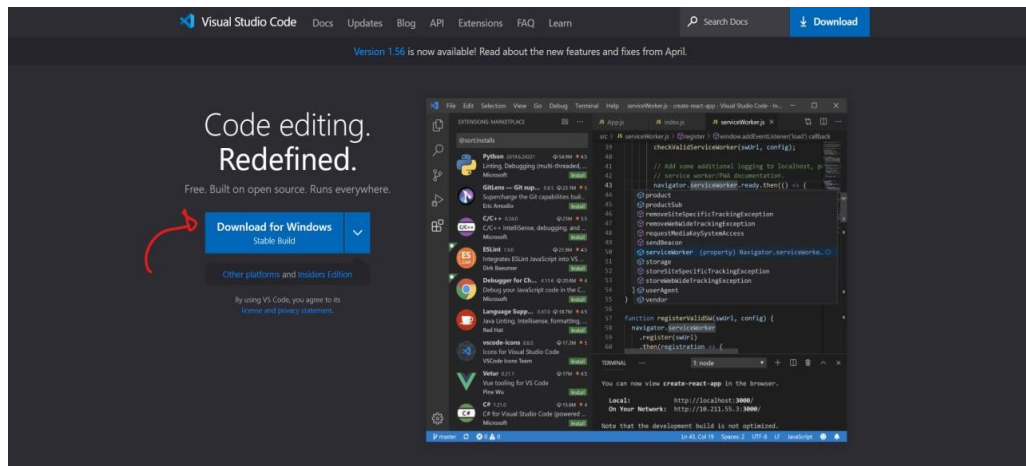


# C 프로그래밍 환경 설정



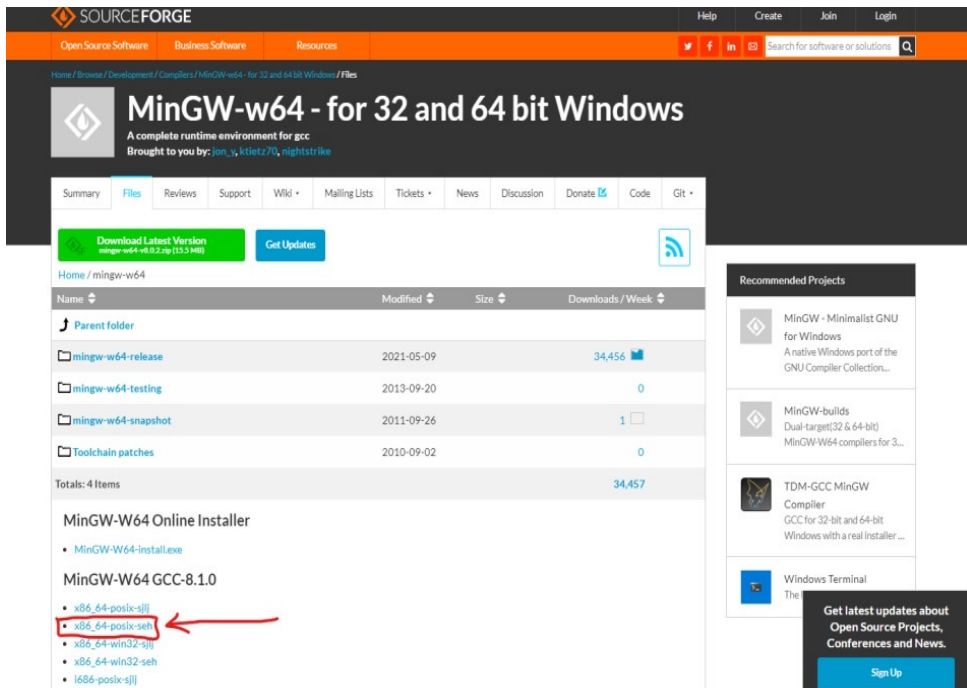
# VISUAL STUDIO CODE 설치

- Visual Studio Code는 코드 작성 프로그램입니다.
- 다음 사이트 접속:  
<https://code.visualstudio.com/>
- 아래 버튼으로 각자 운영체제에 맞는 설치파일을 다운 받고 설치
  - 다 next 누르면 됩니다.



# GCC 설치 (WINDOWS)

- **Windows** 운영체제에만 해당합니다.
- Mingw는 Windows에서 gcc를 사용하게 해 줍니다.
- 다음 사이트 접속:  
<https://sourceforge.net/projects/mingw-w64/files/mingw-w64/>
- x86\_64-posix-seh 을 클릭하면 압축 파일이 받아집니다.



