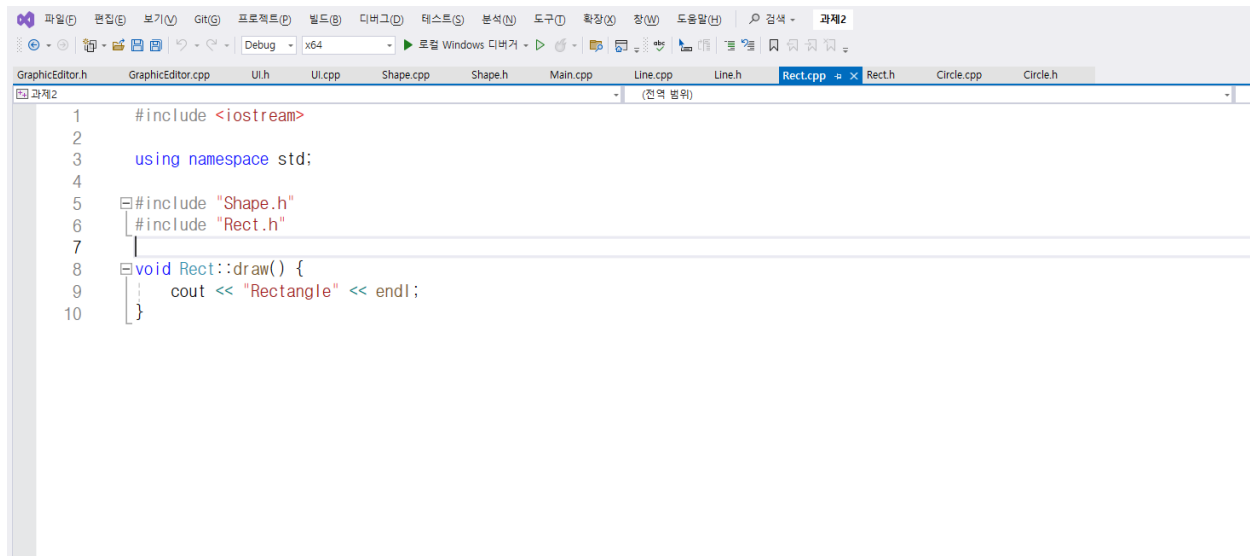


The screenshot shows a C++ IDE with the following components:

- Menu Bar:** 파일(F), 편집(E), 보기(V), Git(G), 프로젝트(P), 빌드(B), 디버그(D), 테스트(S), 분석(N), 도구(T), 확장(X), 창(W), 도움말(H), 검색, 과제2
- Toolbar:** Includes icons for file operations, compilation, and debugging. The status bar shows 'Debug' and 'x64'.
- Tab Bar:** Lists several files: GraphicEditor.h, GraphicEditor.cpp, UI.h, UI.cpp, Shape.cpp, Shape.h, Main.cpp, Line.cpp, Line.h, Rect.cpp, Rect.h (active), Circle.cpp, and Circle.h.
- Editor:** Displays the content of Rect.h with line numbers 1 through 6. The code is as follows:

```
1 #pragma once
2
3 class Rect : public Shape {
4     protected:
5         virtual void draw();
6 };
```

