Pervasive Elastic MetaLearning Laboratory
Department of Computer Engineering
Hongik University

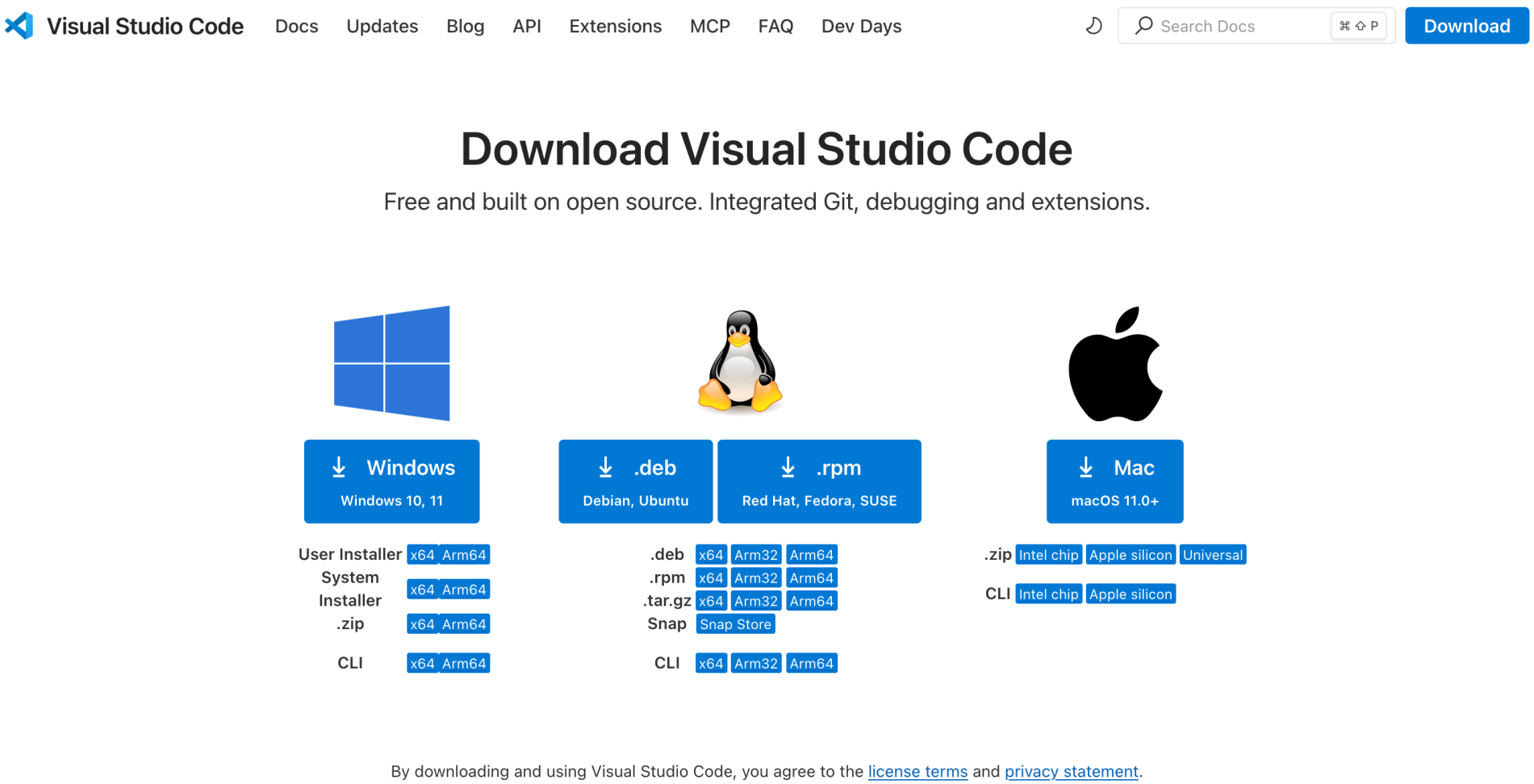
VSCode를 활용한 C++

2025 DS Ex1

Visual Studio Code

- Microsoft 사의 텍스트 에디터
- IDE X → 약간의 환경 설정이 필요함
- 여러 플러그인을 사용해 IDE처럼 사용가능
- 오픈 소스
- Git, 터미널 통합 등...
- 가벼움
- Github에서 학생 인증 시 Copilot 무료 사용 가능
- https://code.visualstudio.com/

VSCode 설치


Visual Studio Code
Docs
Updates
Blog
API
Extensions
MCP
FAQ
Dev Days
Search Docs
⌘ ⇧ P
Download
Download Visual Studio Code
Free and built on open source. Integrated Git, debugging and extensions.
Windows
Windows 10, 11
.deb
Debian, Ubuntu
.rpm
Red Hat, Fedora, SUSE
Mac
macOS 11.0+
User Installer x64 Arm64
System Installer x64 Arm64
.zip x64 Arm64
CLI x64 Arm64
.deb x64 Arm32 Arm64
.rpm x64 Arm32 Arm64
.tar.gz x64 Arm32 Arm64
Snap Snap Store
CLI x64 Arm32 Arm64
.zip Intel chip Apple silicon Universal
CLI Intel chip Apple silicon
By downloading and using Visual Studio Code, you agree to the license terms and privacy statement.