The screenshot shows the SourceForge project page for MinGW-w64. The page title is "MinGW-w64 - for 32 and 64 bit Windows". Below the title, there are tabs for Summary, Files, Reviews, Support, Wiki, Mailing Lists, Tickets, News, Discussion, Donate, Code, and Git. The "Files" tab is selected, showing a list of files. The list includes "mingw-w64-release", "mingw-w64-testing", "mingw-w64-snapshot", and "Toolchain patches". Below this, there is a section for "MinGW-W64 GCC-8.1.0" which lists several files, including "x86\_64-posix-seh". A red arrow points to the "x86\_64-posix-seh" file.

# GCC 설치 (WINDOWS)

- 다운 받은 파일을 압축 해제하면 mingw64 폴더가 있을 것입니다. 해당 폴더를 로컬 디스크(C:)에 위치시킵니다.

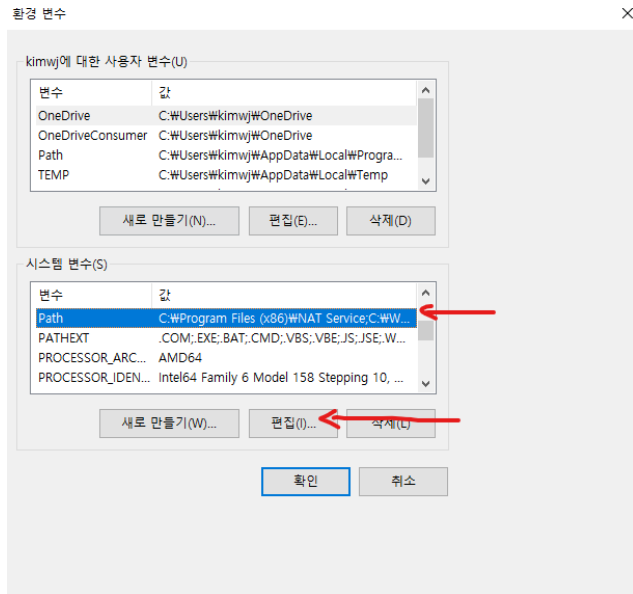
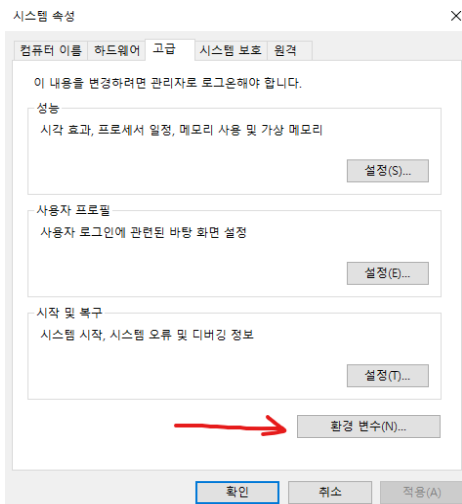


- 윈도우 실행 창에 sysdm.cpl을 입력하고 엔터를 치면 '시스템 속성' 창이 뜹니다.



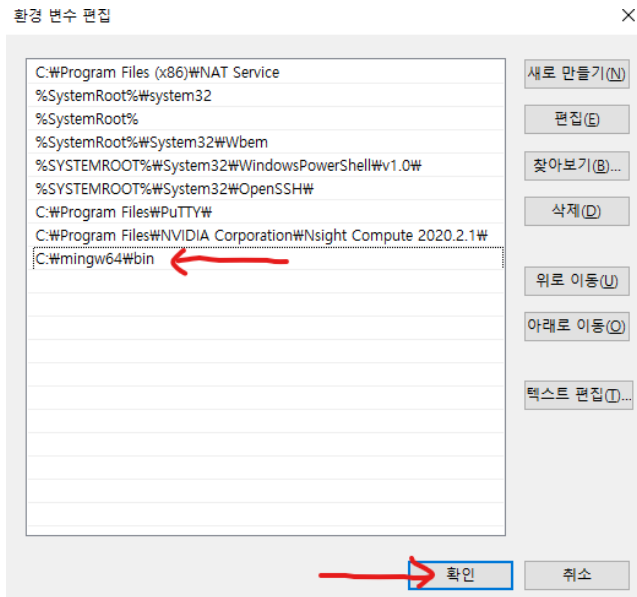
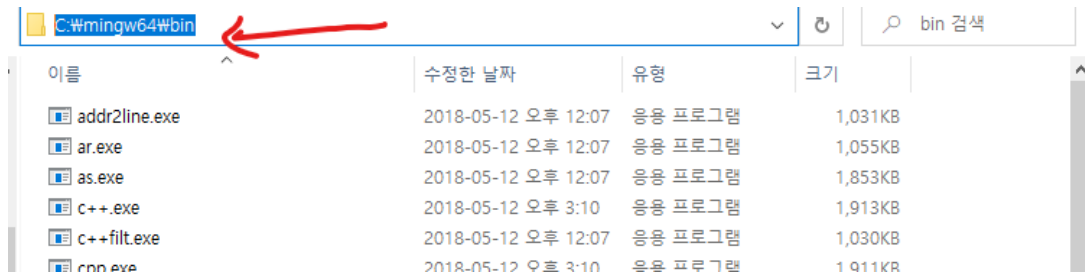
# GCC 설치 (WINDOWS)