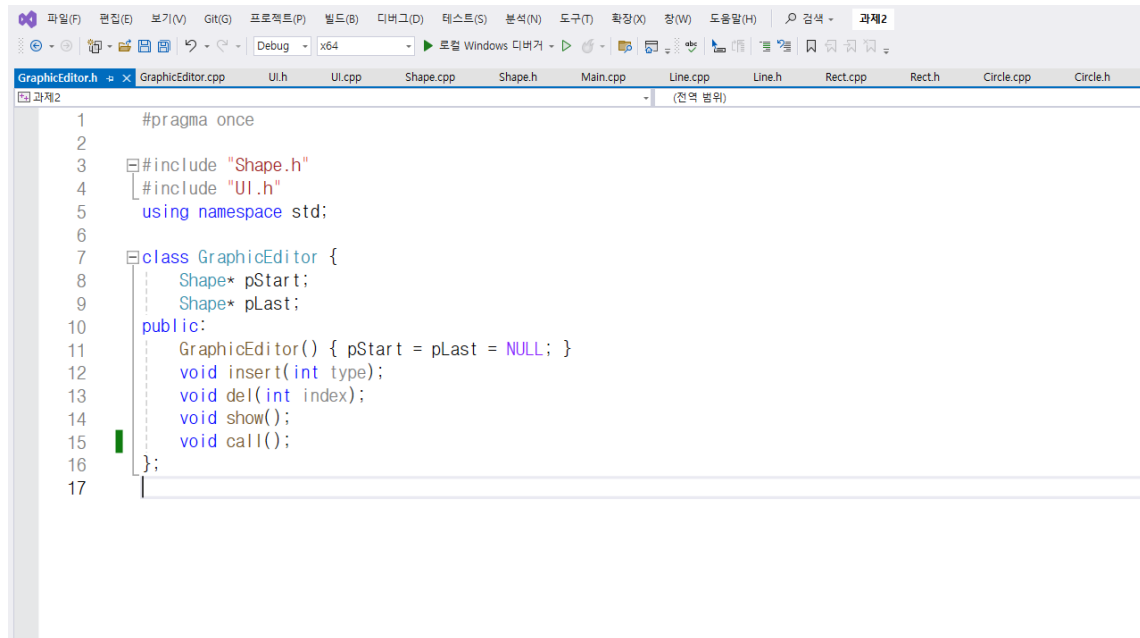

Rect.cpp



The screenshot shows a C++ IDE with a project named '과제2' (Assignment 2). The file explorer on the left lists the following files: GraphicEditor.h, GraphicEditor.cpp, UI.h, UI.cpp, Shape.cpp, Shape.h, Main.cpp, Line.cpp, Line.h, Rect.cpp (selected), Rect.h, Circle.cpp, and Circle.h. The main editor window displays the code for Rect.cpp, which includes iostream, uses the std namespace, includes Shape.h and Rect.h, and defines a draw() method for the Rect class that outputs 'Rectangle' to the console.

```
1  #include <iostream>
2
3  using namespace std;
4
5  #include "Shape.h"
6  #include "Rect.h"
7
8  void Rect::draw() {
9      cout << "Rectangle" << endl;
10 }
```

