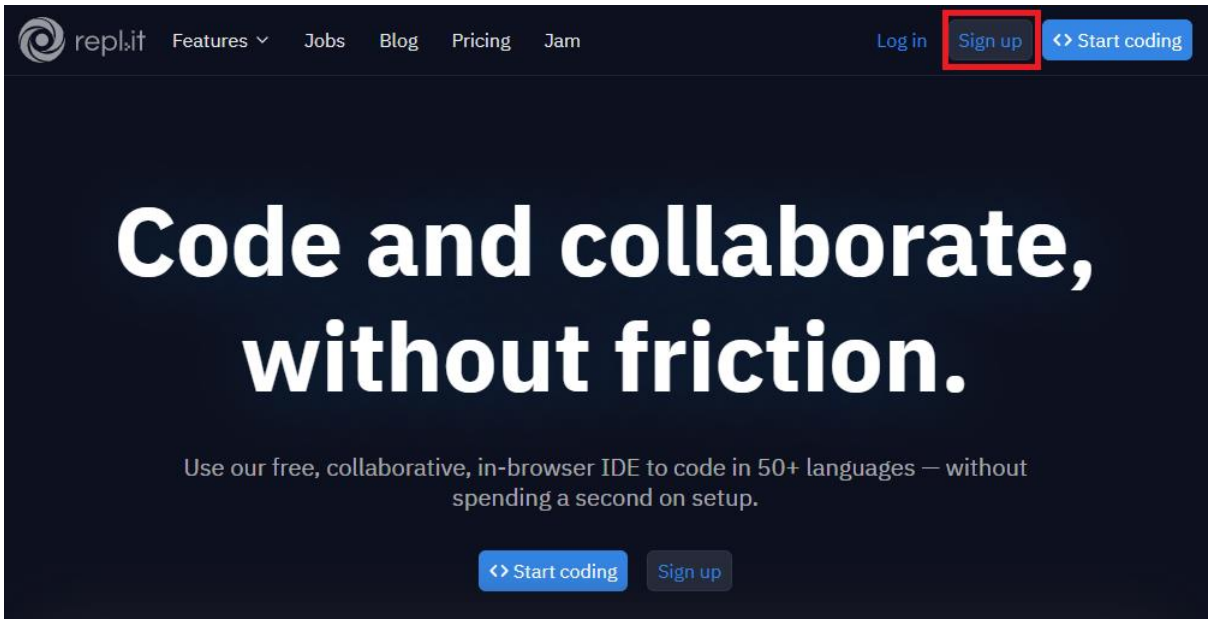
실습 환경 구축

1 온라인에서 실습

C++과 같은 고전적인 프로그래밍 언어는 환경을 설정하고 사용하는 것이 조금 복잡합니다. 초보자 단계에서는 환경을 직접 설정하고 사용하는 것이 힘들 수 있습니다. 따라서 처음 C++을 공부한다면 온라인 IDE를 활용하는 것을 추천합니다. 온라인 IDE로 C++과 어느 정도 익숙해졌다면 '비주얼 스튜디오 설치하기'를 참고해주세요.

repl.it 가입

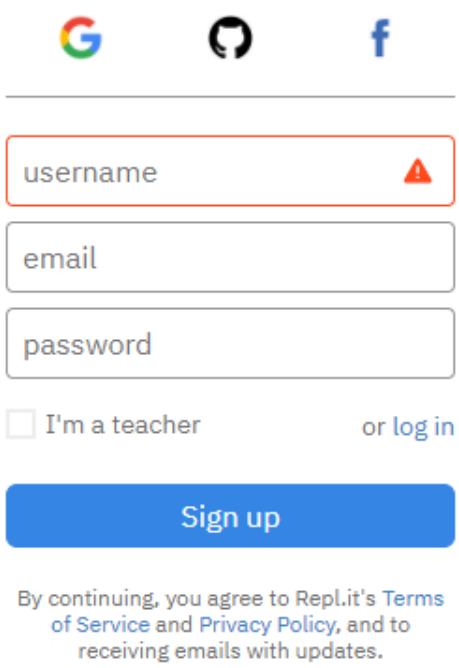
온라인 IDE는 온라인에서 사용할 수 있는 통합 개발 환경입니다. 다양한 온라인 IDE가 있는데, 이 책에서는 무료로 간단하게 활용해볼 수 있는 repl.it을 사용해보겠습니다. https://repl.it에 접속합니다.



repl.it 메인 페이지

repl.it은 가입하지 않아도 간단하게 코드를 실행해볼 수 있지만, 파일을 저장하고 불러오려면 가입해서 사용해야 합니다. repl.it 메인 페이지의 오른쪽 위에 있는 [sign up]을 눌러서 가입을 진행합니다. 가입할 때는 username(사용자 이름), email(이메일), password(비밀번호)만 간단하게 입력하

