

2023.2

Principle of Programming Language

Programming Assignment #1

External Documentation

| | |
|----------|-----|
| 20184256 | 박성민 |
| 20226041 | 김규리 |

MacOS Terminal 에서 실행

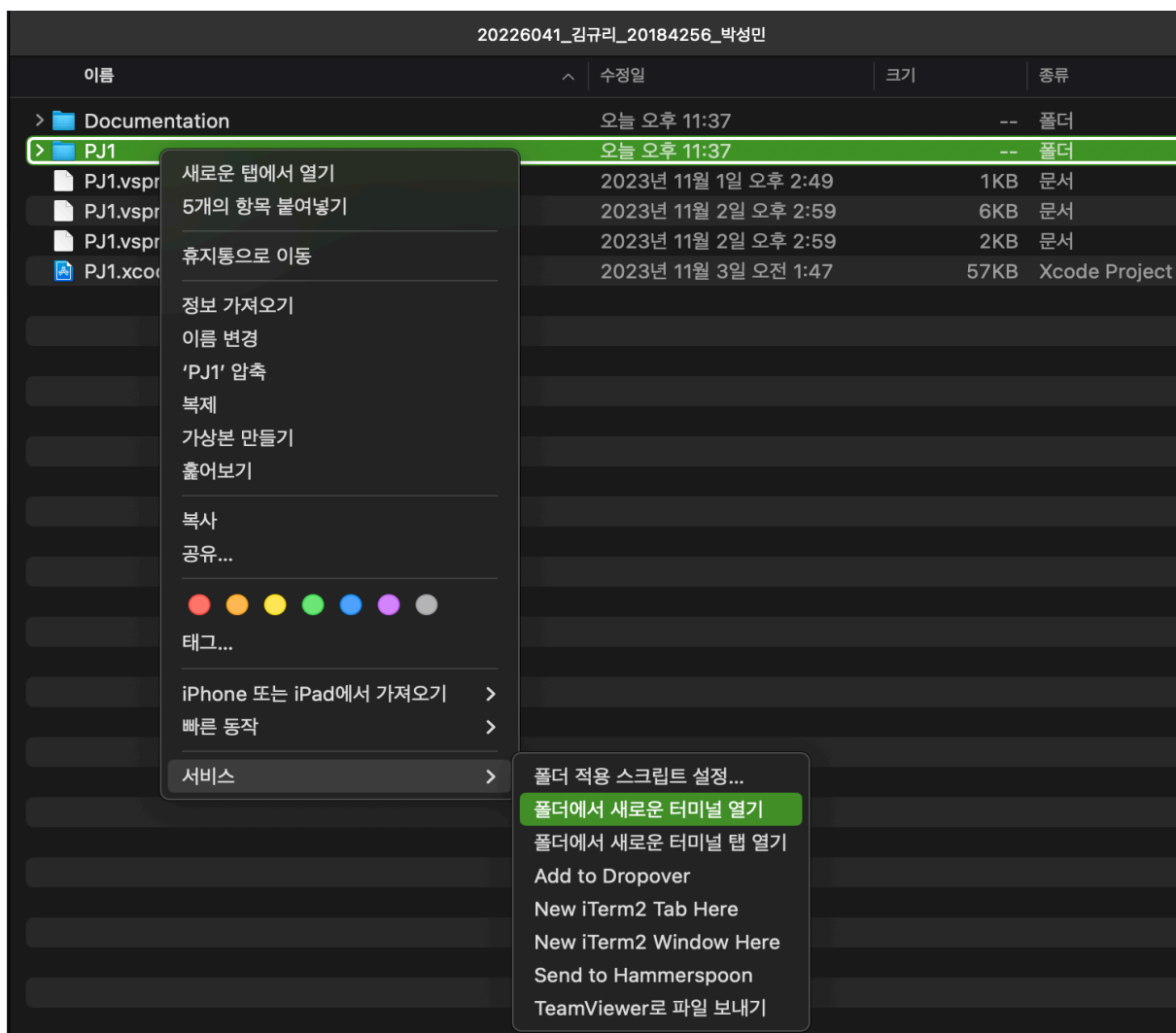
본 프로젝트는 MacOS 환경에서는 XCode 14.0, C++ GNU++20 의 환경에서 진행되었다.