GraphicEditor.h

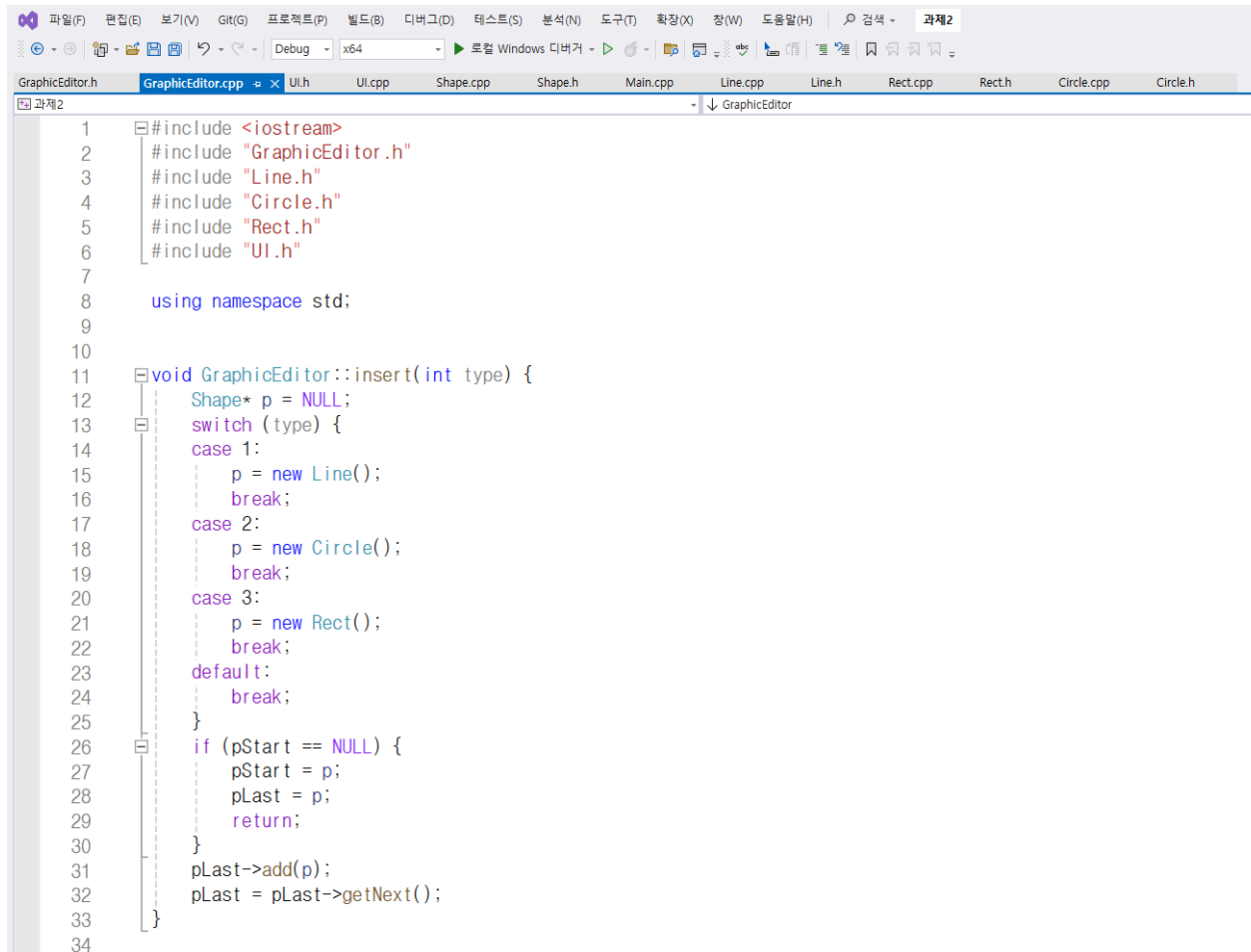


The image shows a screenshot of a C++ IDE, likely Visual Studio, with the 'GraphicEditor.h' file open. The file contains the following code:

```
1  #pragma once
2
3  #include "Shape.h"
4  #include "UI.h"
5  using namespace std;
6
7  class GraphicEditor {
8  public:
9      Shape* pStart;
10     Shape* pLast;
11     GraphicEditor() { pStart = pLast = NULL; }
12     void insert(int type);
13     void del(int index);
14     void show();
15     void call();
16 };
17
```

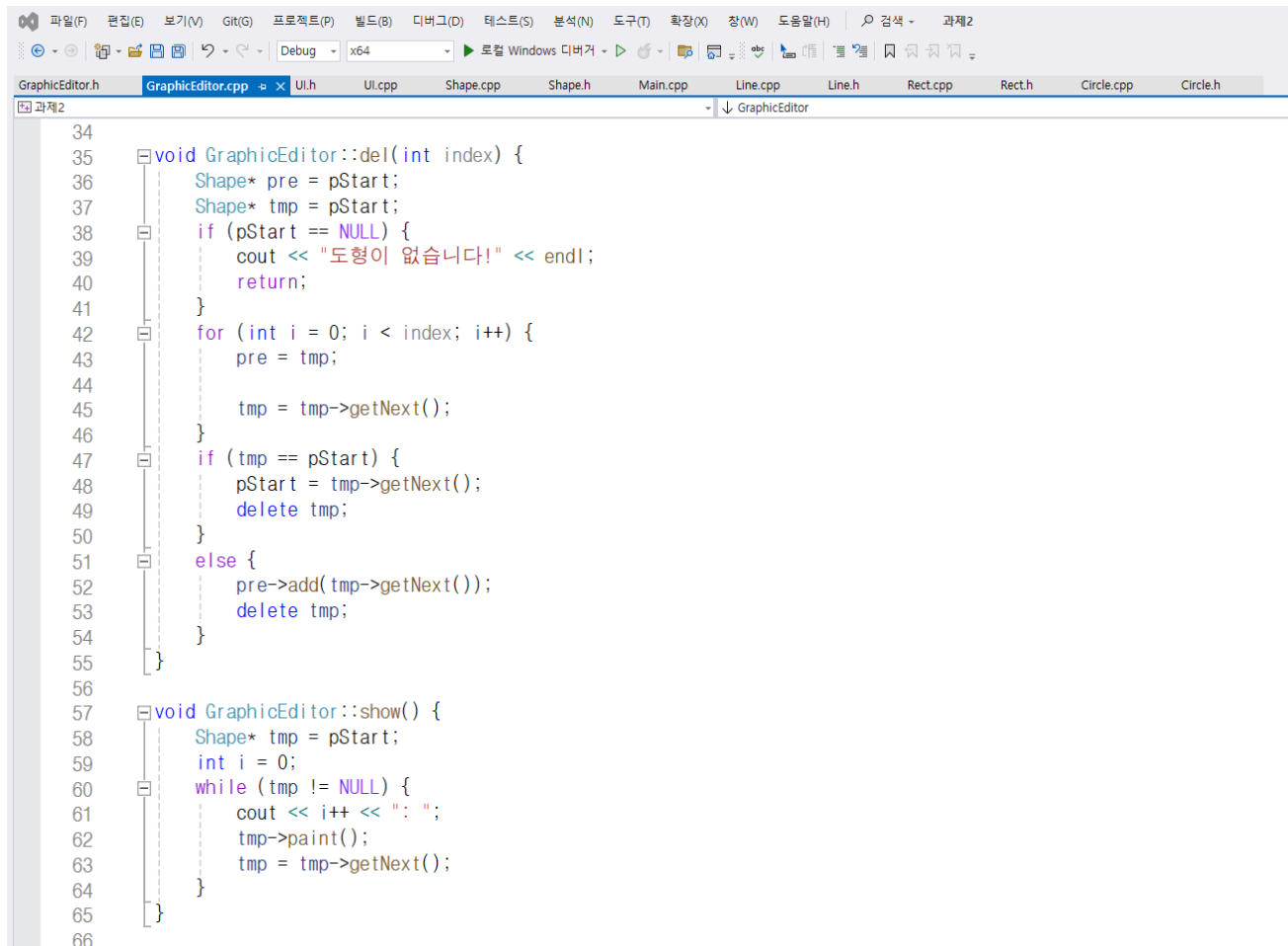
The IDE interface includes a menu bar at the top with options like '파일(F)', '편집(E)', '보기(V)', 'Git(G)', '프로젝트(P)', '빌드(B)', '디버그(D)', '테스트(S)', '분석(N)', '도구(T)', '확장(X)', '창(W)', '도움말(H)', and '검색'. Below the menu bar is a toolbar with icons for various development tasks. The 'Debug' window is active, showing the '로컬 Windows 디버거' (Local Windows Debugger) with a 'x64' architecture. The 'Solution Explorer' on the right shows the project structure, including files like 'GraphicEditor.h', 'GraphicEditor.cpp', 'UI.h', 'UI.cpp', 'Shape.cpp', 'Shape.h', 'Main.cpp', 'Line.cpp', 'Line.h', 'Rect.cpp', 'Rect.h', 'Circle.cpp', and 'Circle.h'. The 'GraphicEditor.h' file is currently selected and its content is displayed in the main editor area.