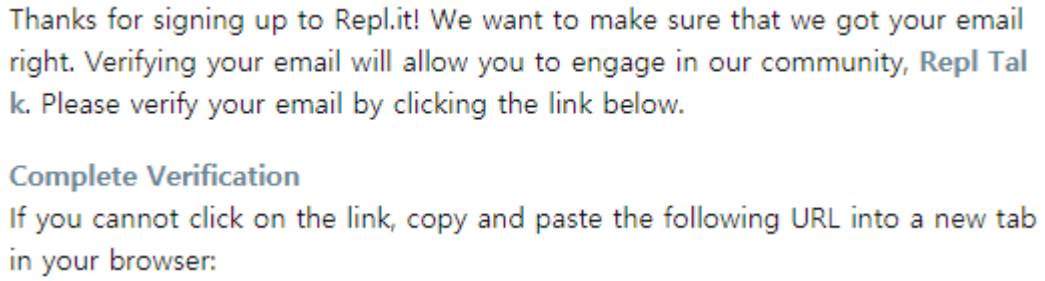
면 됩니다. 사용자 이름은 알파벳과 숫자로만 입력해주세요.



회원 가입 페이지

역자 가입 시, 사람이 가입하는 것이 맞는지 확인하기 위해 간단한 이미지 선택 문제 등이 나올 수 있습니다. 이후에 프로젝트 생성을 할 때도 사람이 맞는지 확인하기 위한 문제가 나올 수 있으니 지시에 따라 문제를 풀어주세요.

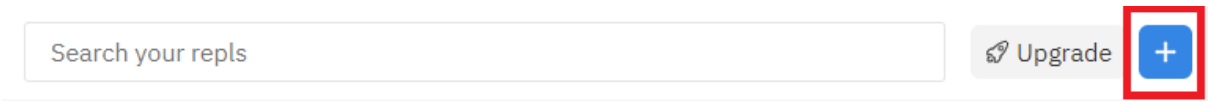
모두 입력하고 [sign up] 버튼을 누르면, 이메일로 다음과 같은 메일이 옵니다. 'Complete Verification' 아래의 링크를 클릭하거나, 주소창에 복사해서 넣으면 계정 인증이 완료됩니다.



계정 인증 메일

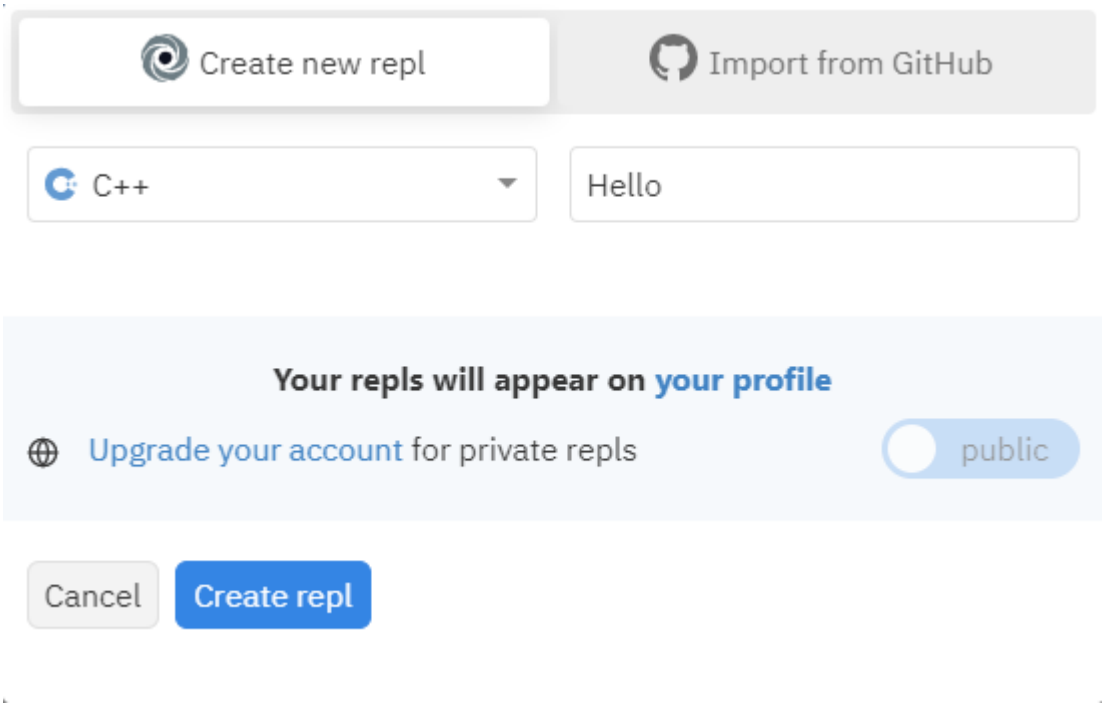
repl.it 프로젝트 생성

계정 인증까지 완료한 뒤 repl.it 사이트에 접속하면, 다음과 같이 사용자 페이지가 나옵니다. 오른쪽 위에 있는 [+] 버튼을 누르면 프로젝트를 만들 수 있습니다.