GCC

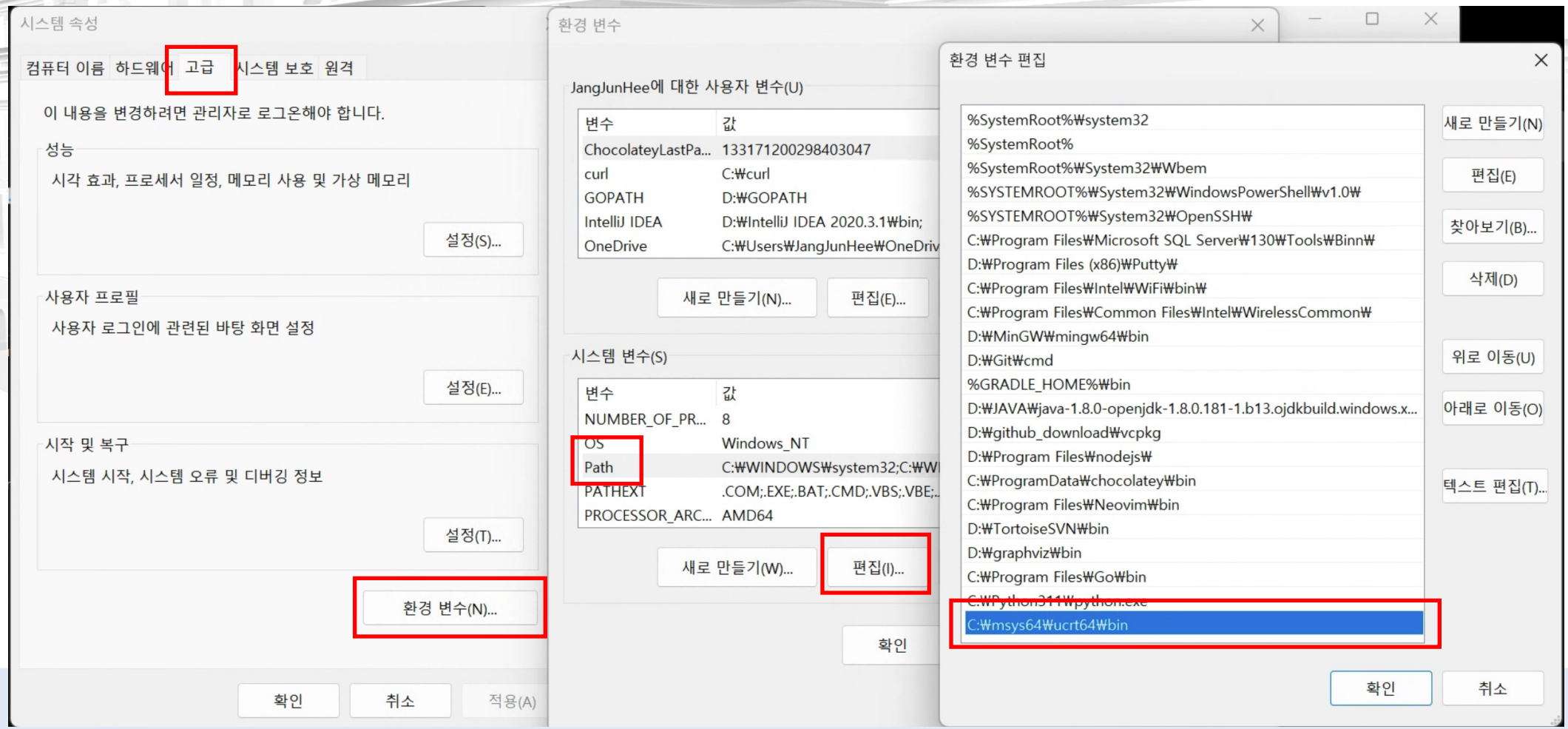
- GNU Compiler Colleciton
- C, C++, Objective C, Fortran, Ada, Go, D
- g++: C++용 컴파일러

GCC 설치

- Window OS
  - 링크의 Installing the MinGW-w64 toolchain의 1~7번
- MAC OS
  - 터미널에서 `xcode-select --install` 로 설치(오래 걸림)
  - Homebrew 사용(패키지 관리 어플리케이션)
    - `brew install gcc`