우선, 터미널에 "g++ -v" 명령어를 입력하여, g++ 컴파일러가 설치가 되어있는지 확인한다.

```
sungmin@bagseongmin-ui-maegbug PJ1 % g++ -v
Apple clang version 15.0.0 (clang-1500.0.40.1)
Target: x86_64-apple-darwin23.0.0
Thread model: posix
InstalledDir: /Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin
```

만약 g++ 컴파일러가 설치가 되어있지 않은 경우에는 "xcode-select --install" 명령어를 터미널에 입력해, Xcode Command Line Tool 을 설치해주면 된다.

설치가 되었다면, 프로젝트 폴더를 우클릭 > 서비스 > 폴더에서 새로운 터미널 열기로 프로젝트 폴더 위치로 터미널을 열어준다.



터미널에서 아래 명령어를 복사하여 입력하면 된다. 명령어 가장 뒤의 input.txt 자리에 원하는 파일명.txt 를 입력해 임의의 파일을 실행시킬 수도 있다.

```
g++ -std=c++20 -stdlib=libc++ main.cpp lexical_analyzer.cpp  
parser.cpp -o pml && ./pml input.txt
```

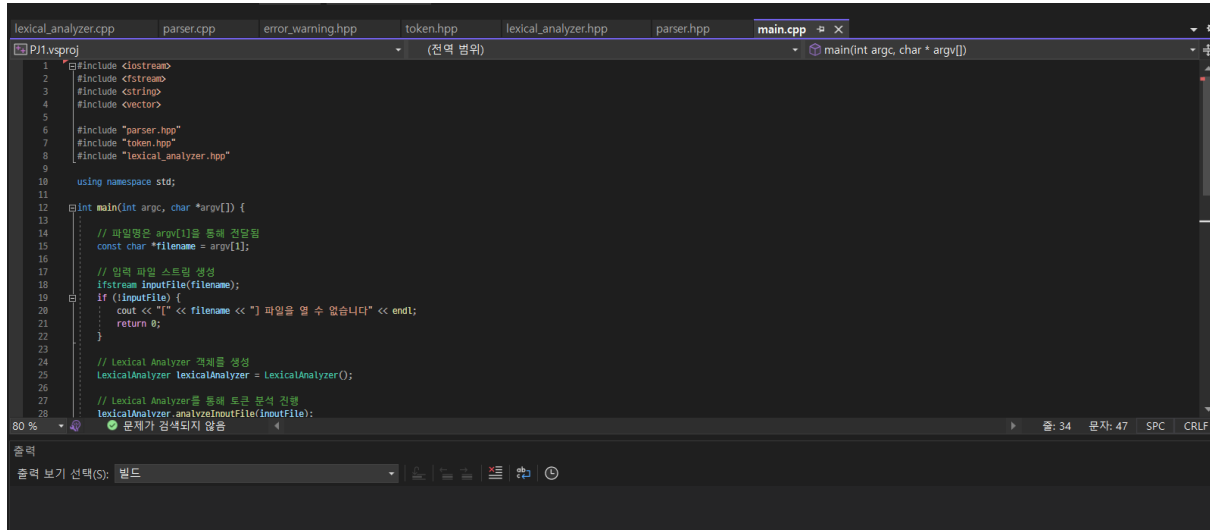
터미널에 아래처럼 실행 결과를 확인할 수 있다.

```
operand1:=3;  
ID: 1; CONST:1; OP: 0  
(OK)  
  
operand2:=operand1+2;  
ID: 2; CONST:1; OP: 1  
[WARNING] : Remove given invalid operation (additional op / invalid position)  
  
target:=operand1+operand2*3  
ID: 3; CONST:1; OP: 2  
(OK)  
  
operand1 : 3  
operand2 : 5  
target : 18  
sungmin@bagseongmin-ui-maegbug 20226041_김규리_20184256_박성민 %
```