로그인 후 첫 페이지

버튼을 누르면 다음과 같이 프로젝트의 종류와 이름을 입력할 수 있는 대화 상자가 나옵니다. [C++]을 선택하고, 프로젝트 이름은 'Hello'로 간단하게 입력하고, [Create Repl] 버튼을 누르면 프로젝트가 생성합니다.



프로젝트 생성 대화 상자

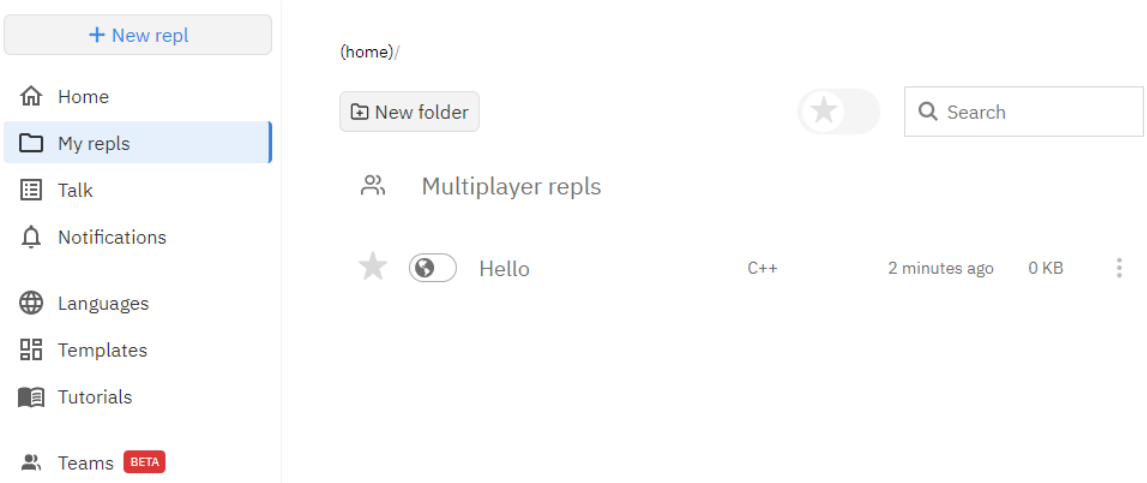
역자무료 계정으로는 공개 프로젝트만 만들 수 있습니다. 공개 프로젝트는 프로젝트의 내용이 공개되는 프로젝트입니다. 비공개 프로젝트를 만들고 싶다면 매달 일정 금액을 지불하고 유료 계정

으로 전환해야 합니다. 일단 학습 중이므로 무료 계정으로 진행합시다. C++과 어느 정도 익숙해지면, '비주얼 스튜디오 설치하기'를 참고해서 자신의 컴퓨터에 C++ 개발 환경을 구축해서 사용해보세요.

프로젝트가 만들어지면 다음과 같이 코드를 입력하고 실행할 수 있는 환경이 열립니다.