GraphicEditor.cpp(1)



```
1  #include <iostream>
2  #include "GraphicEditor.h"
3  #include "Line.h"
4  #include "Circle.h"
5  #include "Rect.h"
6  #include "UI.h"
7
8  using namespace std;
9
10
11 void GraphicEditor::insert(int type) {
12     Shape* p = NULL;
13     switch (type) {
14     case 1:
15         p = new Line();
16         break;
17     case 2:
18         p = new Circle();
19         break;
20     case 3:
21         p = new Rect();
22         break;
23     default:
24         break;
25     }
26     if (pStart == NULL) {
27         pStart = p;
28         pLast = p;
29         return;
30     }
31     pLast->add(p);
32     pLast = pLast->getNext();
33 }
34
```

GraphicEditor.cpp(2)

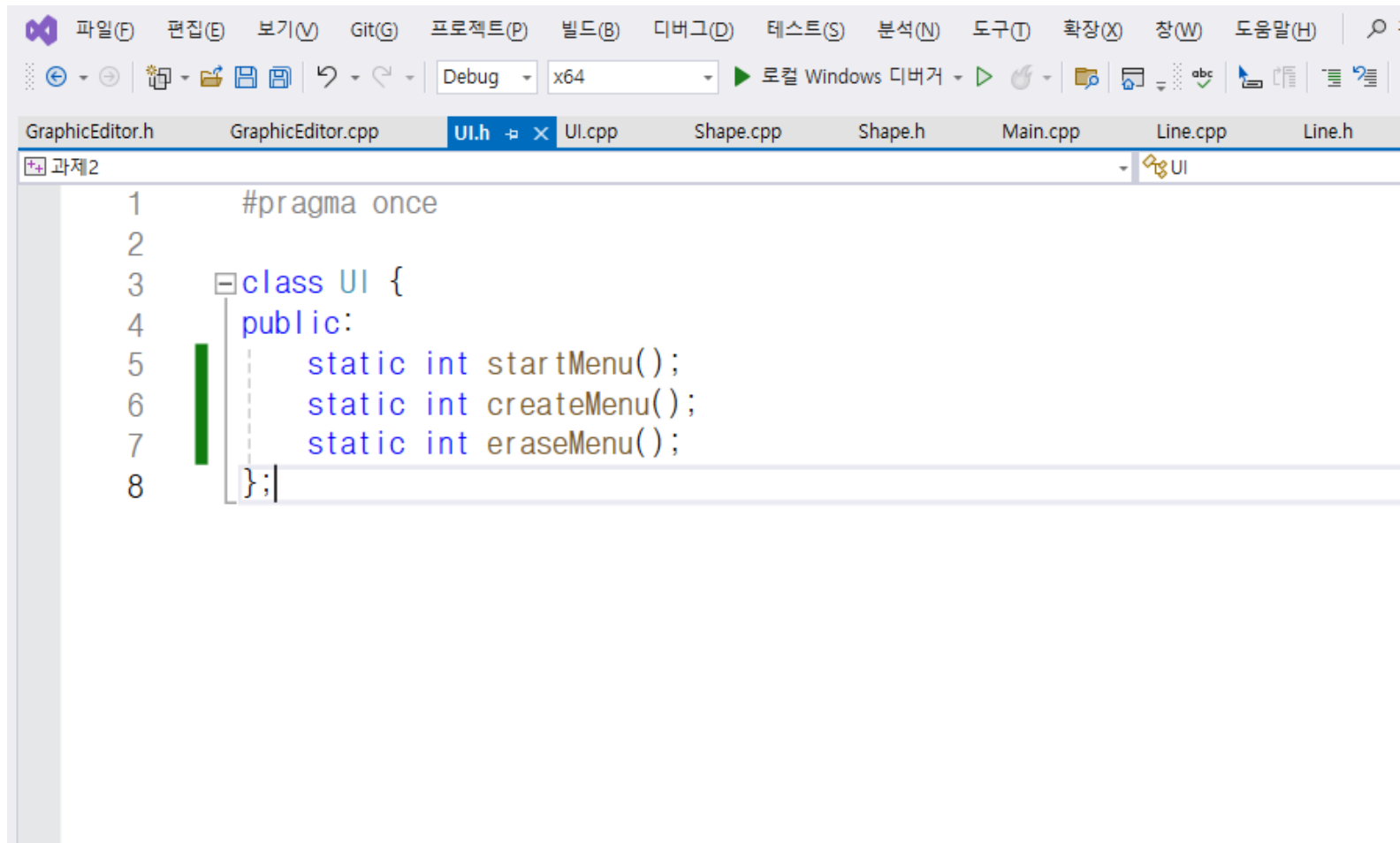


```
34
35 void GraphicEditor::del(int index) {
36     Shape* pre = pStart;
37     Shape* tmp = pStart;
38     if (pStart == NULL) {
39         cout << "도형이 없습니다!" << endl;
40         return;
41     }
42     for (int i = 0; i < index; i++) {
43         pre = tmp;
44
45         tmp = tmp->getNext();
46     }
47     if (tmp == pStart) {
48         pStart = tmp->getNext();
49         delete tmp;
50     }
51     else {
52         pre->add(tmp->getNext());
53         delete tmp;
54     }
55 }
56
57 void GraphicEditor::show() {
58     Shape* tmp = pStart;
59     int i = 0;
60     while (tmp != NULL) {
61         cout << i++ << ": ";
62         tmp->paint();
63         tmp = tmp->getNext();
64     }
65 }
66
```

GraphicEditor.cpp(3)