- '고급' 탭의 '환경 변수' 클릭
- '시스템 변수' 칸에서 'Path'를 선택 후 '편집' 클릭



# GCC 설치 (WINDOWS)

- 새로 만들기를 누르고 아까 C드라이브에 위치 시킨 mingw64 폴더의 bin 폴더의 위치를 입력합니다.



- 그 다음 열렸던 창을 다 확인을 누릅니다.



# GCC 설치 (WINDOWS)

- 윈도우 검색 창에 cmd를 입력해서 커맨드 창을 띄운 뒤 gcc -v 를 입력했을 때, 다음과 같이 설치가 된 것을 확인할 수 있습니다.

```
명령 프롬프트
Microsoft Windows [Version 10.0.19041.928]
(c) Microsoft Corporation. All rights reserved.

C:\Users\kimwj>gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=C:/mingw64/bin/./libexec/gcc/x86_64-w64-mingw32/8.1.0/lto-wrapper.exe
Target: x86_64-w64-mingw32
Configured with: ../../src/gcc-8.1.0/configure --host=x86_64-w64-mingw32 --build=x86_64-w64-mingw32 --target=x86_64-w64-mingw32 --prefix=/mingw64 --with-sysroot=/c/mingw810/x86_64-810-posix-seh-rt_v6-rev0/mingw64 --enable-shared --enable-static --disable-multilib --enable-languages=c,c++,fortran,lto --enable-libstdcxx-time=yes --enable-threads=posix --enable-libgomp --enable-libatomic --enable-lto --enable-graphite --enable-checking=release --enable-fully-dynamic-string --enable-version-specific-runtime-libs --disable-libstdcxx-pch --disable-libstdcxx-debug --enable-bootstrap --disable-rpath --disable-win32-registry --disable-nls --disable-werror --disable-symvers --with-gnu-as --with-gnu-ld --with-arch=nocona --with-tune=core2 --with-libiconv --with-system-zlib --with-gmp=/c/mingw810/prerequisites/x86_64-w64-mingw32-static --with-mpfr=/c/mingw810/prerequisites/x86_64-w64-mingw32-static --with-mpc=/c/mingw810/prerequisites/x86_64-w64-mingw32-static --with-isl=/c/mingw810/prerequisites/x86_64-w64-mingw32-static --with-pkgversion='x86_64-posix-seh-rev0, Built by MinGW-W64 project' --with-bugurl=https://sourceforge.net/projects/mingw-w64 CFLAGS='-O2 -pipe -fno-ident -I/c/mingw810/x86_64-810-posix-seh-rt_v6-rev0/mingw64/opt/include -I/c/mingw810/prerequisites/x86_64-zlib-static/include -I/c/mingw810/prerequisites/x86_64-w64-mingw32-static/include' CXXFLAGS='-O2 -pipe -fno-ident -I/c/mingw810/x86_64-810-posix-seh-rt_v6-rev0/mingw64/opt/include -I/c/mingw810/prerequisites/x86_64-zlib-static/include -I/c/mingw810/prerequisites/x86_64-w64-mingw32-static/include' CPPFLAGS='-I/c/mingw810/x86_64-810-posix-seh-rt_v6-rev0/mingw64/opt/include -I/c/mingw810/prerequisites/x86_64-zlib-static/include -I/c/mingw810/prerequisites/x86_64-w64-mingw32-static/include' LDFLAGS='-pipe -fno-ident -L/c/mingw810/x86_64-810-posix-seh-rt_v6-rev0/mingw64/opt/lib -L/c/mingw810/prerequisites/x86_64-zlib-static/lib -L/c/mingw810/prerequisites/x86_64-w64-mingw32-static/lib'
Thread model: posix
gcc version 8.1.0 (x86_64-posix-seh-rev0, Built by MinGW-W64 project)
```