생성된 프로젝트

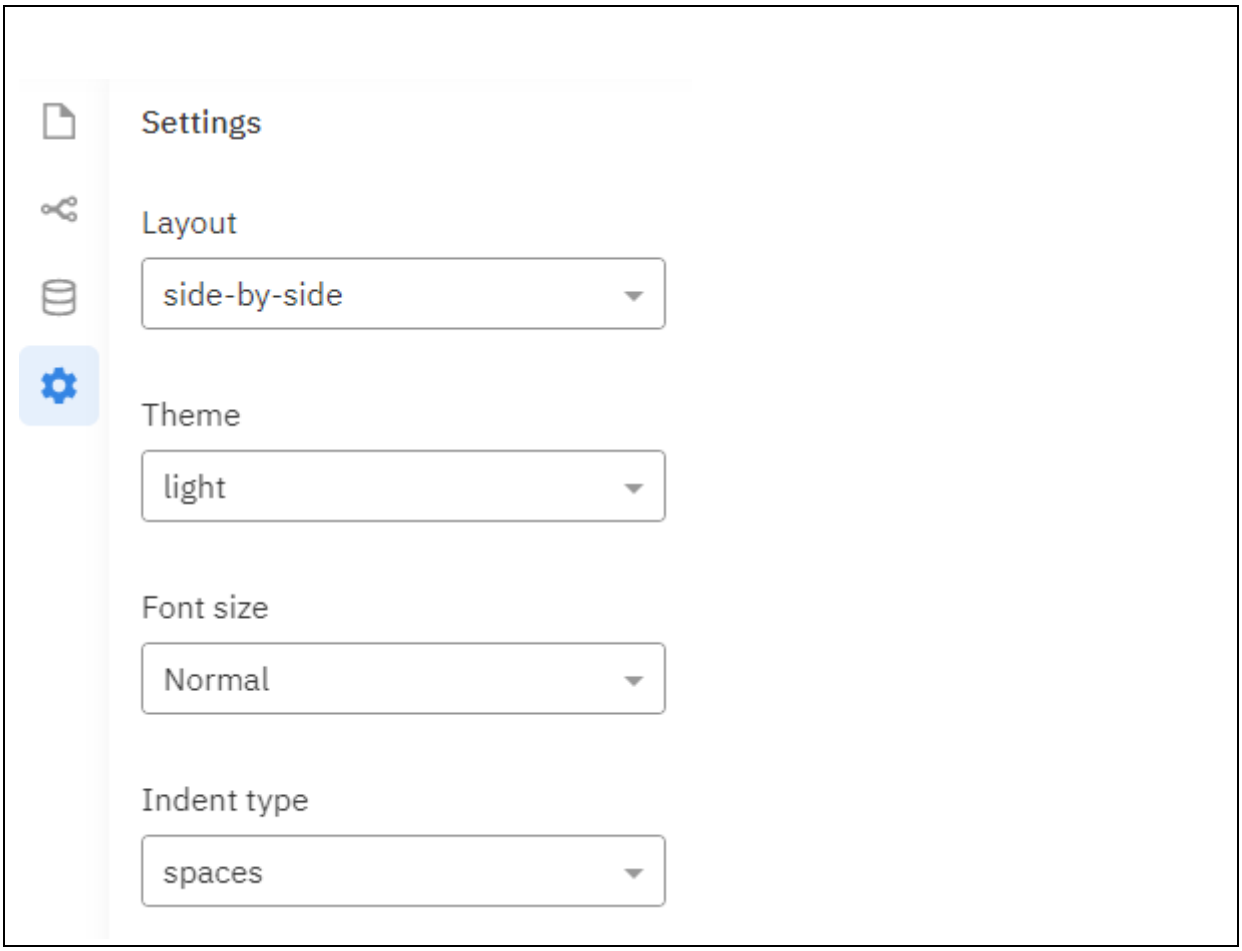
이후에 프로젝트를 다시 열고 싶을 때는 repl.it 사용자 페이지에서 [My Repls] 메뉴에 들어간 뒤, 열고 싶은 프로젝트 이름을 클릭하면 됩니다.



[My Repls] 화면

역자의 잠깐! **프로젝트 화면 설정하기**

프로젝트의 왼쪽에 있는 3개의 아이콘 중에서 세 번째 톱니바퀴 모양 아이콘을 클릭하면 프로젝트 화면을 설정할 수 있습니다. 레이아웃, 테마, 폰트 크기 등을 설정할 수 있으므로 편한 형태로 변경해서 사용하기 바랍니다.



repl.it 프로젝트 실행

생성된 프로젝트에는 기본적으로 다음과 같은 코드가 적혀 있습니다. 본문에서 배우겠지만 이는 'Hello World!'라는 문자열을 출력하는 코드입니다. 이 코드를 간단하게 실행해보도록 합시다.

```
#include <iostream>

int main() {
   std::cout << "Hello World!₩n";
}
```

repl.it 프로젝트 화면에서 위에 있는 [▶] 버튼을 클릭하면 코드를 실행할 수 있습니다.

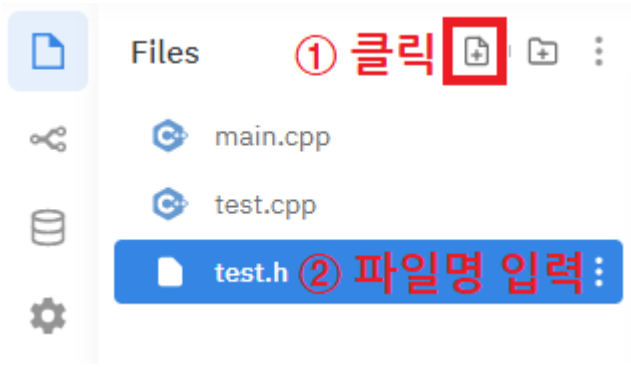
repl.it 프로젝트 실행

코드를 실행하면 오른쪽 터미널 화면에 명령어들이 입력되고, 'Hello World!'라는 문자열이 출력되는 것을 볼 수 있습니다. 'Hello World!'라는 문자열 앞뒤에 나오는 것은 C++ 코드를 실행하기 위한 명령어이므로 신경 쓰지 않아도 괜찮습니다.

```
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main
Hello World!
```

repl.it 프로젝트 분할 컴파일

책의 본문에서는 파일을 여러 개 만들어서 컴파일하기도 합니다. repl.it과 같은 온라인 IDE를 사용하면 별도의 명령어 없이 쉽게 파일을 여러 개 만들어서 컴파일하고 실행할 수 있습니다. 일단 실행 방법을 알아보기 위해서 test.h와 test.cpp라는 파일을 추가합니다. 프로젝트 왼쪽에 있는 [Files] 패널에서 위에 있는 [Add file] 아이콘을 클릭하고, 파일 이름을 지정하면 파일이 생성됩니다.