참고) 윈도우 환경변수 설정

- 돋보기 > 환경 변수 > 클릭

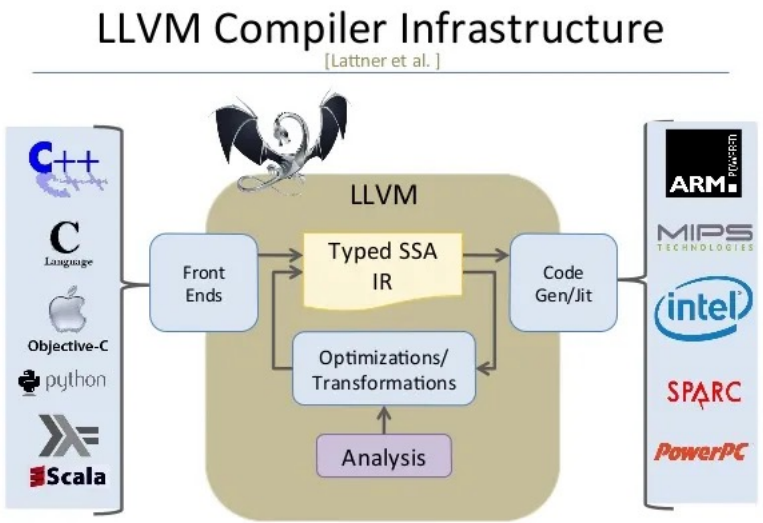


설치 확인

- `gcc --version`
- `g++ --version`
- `gdb --version`

llvm

- 프론트엔드-미들엔드-백엔드
- 예) C++ → 중간 언어 → 플랫폼 별 기계어
- 크로스플랫폼 컴파일러
  - GCC : 윈도우 정식 지원 X
  - VC++: 유닉스 계열 지원 X
- clang++ 사용



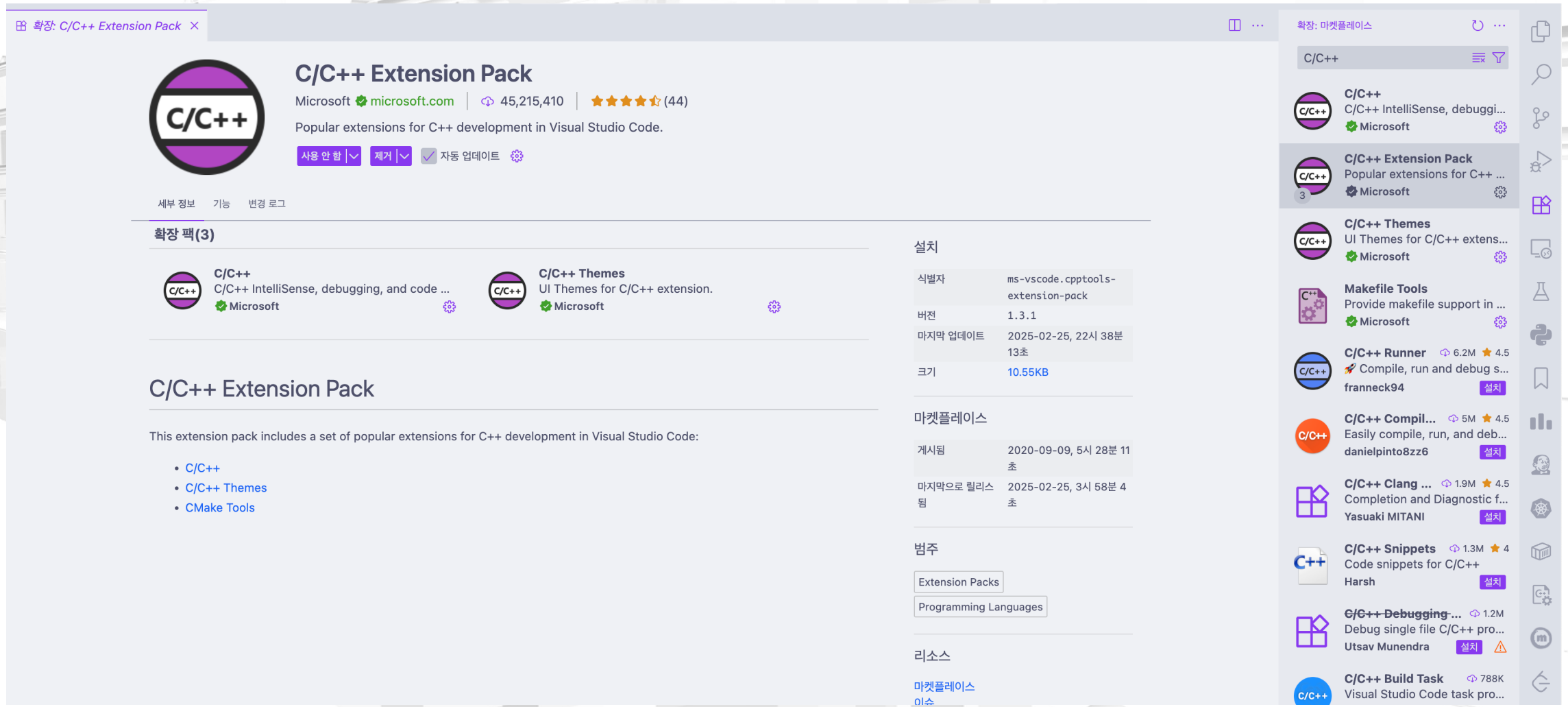
Clang 설치

- Window
  - gcc 설치 과정과 동일하게 msys2 ucrt terminal 실행
  - `pacman -S --needed mingw-w64-ucrt-x86_64-llvm mingw-w64-ucrt-x86_64-clang mingw-w64-ucrt-x86_64-lldb`
- Mac OS
  - 기본적으로 설치되어 있음.
  - `xcode-select --install`
  - `brew install llvm`