# GCC 설치 (MACOS)

- **MacOS인 경우**, terminal을 열고 terminal에 다음과 같이 명령어를 입력합니다.

```
xcode-select --install
```

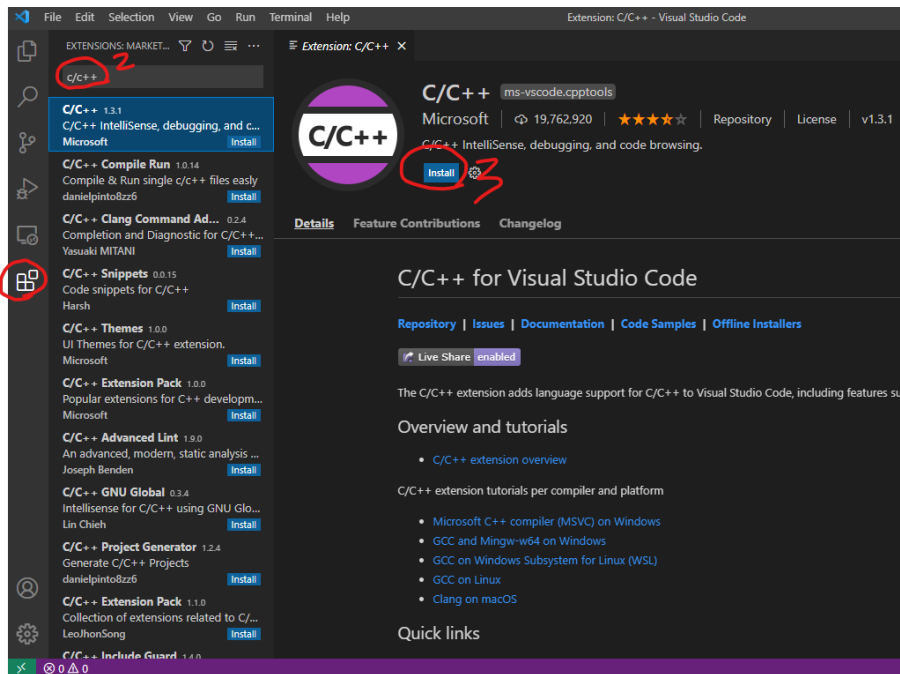
- 설치 창이 뜨면 install 버튼으로 설치를 진행합니다.
- 설치가 끝나면 terminal에 다음과 같이 입력했을 때, 오류가 발생하지 않아야 합니다.

```
gcc -v
```

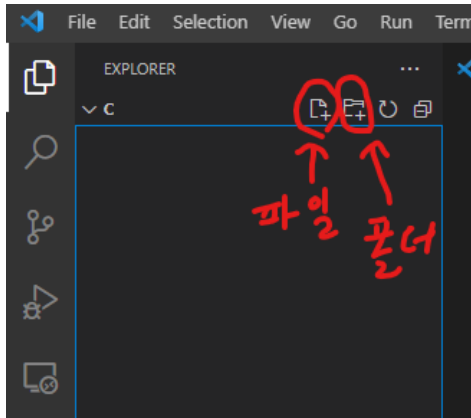


# C 코드 작성

- Visual Studio Code를 실행하고, 다음과 같이 'C/C++' extension을 설치합니다. 오른쪽 아래에 설치 중이라는 알림이 뜨고, 설치가 다 되면 Visual Studio Code를 껏다 겁니다.