파일 추가

아직 배우지 않았지만, 파일에 간단하게 다음과 같은 코드를 입력해봅시다. test.h 파일에 아래와 같은 코드를 입력합니다.

```
#include <string>

std::string test();
```

test.cpp 파일에 아래와 같은 코드를 입력합니다.

```
#include "test.h"

std::string test() {
   return "Hello Compile";
}
```

main.cpp 파일에 아래와 같은 코드를 입력합니다.

```
#include <iostream>
#include "test.h"

int main() {
   std::cout << test() << std::endl;
   return 0;
}
```

이렇게 파일을 나누어 코드를 만든 경우에도 프로젝트의 run 버튼을 클릭하기만 하면 알아서 컴파일이 일어나고 실행됩니다.

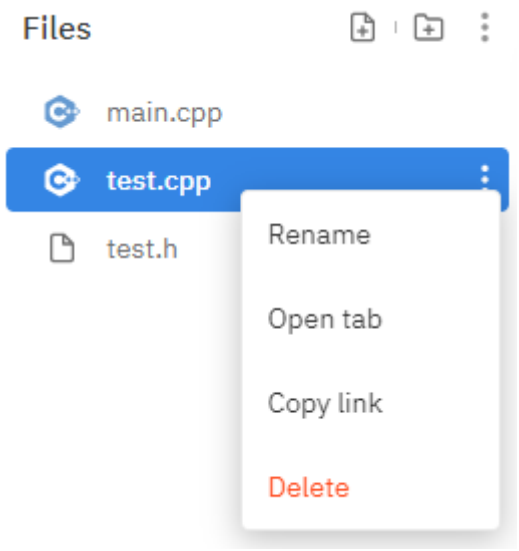
```
> clang++-7 -pthread -std=c++17 -o main main.cpp test.cpp
> ./main
Hello Compile
```

온라인 IDE를 사용하면 별도의 설치 없이 쉽게 코드를 입력할 수 있습니다. 또한 어디에서나 코

드를 작성할 수 있으므로, 학교에서 작성한 코드를 집에 와서도 이어서 작성할 수 있습니다. C++ 이외에도 다양한 프로그래밍 언어를 활용해볼 수 있으므로, 다른 프로그래밍 언어를 공부할 때도 적극적으로 활용해보기 바랍니다.

역자의 잠깐! **파일 이름 변경과 제거**

파일 이름을 변경하거나 제거하고 싶을 때는 파일에 마우스를 올릴 때 나오는 점 3개가 찍힌 아이콘을 클릭했을 때 나오는 컨텍스트 메뉴에서 Rename 또는 Delete 등을 선택해주세요.



2 로컬에서 실습

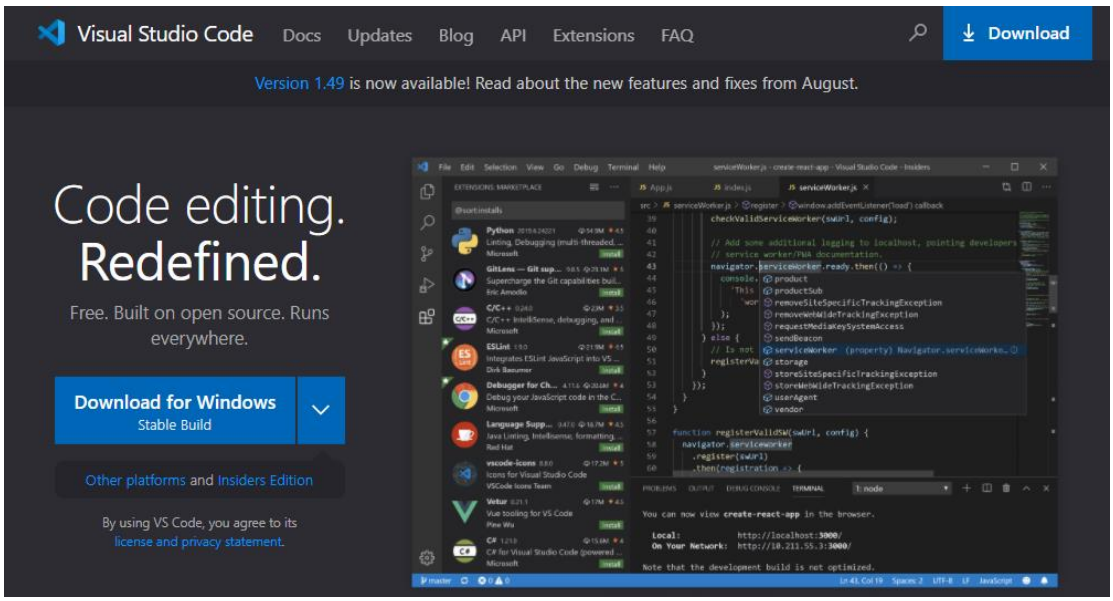
프로그래밍 언어는 기본적으로 '코드를 입력할 수 있는 코드 에디터'와 '코드를 실행할 수 있게 만들어 주는 프로그램(컴파일러 또는 인터프리터 등)'을 설치해야 사용할 수 있습니다. 이러한 2개의 요소와 다양한 지원 기능을 포함한 큰 규모의 프로그램을 통합 개발 환경(IDE)라고 부릅니다.