설치 확인

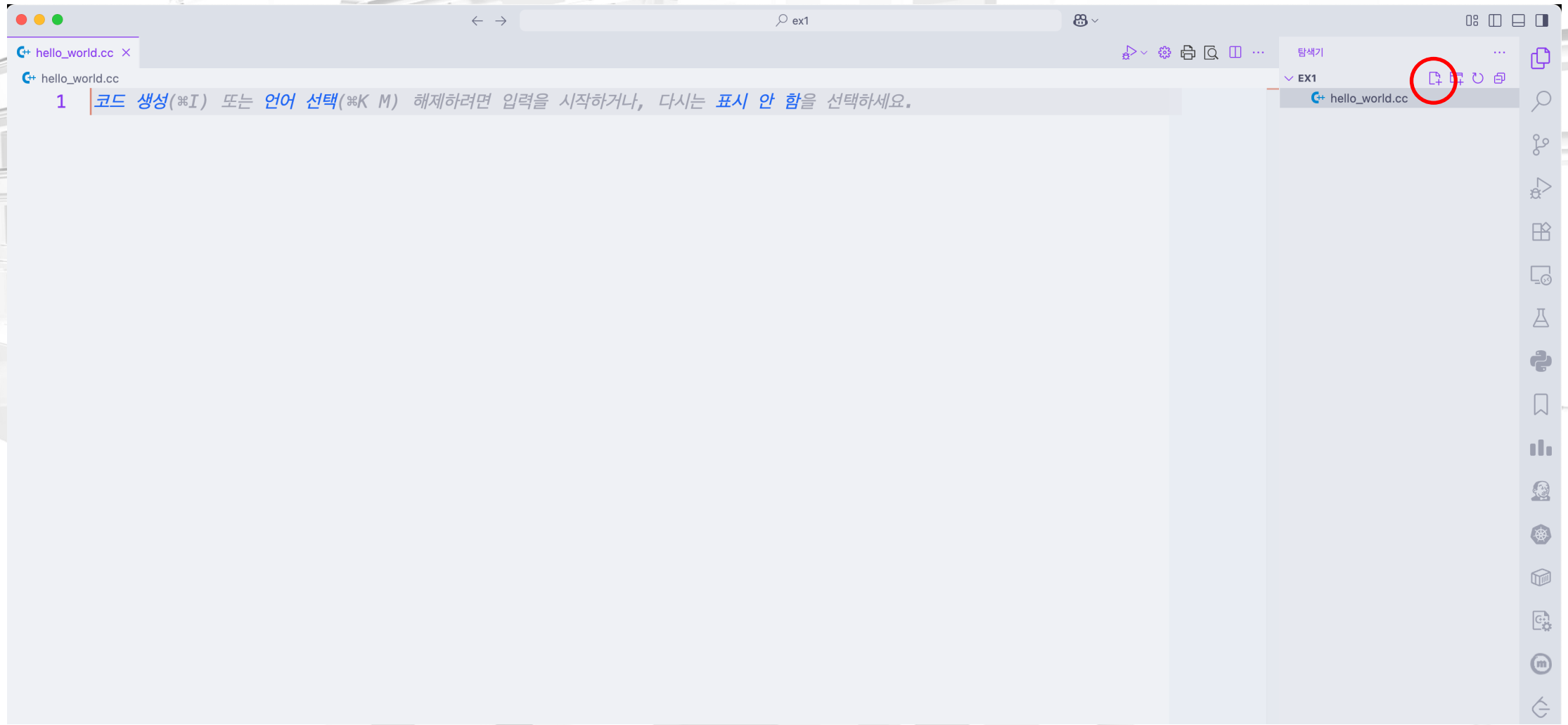
- `clang --version`
- `clang++ --version`
- `lldb --version`
- `...`