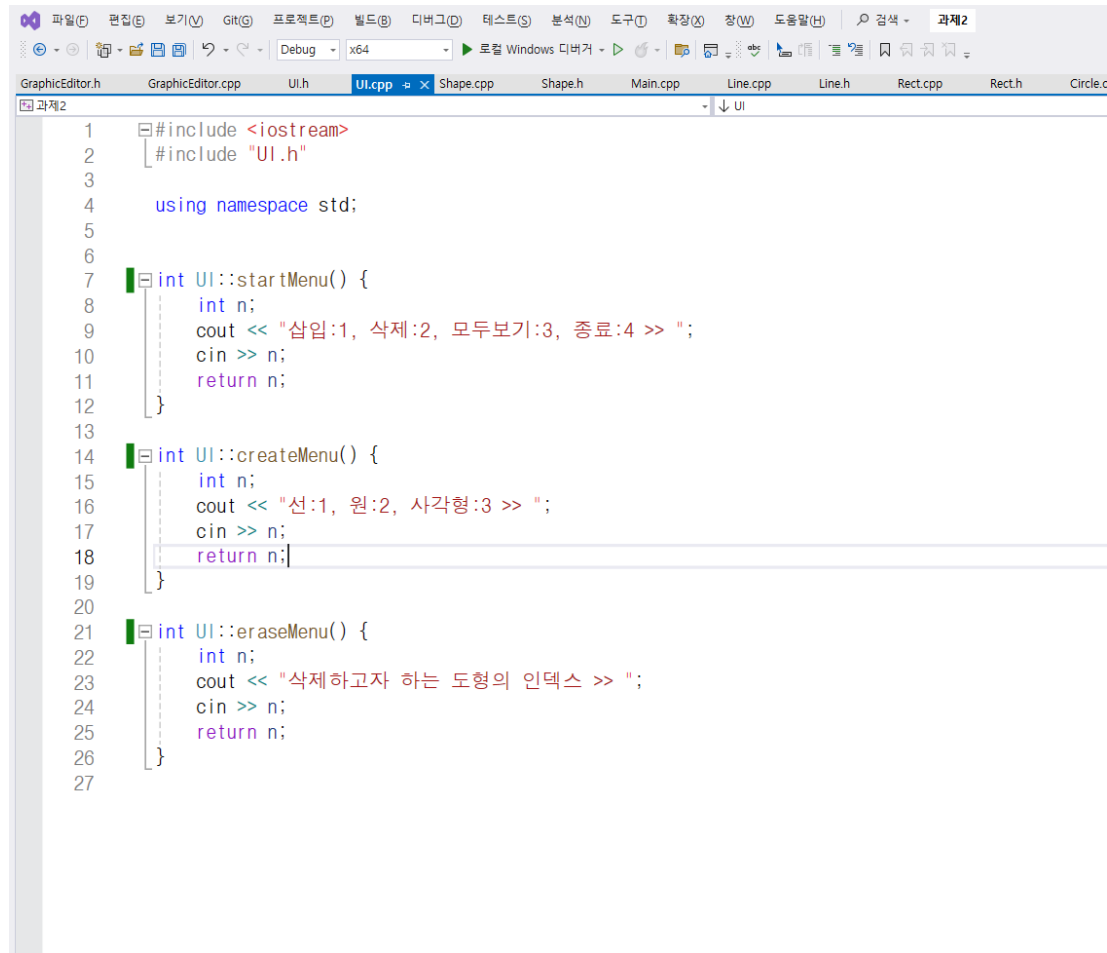
```
66
67 void GraphicEditor::call() {
68     cout << "그래픽 에디터입니다." << endl;
69     int menu, index, type;
70     while (true) {
71         menu = UI::startMenu();
72         switch (menu) {
73             case 1:
74                 type = UI::createMenu();
75                 insert(type);
76                 break;
77             case 2:
78                 index = UI::eraseMenu();
79                 del(index);
80                 break;
81             case 3:
82                 show();
83                 break;
84             default:
85                 return;
86         }
87     }
88 }
89
```