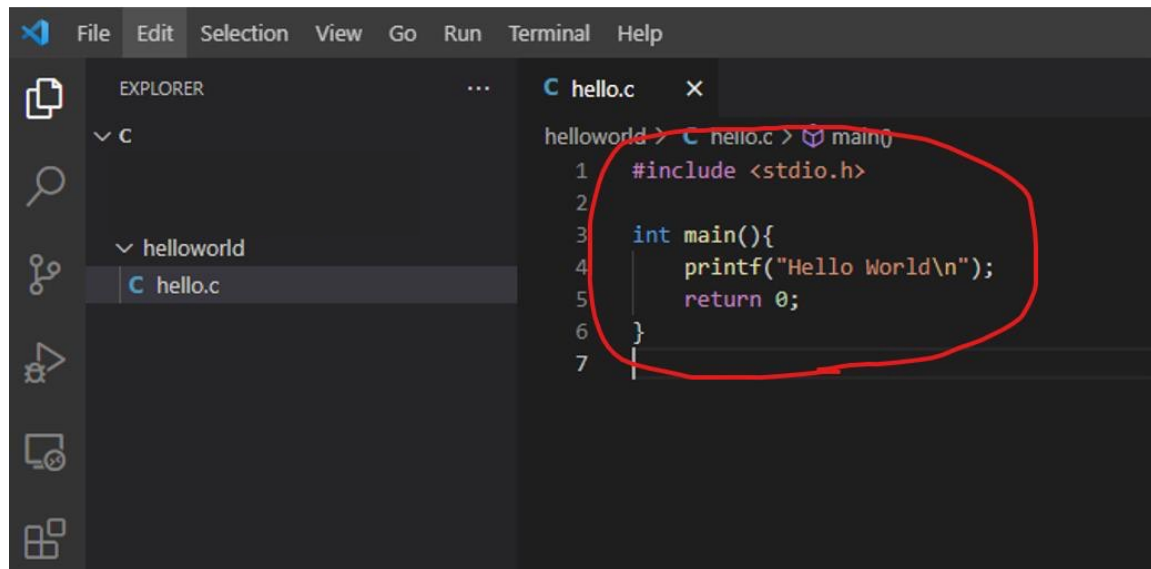
# C 코드 작성



- 왼쪽의 Open Folder를 눌러서 작업을 진행할 폴더를 선택합니다.
- 작업할 위치에서 다음과 같이 파일이나 폴더를 추가할 수 있습니다. 저는 helloworld라는 폴더를 만들었습니다.

# C 코드 작성

- 이제 파일 추가 버튼을 누른 다음 hello.c 라는 파일을 만들고 다음과 같이 입력을 하고 ctrl+s로 저장을 합니다.

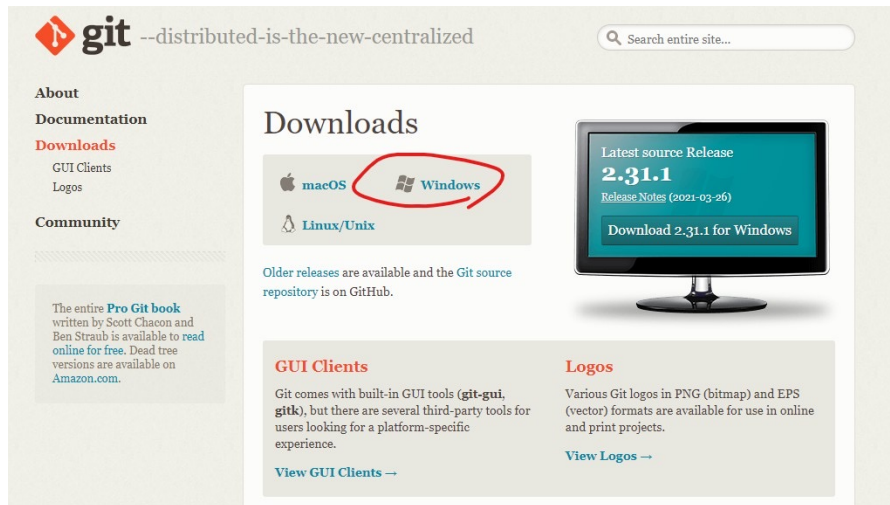


The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar shows a project structure with a folder named 'C' containing a subfolder 'helloworld' and a file 'hello.c'. The main editor window displays the contents of 'hello.c', which is circled in red. The code is as follows:

```
helloworld > C hello.c > main()
1  #include <stdio.h>
2
3  int main(){
4      printf("Hello World\n");
5      return 0;
6  }
7  |
```