C/C++ 플러그인 설치

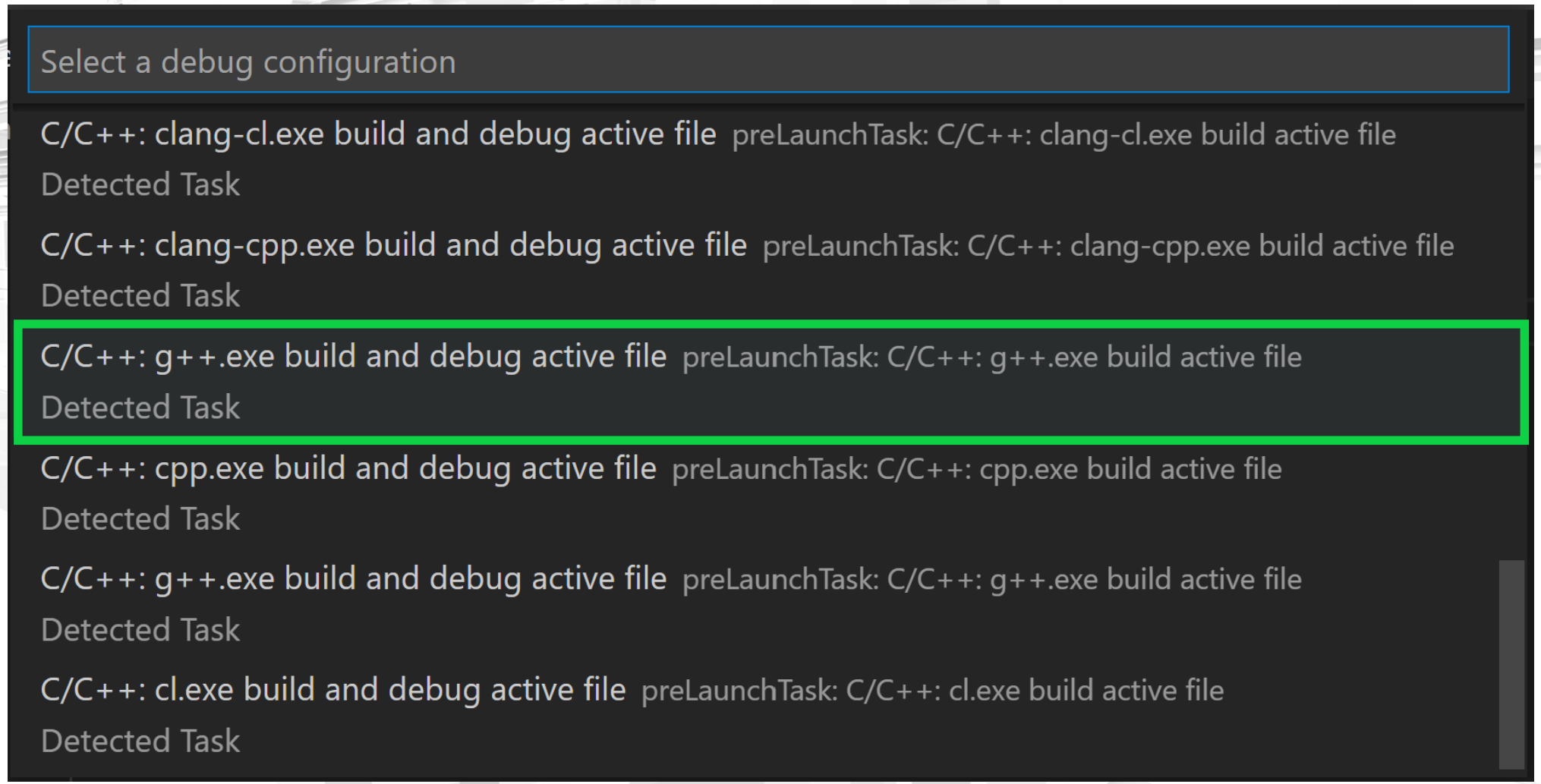

확장: C/C++ Extension Pack
C/C++ Extension Pack
Microsoft microsoft.com 45,215,410 (44)
Popular extensions for C++ development in Visual Studio Code.
사용 안 함
제거
자동 업데이트
세부 정보
기능
변경 로그
확장 팩(3)
C/C++
C/C++ IntelliSense, debugging, and code ...
Microsoft
C/C++ Themes
UI Themes for C/C++ extension.
Microsoft
C/C++ Extension Pack
This extension pack includes a set of popular extensions for C++ development in Visual Studio Code:
C/C++
C/C++ Themes
CMake Tools
설치
식별자 ms-vscode.cpptools-extension-pack
버전 1.3.1
마지막 업데이트 2025-02-25, 22시 38분 13초
크기 10.55KB
마켓플레이스
게시됨 2020-09-09, 5시 28분 11초
마지막으로 릴리스됨 2025-02-25, 3시 58분 4초
범주
Extension Packs
Programming Languages
리소스
마켓플레이스
확장: 마켓플레이스
C/C++
C/C++
C/C++ IntelliSense, debuggi...
Microsoft
C/C++ Extension Pack
Popular extensions for C++ ...
Microsoft
C/C++ Themes
UI Themes for C/C++ extens...
Microsoft
Makefile Tools
Provide makefile support in ...
Microsoft
C/C++ Runner 6.2M 4.5
Compile, run and debug s...
franneck94 설치
C/C++ Compil... 5M 4.5
Easily compile, run, and deb...
danielpinto8zz6 설치
C/C++ Clang ... 1.9M 4.5
Completion and Diagnostic f...
Yasuaki MITANI 설치
C/C++ Snippets 1.3M 4
Code snippets for C/C++
Harsh 설치
C/C++ Debugging ... 1.2M
Debug single file C/C++ pro...
Utsav Munendra 설치
C/C++ Build Task 788K
Visual Studio Code task pro...