UI.h



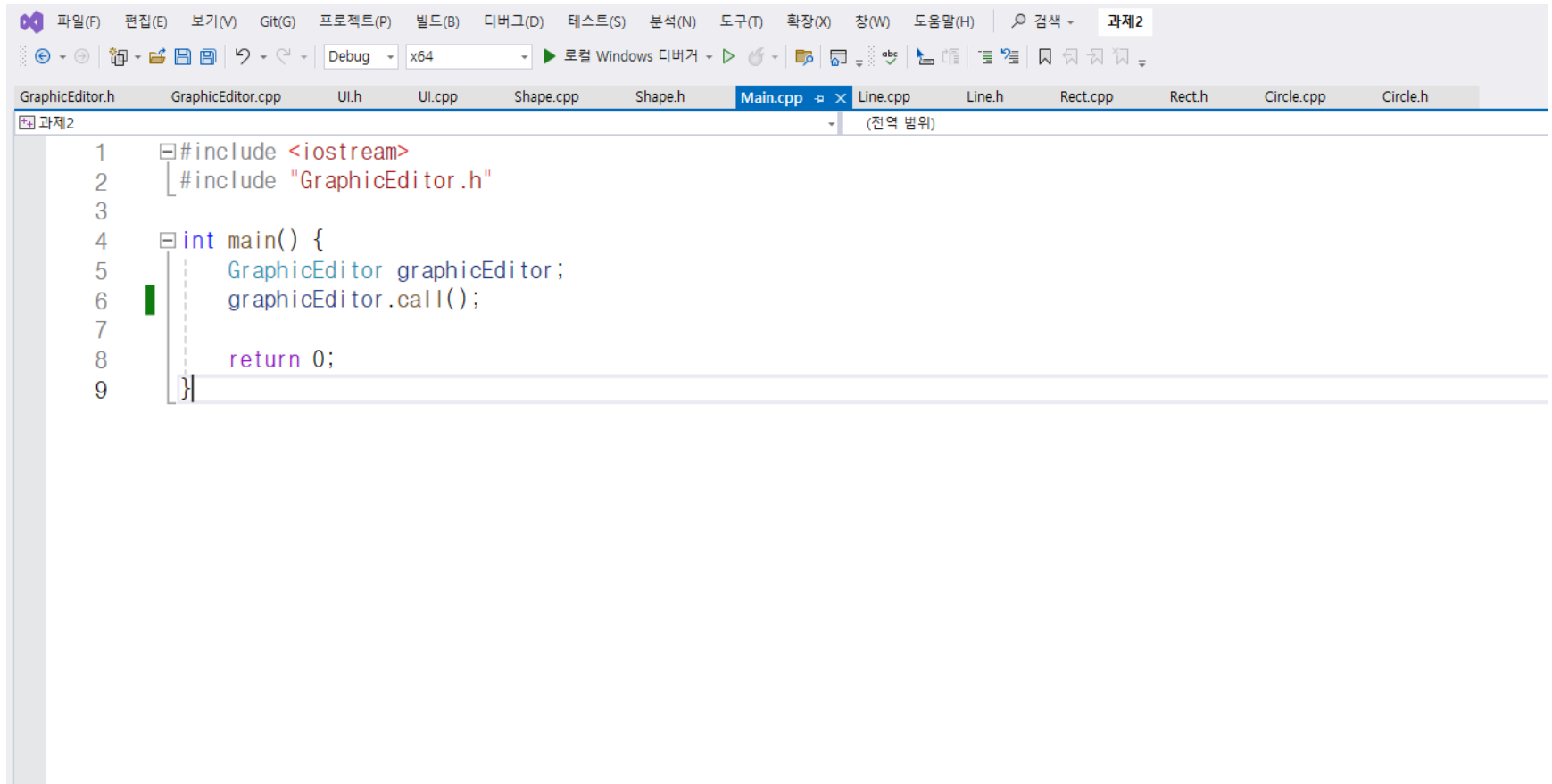
```
1  #pragma once
2
3  class UI {
4  public:
5      static int startMenu();
6      static int createMenu();
7      static int eraseMenu();
8  };
```

UI.cpp



```
1  #include <iostream>
2  #include "UI.h"
3
4  using namespace std;
5
6
7  int UI::startMenu() {
8      int n;
9      cout << "삼십:1, 삭제:2, 모두보기:3, 종료:4 >> ";
10     cin >> n;
11     return n;
12 }
13
14 int UI::createMenu() {
15     int n;
16     cout << "선:1, 원:2, 사각형:3 >> ";
17     cin >> n;
18     return n;
19 }
20
21 int UI::eraseMenu() {
22     int n;
23     cout << "삭제하고자 하는 도형의 인덱스 >> ";
24     cin >> n;
25     return n;
26 }
27
```