이 책에서는 간단하게 2개의 요소를 각각 설치하고 사용하는 방법을 살펴보도록 하겠습니다. 초보자 단계에서는 명령어를 입력하는 것이 조금 어렵게 느껴질 수 있지만, 익숙해지면 쉽게 사용할 수 있을 것입니다.

설치

■ 에디터 설치

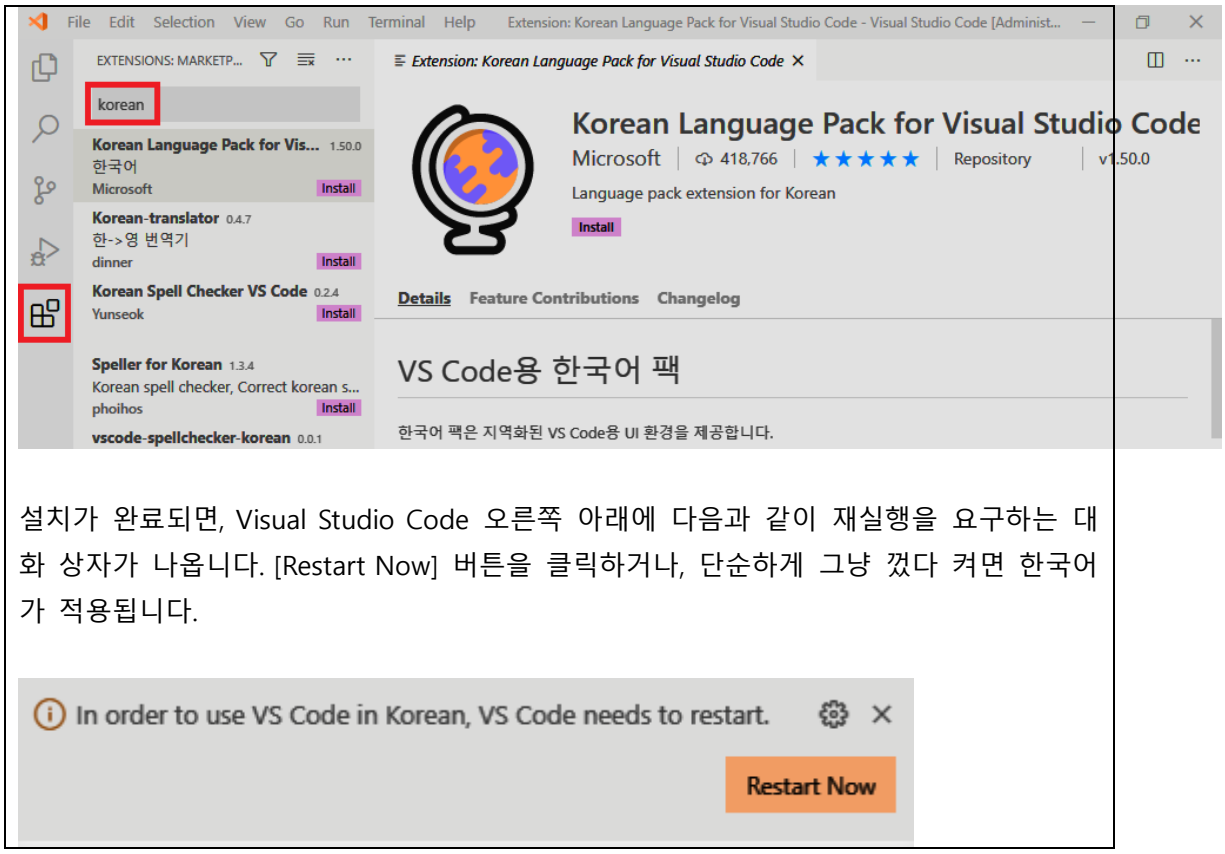
메모장 등의 단순하게 글자를 작성할 수 있는 프로그램도 사용할 수 있지만, 코드를 더 쉽게 입력할 수 있는 Visual Studio Code라는 에디터를 설치해서 사용하겠습니다. Visual Studio Code는 https://code.visualstudio.com에서 내려받을 수 있습니다. 메인 페이지의 [Download for Windows] 버튼을 클릭하면 설치 파일이 다운로드 됩니다.