VScode 프로젝트 실행

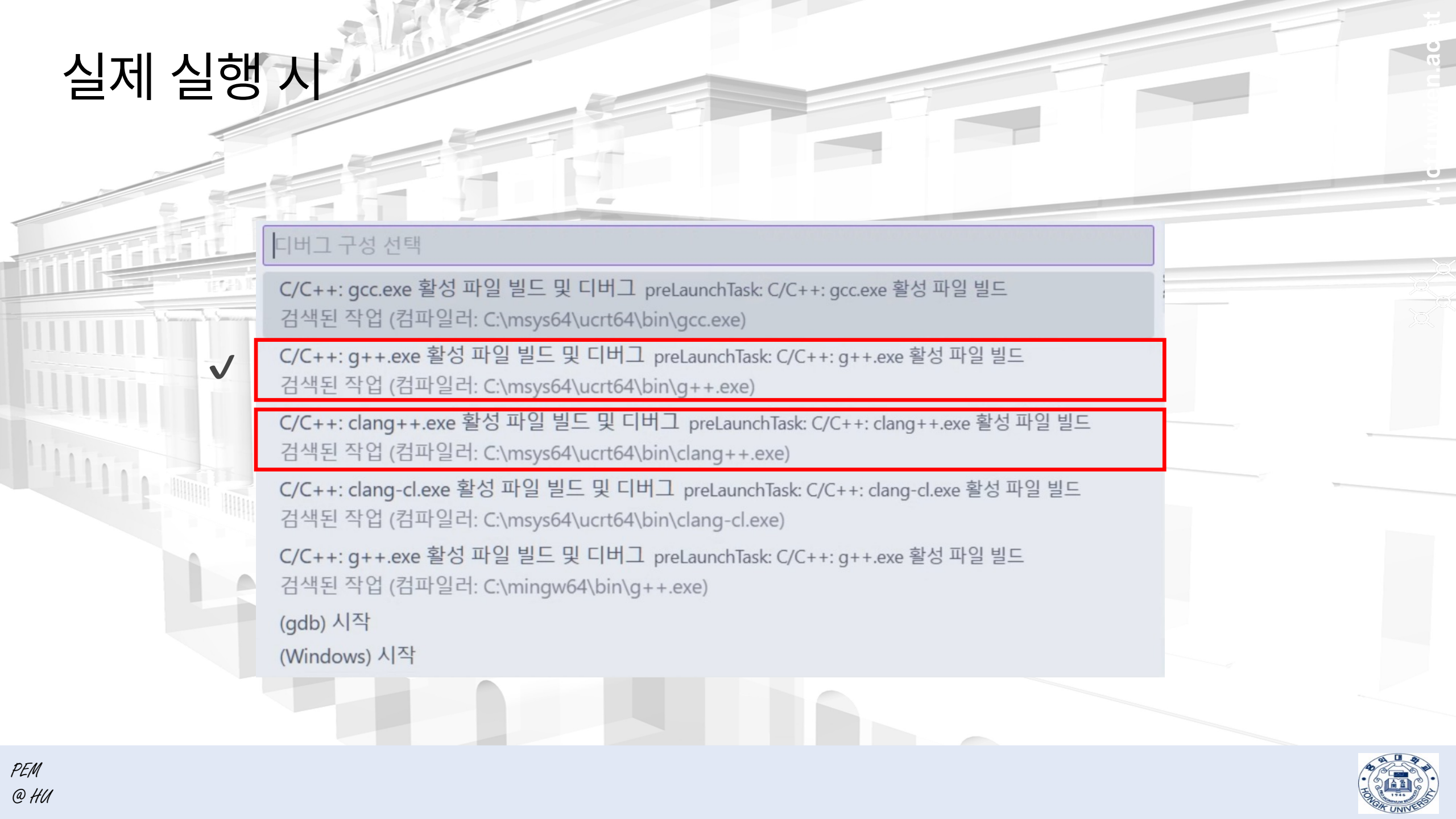
- VSCode 실행 > 열기 > 원하는 폴더


ex1
hello_world.cc
hello_world.cc
1 코드 생성(⌘I) 또는 언어 선택(⌘K M) 해제하려면 입력을 시작하거나, 다시는 표시 안 함을 선택하세요.
탐색기
EX1
hello_world.cc


EXPLORER
OPEN EDITORS
helloworld.cpp
HELLOWORLD
.vscode
helloworld.cpp
helloworld.cpp
helloworld.cpp > main()
Debug C/C++ File
Run C/C++ File
#include <iostream>
#include <vector>
#include <string>
using namespace std;
int main()
{
vector<string> msg{"Hello", "C++", "World", "from", "VS Code", "and the C++ ext
for (const string &word : msg)
{
cout << word << " ";
}
cout << endl;
}