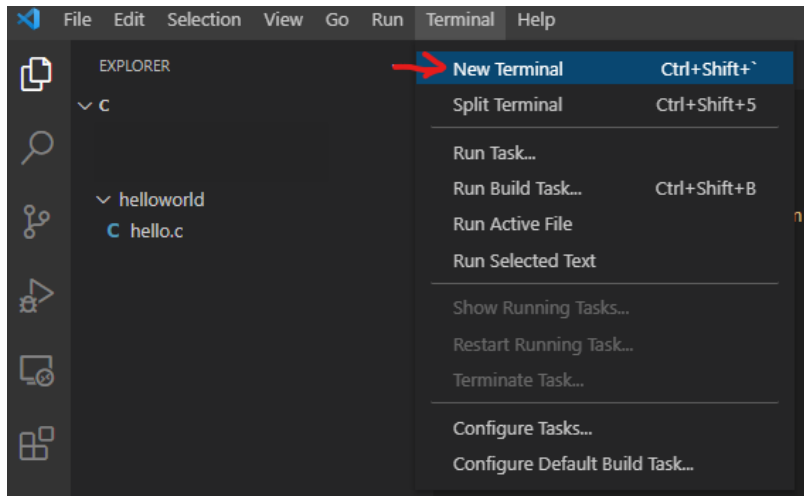
# GIT BASH 설치

- **Windows 운영체제인 경우**, Visual Studio Code에서 linux terminal을 사용하기 위해 git bash를 설치합니다. **MacOS인 경우** 굳이 설치할 필요 없습니다.
- 다음 사이트에서 git bash를 설치합니다.  
<https://git-scm.com/downloads>
- 아래 버튼을 눌러서 실행파일을 받고 다 next를 누르면 됩니다.



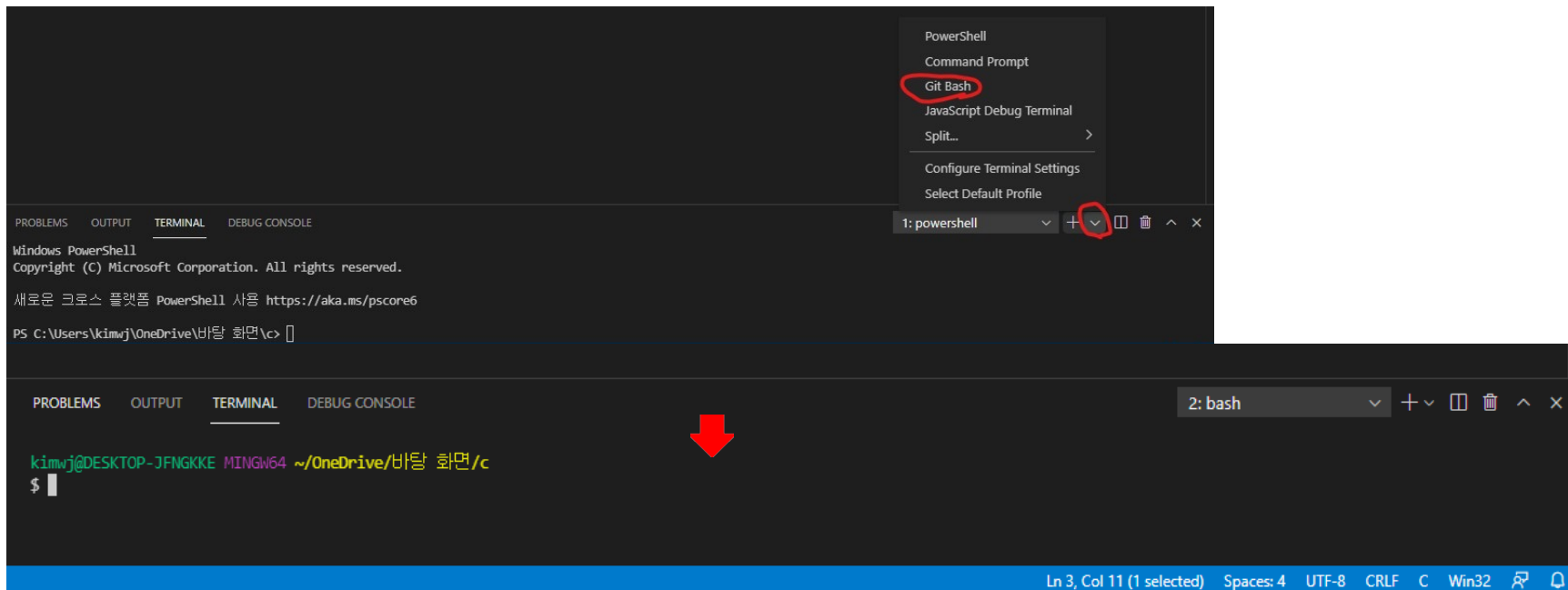
# C 코드 컴파일

- 다시 Visual Studio Code로 돌아와서, Terminal → New Terminal을 클릭하면 아래에 terminal이 뜹니다.