Visual Studio Code 메인 페이지

다운로드가 완료되면, 설치 파일을 실행해서 설치를 진행해주세요. 설치는 [Next] 버튼만 누르면 간단하게 완료되므로 추가 설명하지 않겠습니다.

역자의 잠깐! **한국어팩 설치하기**
초보자라면 메뉴가 영어로 되어 있는 것이 부담으로 다가올 수 있습니다. 따라서 한국어팩을 설치하고 사용하는 것을 추천합니다. Visual Studio Code 왼쪽에 있는 5개의 아이콘 중에 가장 아래에 있는 [확장] 아이콘을 클릭하고, 'Korean'을 검색해서 'Korean Language Pack for Visual Studio Code'를 찾아서 클릭합니다. 클릭해서 오른쪽에 나오는 페이지에서 [Install] 버튼을 클릭하면 한국어팩이 설치됩니다.

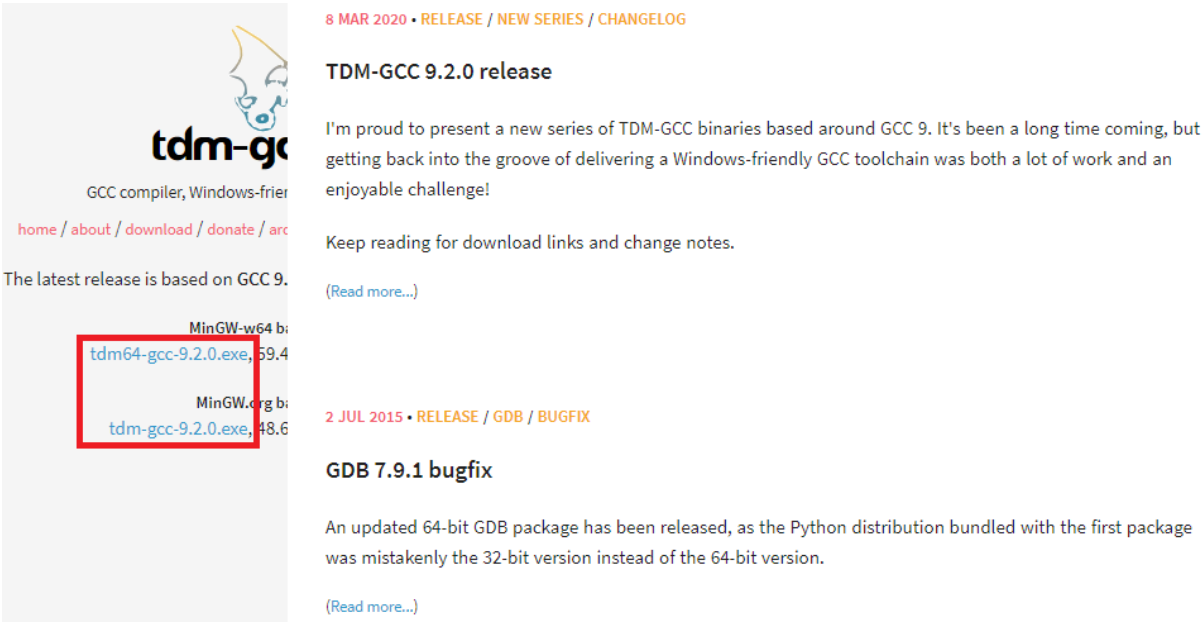


설치가 완료되면, Visual Studio Code 오른쪽 아래에 다음과 같이 재실행을 요구하는 대화 상자가 나옵니다. [Restart Now] 버튼을 클릭하거나, 단순하게 그냥 껐다 켜면 한국어가 적용됩니다.

■ gcc 컴파일러 설치

이제 코드를 실행할 수 있게 만들어주는 프로그램을 설치하겠습니다. C++의 경우는 컴파일러가 이러한 역할을 합니다. 윈도우에서 사용할 수 있는 다양한 컴파일러가 있는데, 이 책에서는 tdm-gcc 컴파일러를 사용하겠습니다. 맥과 우분투 리눅스에서 컴파일러를 설치하는 방법은 이번 절의 마지막 부분을 참고해주세요.