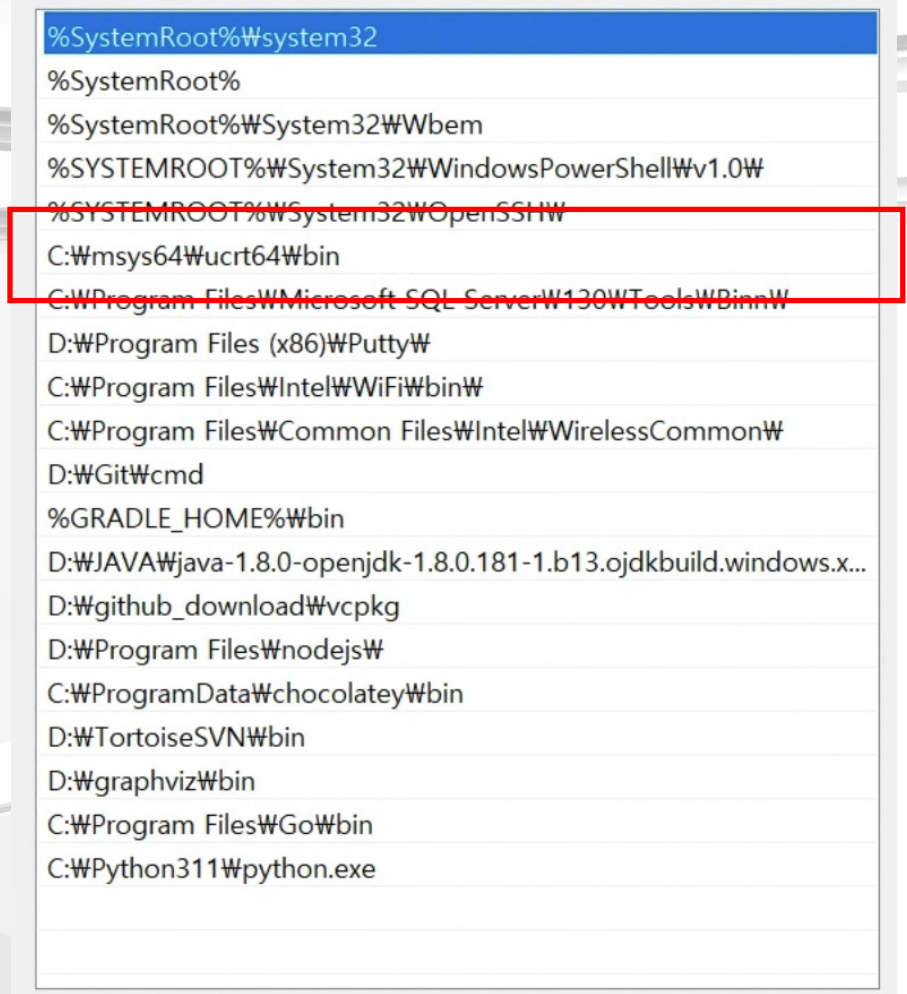

Select a debug configuration
C/C++: clang-cl.exe build and debug active file preLaunchTask: C/C++: clang-cl.exe build active file
Detected Task
C/C++: clang-cpp.exe build and debug active file preLaunchTask: C/C++: clang-cpp.exe build active file
Detected Task
C/C++: g++.exe build and debug active file preLaunchTask: C/C++: g++.exe build active file
Detected Task
C/C++: cpp.exe build and debug active file preLaunchTask: C/C++: cpp.exe build active file
Detected Task
C/C++: g++.exe build and debug active file preLaunchTask: C/C++: g++.exe build active file
Detected Task
C/C++: cl.exe build and debug active file preLaunchTask: C/C++: cl.exe build active file
Detected Task

실제 실행 시


디버그 구성 선택
C/C++: gcc.exe 활성 파일 빌드 및 디버그 preLaunchTask: C/C++: gcc.exe 활성 파일 빌드
검색된 작업 (컴파일러: C:\msys64\ucrt64\bin\gcc.exe)
C/C++: g++.exe 활성 파일 빌드 및 디버그 preLaunchTask: C/C++: g++.exe 활성 파일 빌드
검색된 작업 (컴파일러: C:\msys64\ucrt64\bin\g++.exe)
C/C++: clang++.exe 활성 파일 빌드 및 디버그 preLaunchTask: C/C++: clang++.exe 활성 파일 빌드
검색된 작업 (컴파일러: C:\msys64\ucrt64\bin\clang++.exe)
C/C++: clang-cl.exe 활성 파일 빌드 및 디버그 preLaunchTask: C/C++: clang-cl.exe 활성 파일 빌드
검색된 작업 (컴파일러: C:\msys64\ucrt64\bin\clang-cl.exe)
C/C++: g++.exe 활성 파일 빌드 및 디버그 preLaunchTask: C/C++: g++.exe 활성 파일 빌드
검색된 작업 (컴파일러: C:\mingw64\bin\g++.exe)
(gdb) 시작
(Windows) 시작