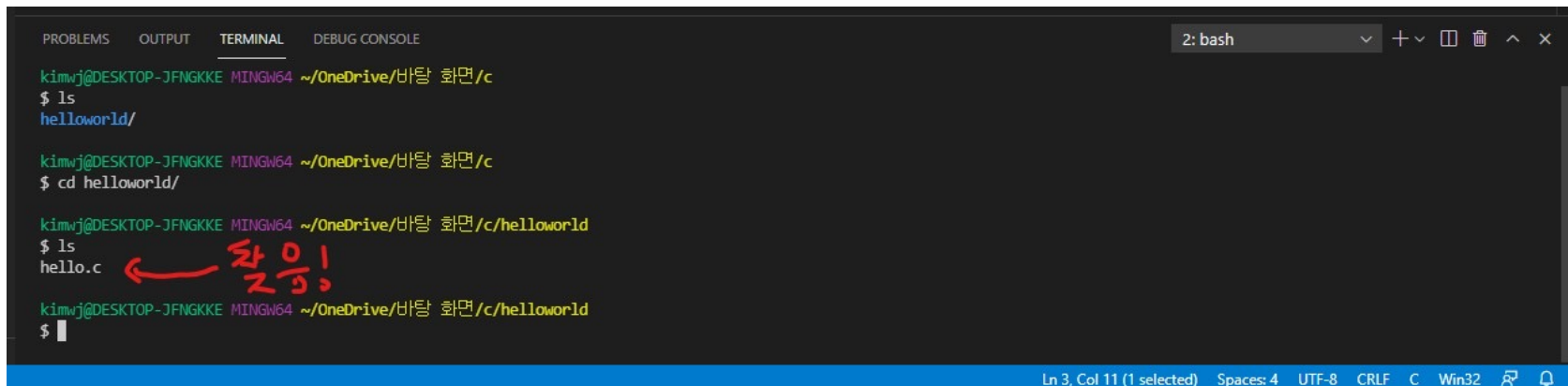
# C 코드 컴파일

- Window의 경우, 다음과 같이 아까 설치한 Git Bash를 클릭하면 terminal이 바뀝니다.



# C 코드 컴파일

- 이제 terminal에서 아까 작성한 hello.c 파일의 위치로 가야 합니다. 다음 명령어를 이용합니다.
  - ls → 현재 directory에 있는 파일, 폴더 목록이 뜹니다.
  - cd 폴더이름 → 해당 폴더로 이동
  - 파일 또는 폴더 이름을 적을 때, tab 버튼을 누르면 자동 완성이 됩니다. (prefix가 유일한 경우)



```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE  2: bash
kimwj@DESKTOP-JFNGKKE MINGW64 ~/OneDrive/바탕 화면/c
$ ls
helloworld/

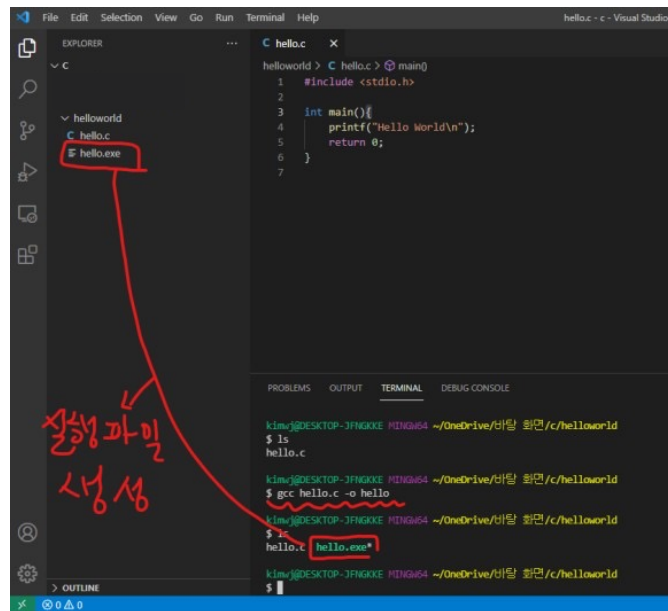
kimwj@DESKTOP-JFNGKKE MINGW64 ~/OneDrive/바탕 화면/c
$ cd helloworld/

kimwj@DESKTOP-JFNGKKE MINGW64 ~/OneDrive/바탕 화면/c/helloworld
$ ls
hello.c
kimwj@DESKTOP-JFNGKKE MINGW64 ~/OneDrive/바탕 화면/c/helloworld
$
```

Handwritten red text: "좌우!" with an arrow pointing to "hello.c".