Visual Studio 환경에서의 실행

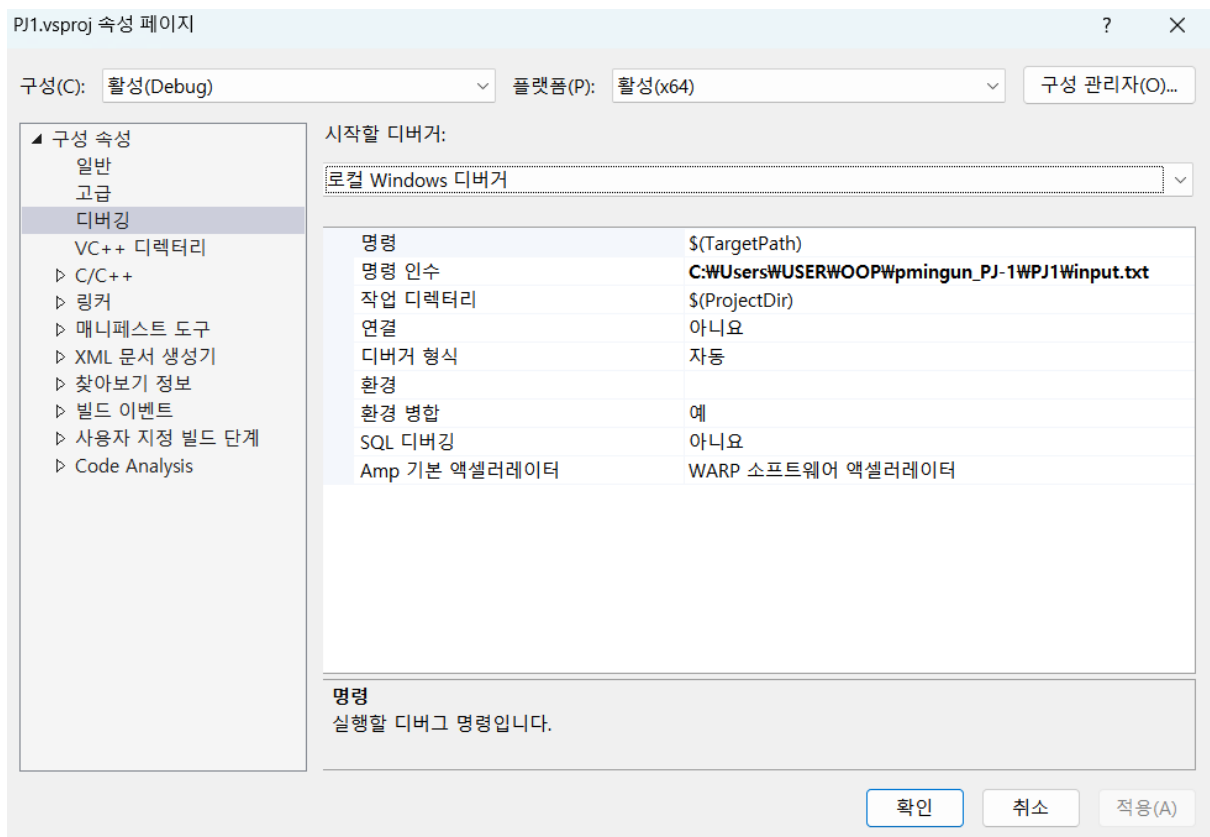
본 프로젝트는 Window 환경에서는 Visual Studio 2022, C++ 14 의 환경에서 진행되었다.

첨부된 파일 중 sln 확장자의 파일을 열어 Visual Studio 프로그램을 실행시킨다.