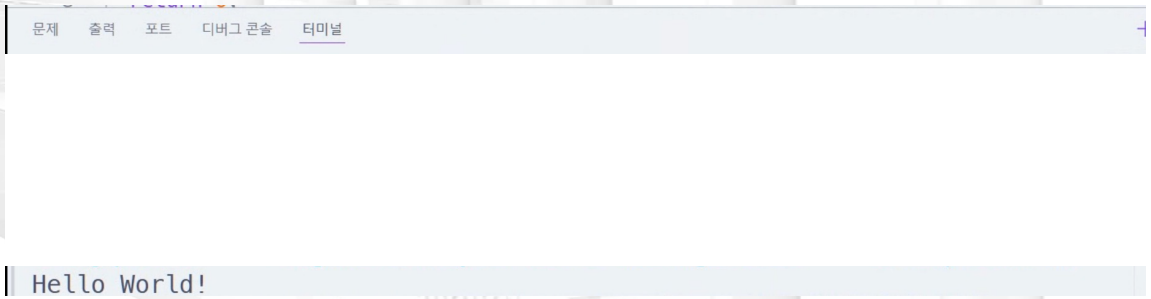
Hello World 실행 시 Seg Fault

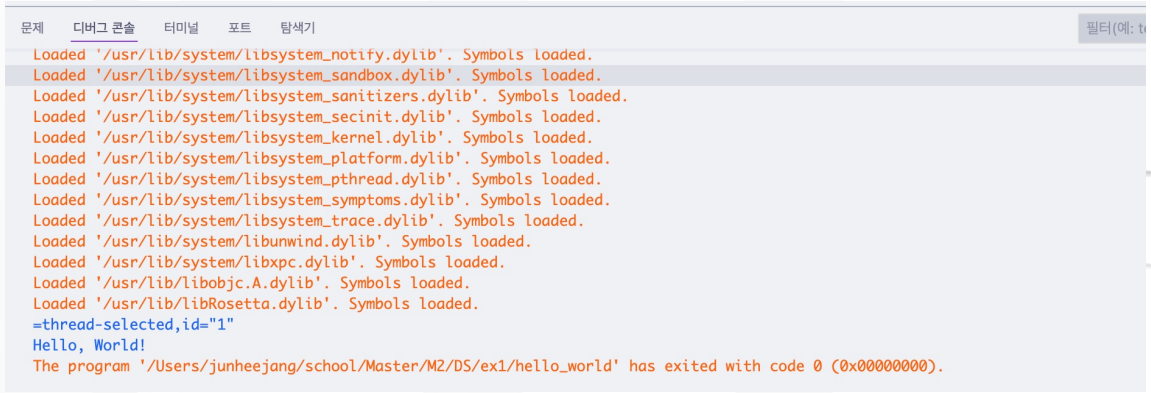
- Windows)
  - 기존에 mingw 등 gnu 관련 툴을 설치했을 경우 환경변수 우선순위에서 밀리면 다른 dll 을 가져와서 문제가 발생할 수 있음
  - 우선순위를 올려주면 해결가능



실행 완료

터미널 또는 디버그 콘솔에서 출력값을 볼 수 있음





tasks.json과 launch.json

```jsonc
{
    "tasks": [
        {
            "type": "cppbuild",
            "label": "C/C++: g++ 활성 파일 빌드",
            "command": "/usr/bin/g++",
            "args": [
                "-fdiagnostics-color=always",
                "-g",
                "${file}",
                "-o",
                "${fileDirname}/${fileBasenameNoExtension}"
            ],
            "options": {
                "cwd": "${fileDirname}"
            },
            "problemMatcher": [
                "$gcc"
            ],
            "group": {
                "kind": "build",
                "isDefault": true
            },
            "detail": "디버거에서 생성된 작업입니다."
        }
    ],
    "version": "2.0.0"
}
```

```jsonc
"configurations": [
        {
            "name": "C++ Launch",
            "type": "cppdbg",
            "request": "launch",
            "program": "${workspaceFolder}/a.out",
            "stopAtEntry": false,
            "customLaunchSetupCommands": [
                {
                    "text": "target-run",
                    "description": "run target",
                    "ignoreFailures": false
                }
            ],
            "launchCompleteCommand": "exec-run",
            "linux": {
                "MIMode": "gdb",
                "miDebuggerPath": "/usr/bin/gdb"
            },
            "osx": {
                "MIMode": "lldb"
            },
            "windows": {
                "MIMode": "gdb",
                "miDebuggerPath": "C:\\MinGw\\bin\\gdb.exe"
            }
        }
    ]
```

tasks.json

- 반복적으로 수행되는 작업(빌드 등)을 자동화하는 데 사용
- 터미널에서 g++를 이용한 컴파일
  - `g++ helloworld.cc -o helloworld`
    - `g++ -c helloworld.cc` → `helloworld.o`
    - `g++ helloworld.o -o helloworld`
- tasks.json 파일에 여러 파일을 동시에 빌드하게 하거나, make 명령 등을 사용하는 것 역시 가능

launch.json

- 디버깅 기능을 활용하기 위한 설정 파일
  - lldb(clang++)를 쓸 때는 CodeLLDB 확장 설치 시 더욱 편하게 사용가능
- 자세한 것은 링크 참고
- C++와 관련된 실습에서 같이 실습하면서 설명 예정