# C 코드 컴파일

- hello.c 를 찾았으면, 이제 컴파일을 합니다. 다음과 같이 terminal에 입력합니다. gcc hello.c -o hello  
(hello.c 파일을 컴파일해서 hello라는 실행파일을 만들겠다는 의미)
- 그러면 hello라는 실행파일이 생깁니다. (MacOS의 경우 .exe가 없이 그냥 hello일 것입니다.)



Visual Studio Code interface showing the compilation of a C program. The Explorer pane on the left shows the file structure with 'hello.exe' highlighted. The Editor pane shows the source code of 'hello.c'. The Terminal pane at the bottom shows the command 'gcc hello.c -o hello' being executed, resulting in the creation of 'hello.exe'.

```

File Edit Selection View Go Run Terminal Help
hello.c - c - Visual Studio Code

EXPLORER
C
  hello.c
  hello.exe

C hello.c
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello World\n");
5     return 0;
6 }
7

TERMINAL
kimg@DESKTOP-3FNGKKE HING64 ~/OneDrive/바탕 화면/c/helloworld
$ ls
hello.c
kimg@DESKTOP-3FNGKKE HING64 ~/OneDrive/바탕 화면/c/helloworld
$ gcc hello.c -o hello
kimg@DESKTOP-3FNGKKE HING64 ~/OneDrive/바탕 화면/c/helloworld
$ ls
hello.c hello.exe
  
```

실행파일 생성



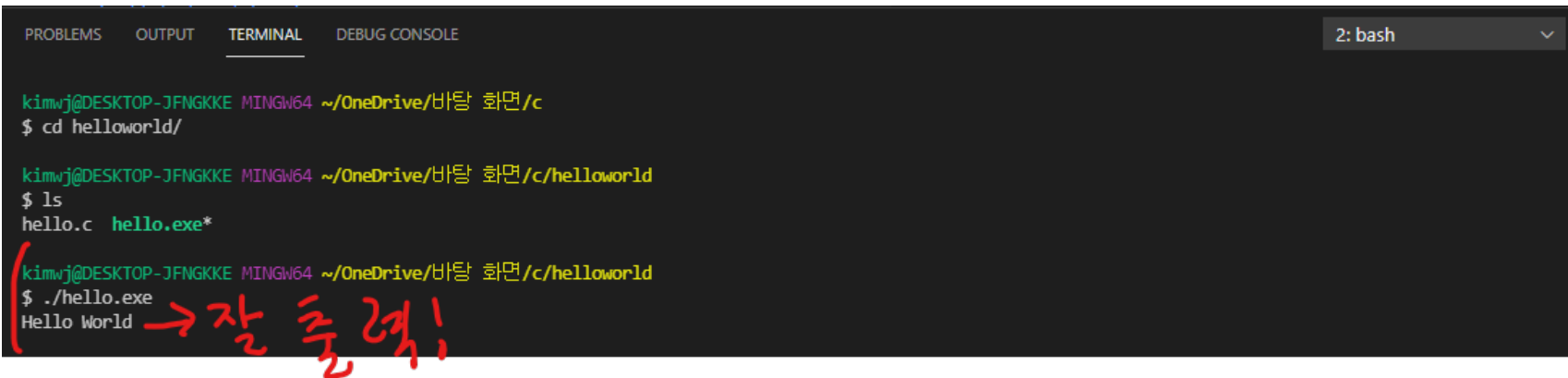
# C 코드 실행

- 다음과 같이 입력해서 만들어진 실행 파일을 실행합니다.

`./hello.exe` ( MacOS의 경우 `./hello` )

(./ 는 것은 현재 directory 를 의미합니다)

- Hello World가 출력되면 성공입니다.
- 잘 안 되면 구글링 및 etl 질문 게시판 이용해 주세요.



```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 2: bash v
kimwj@DESKTOP-JFNGKKE MINGW64 ~/OneDrive/바탕 화면/c
$ cd helloworld/

kimwj@DESKTOP-JFNGKKE MINGW64 ~/OneDrive/바탕 화면/c/helloworld
$ ls
hello.c  hello.exe*

kimwj@DESKTOP-JFNGKKE MINGW64 ~/OneDrive/바탕 화면/c/helloworld
$ ./hello.exe
Hello World → 잘 풀렸!
```