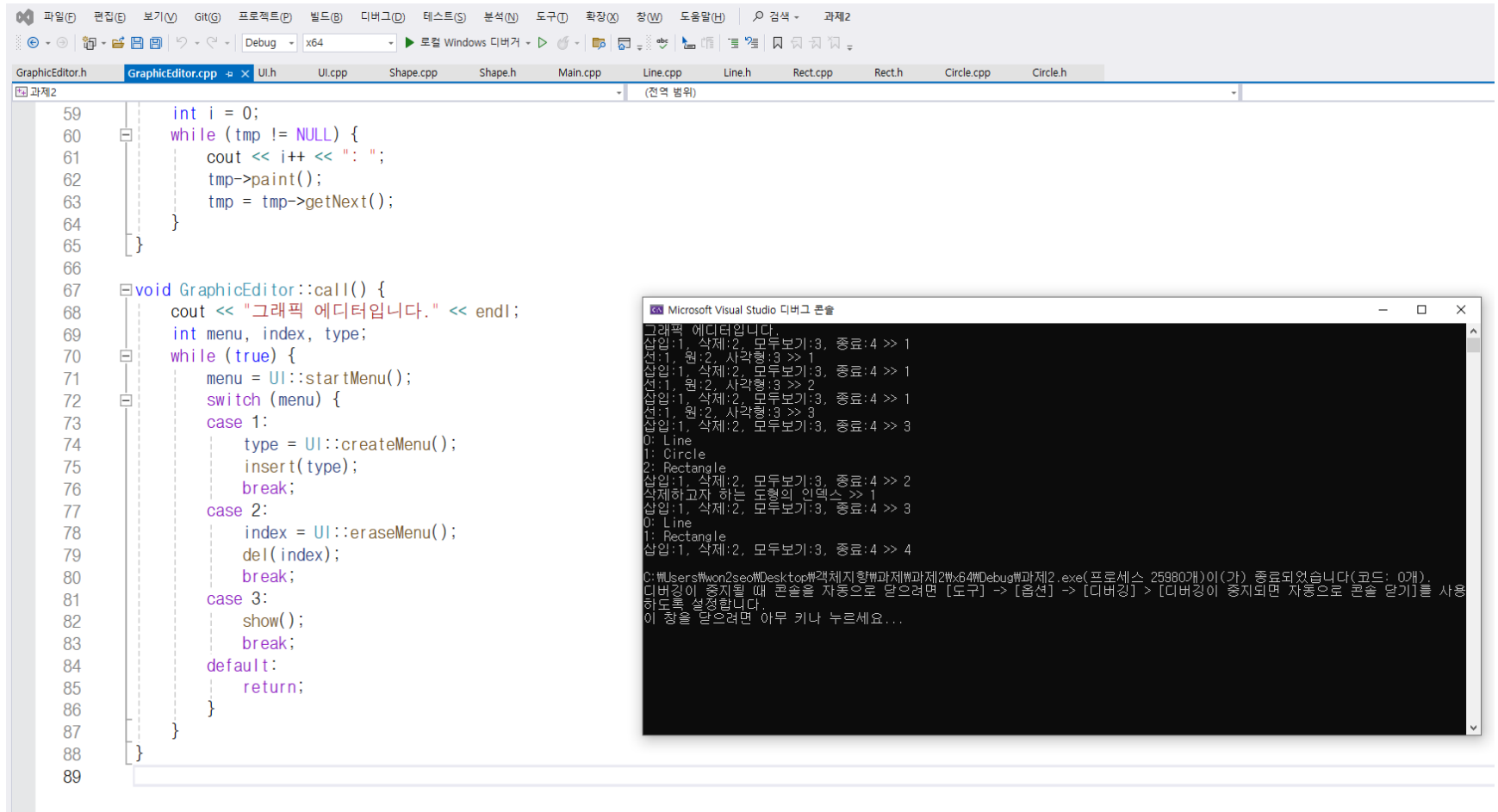
Main.cpp



```
1 #include <iostream>
2 #include "GraphicEditor.h"
3
4 int main() {
5     GraphicEditor graphicEditor;
6     graphicEditor.call();
7
8     return 0;
9 }
```

The screenshot shows a C++ IDE with a menu bar (파일(F), 편집(E), 보기(V), Git(G), 프로젝트(P), 빌드(B), 디버그(D), 테스트(S), 분석(N), 도구(T), 확장(X), 창(W), 도움말(H)) and a toolbar. The 'Main.cpp' file is selected in the tab bar. The code editor displays the following code:

실행결과



The image shows a Visual Studio IDE with a C++ project named '과제2'. The main code file is 'GraphicEditor.cpp', which contains the following code:

```
59     int i = 0;
60     while (tmp != NULL) {
61         cout << i++ << ": ";
62         tmp->paint();
63         tmp = tmp->getNext();
64     }
65 }
66
67 void GraphicEditor::call() {
68     cout << "그래픽 에디터입니다." << endl;
69     int menu, index, type;
70     while (true) {
71         menu = UI::startMenu();
72         switch (menu) {
73             case 1:
74                 type = UI::createMenu();
75                 insert(type);
76                 break;
77             case 2:
78                 index = UI::eraseMenu();
79                 del(index);
80                 break;
81             case 3:
82                 show();
83                 break;
84             default:
85                 return;
86         }
87     }
88 }
89 }
```

The debug console output shows the execution of the program, displaying the menu and the results of the operations:

```
Microsoft Visual Studio 디버그 콘솔
그래픽 에디터입니다.
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 1
선:1, 원:2, 사각형:3 >> 1
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 1
선:1, 원:2, 사각형:3 >> 2
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 1
선:1, 원:2, 사각형:3 >> 3
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 3
0: Line
1: Circle
2: Rectangle
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 2
삭제하고자 하는 도형의 인덱스 >> 1
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 3
0: Line
1: Rectangle
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 4
C:\Users\won2\Desktop\과제2\과제2\Debug\과제2.exe (프로세스 25980개)이 (가) 종료되었습니다(코드: 0개).
디버깅이 중지될 때 콘솔을 자동으로 닫으려면 [도구] -> [옵션] -> [디버깅] > [디버깅이 중지되면 자동으로 콘솔 닫기]를 사용
하도록 설정합니다.
이 창을 닫으려면 아무 키나 누르세요...
```