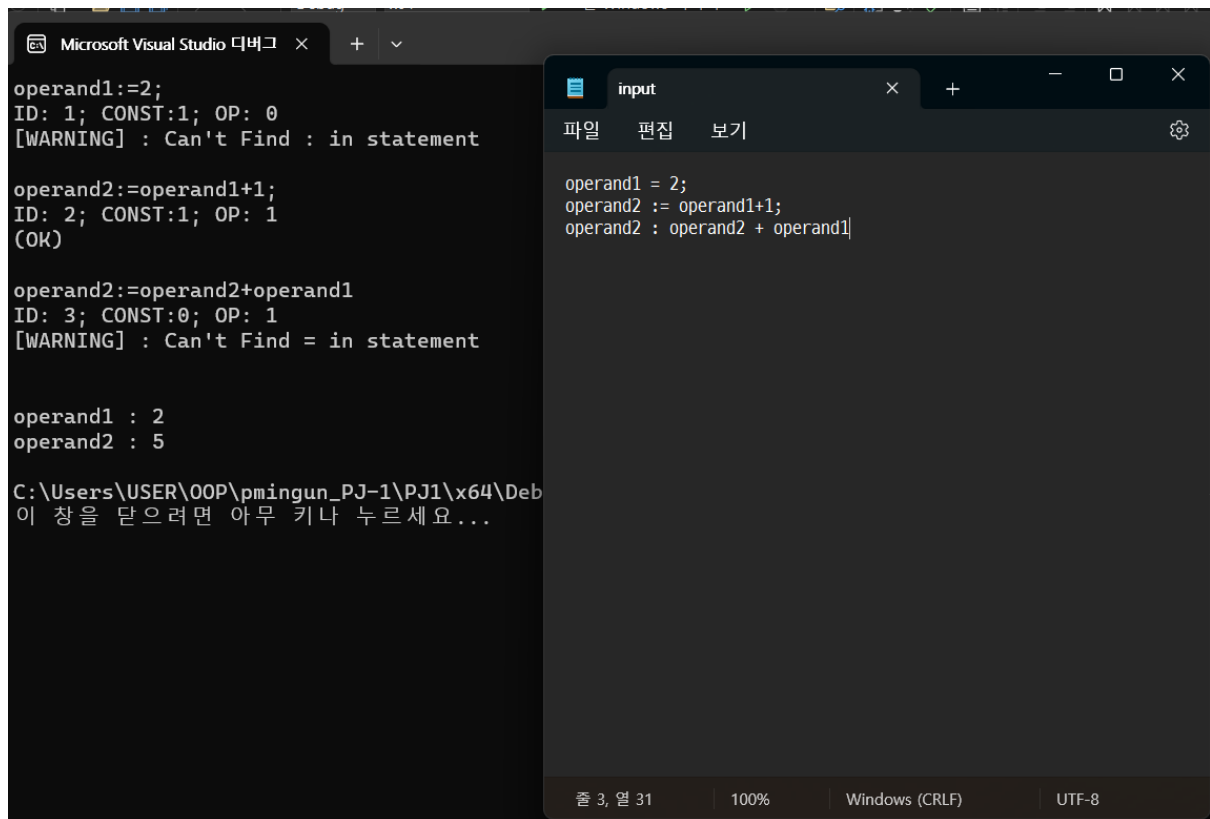


```
1 #include <iostream>
2 #include <fstream>
3 #include <string>
4 #include <vector>
5
6 #include "parser.hpp"
7 #include "token.hpp"
8 #include "lexical_analyzer.hpp"
9
10 using namespace std;
11
12 int main(int argc, char *argv[]) {
13     // 파일명은 argv[1]을 통해 전달됨
14     const char *filename = argv[1];
15
16     // 입력 파일 스트림 생성
17     ifstream inputFile(filename);
18     if (!inputFile) {
19         cout << "[*] << filename << "]" 파일을 열 수 없습니다" << endl;
20         return 0;
21     }
22
23     // Lexical Analyzer 객체 생성
24     LexicalAnalyzer lexicalAnalyzer = LexicalAnalyzer();
25
26     // Lexical Analyzer를 통해 토큰 분석 진행
27     lexicalAnalyzer.analyze(inputFile);
28 }
```

프로젝트 -> 프로젝트 속성으로 이동하여 명령인수를 추가한다. 아래 input.txt 예시에서처럼 input file 로 사용할 txt 파일의 경로를 인수로 작성하면 원하는 파일을 프로그램에 전달할 수 있다.



컴파일 후 실행(Ctrl+F5)으로 실행한다.



The screenshot shows the Microsoft Visual Studio IDE. On the left, the '디버그' (Debug) window is open, displaying the execution state of a program. It shows three statements being executed, each with its ID, constant value, and operation. The first statement is 'operand1:=2;' with ID 1, CONST 1, and OP 0. The second is 'operand2:=operand1+1;' with ID 2, CONST 1, and OP 1. The third is 'operand2:=operand2+operand1' with ID 3, CONST 0, and OP 1. Below these, the current values of 'operand1' (2) and 'operand2' (5) are shown. At the bottom of the debug window, the file path 'C:\Users\USER\OOP\pmingun_PJ-1\PJ1\x64\Deb' is visible, along with a message in Korean: '이 창을 닫으려면 아무 키나 누르세요...' (Press any key to close this window...). On the right, the 'input' code editor window is open, showing the source code: 'operand1 = 2;', 'operand2 := operand1+1;', and 'operand2 : operand2 + operand1|'. The status bar at the bottom indicates '줄 3, 열 31' (Line 3, Column 31), '100%' zoom, 'Windows (CRLF)' line endings, and 'UTF-8' encoding.

```
operand1:=2;
ID: 1; CONST:1; OP: 0
[WARNING] : Can't Find : in statement

operand2:=operand1+1;
ID: 2; CONST:1; OP: 1
(OK)

operand2:=operand2+operand1
ID: 3; CONST:0; OP: 1
[WARNING] : Can't Find = in statement

operand1 : 2
operand2 : 5

C:\Users\USER\OOP\pmingun_PJ-1\PJ1\x64\Deb
이 창을 닫으려면 아무 키나 누르세요...
```

```
input
operand1 = 2;
operand2 := operand1+1;
operand2 : operand2 + operand1|
```

줄 3, 열 31 | 100% | Windows (CRLF) | UTF-8