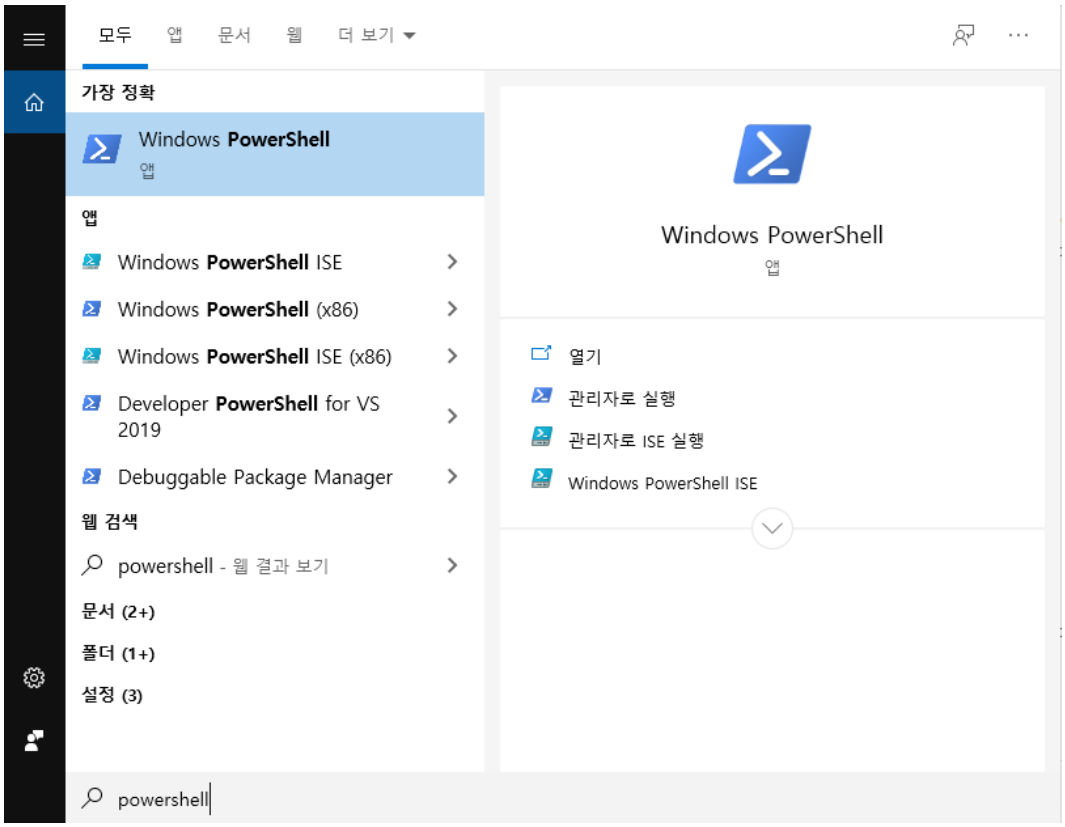
tdm-gcc 컴파일러는 다음 사이트에서 내려받을 수 있습니다. 자신의 컴퓨터 아키텍처(64비트 또는 32비트)에 맞춰 메인 페이지 왼쪽에 있는 tdm64-gcc-9.2.0.exe 또는 tdm-gcc-9.2.0.exe 링크를 선택해서 설치 파일을 내려받습니다. https://jmeubank.github.io/tdm-gcc에 접속하세요.



tdm-gcc 메인 페이지

이전과 마찬가지로 [Next] 버튼만 누르면 설치가 완료되므로 설치와 관련된 별도의 설명은 하지 않겠습니다. 설치가 완료되면 윈도우에서 powershell을 프로그램을 실행합니다. 이전 버전의 윈도우를 사용하고 있다면 명령 프롬프트를 실행해주세요.

윈도우 최신 버전의 경우 키보드의 [아이콘 추가]를 누릅니다. powershell을 입력하여 검색한 뒤 선택하면 간단하게 실행할 수 있습니다. 이전 버전의 윈도우를 사용하고 있다면, [아이콘 추가] + r을 누르고 'cmd'라고 입력하면 간단하게 실행할 수 있습니다.



powershell 실행

이어서 다음과 같이 'gcc --version'을 입력하면, 컴파일러가 제대로 설치된 경우에는 오류 없이 컴파일러의 정보가 출력됩니다. 설치했는데 출력이 제대로 되지 않는 경우에는 컴퓨터를 재시작하고 다시 해주세요.

```
> gcc --version
gcc.exe (tdm64-1) 9.2.0
Copyright (C) 2019 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
PS C:₩Users> gcc --version
gcc.exe (tdm64-1) 9.2.0
Copyright (C) 2019 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

정상으로 설치된 경우의 출력

역자의 잠깐! **macOS이나 우분투에서 gcc 사용하기**
맥의 경우, 터미널에서 다음 명령어를 입력하면 컴파일러가 설치됩니다. $ 뒤의 부분을 입력합니다.

```
# 기본 빌드 도구를 설치합니다
$ xcode-select --install
```

우분투의 경우, 터미널에서 다음 명령어를 입력하면 컴파일러가 설치됩니다. $ 뒤의 부분을 입력합니다.

```
# 최신 버전으로 설치하기 위해서 패키지 관리자를 업데이트합니다.
$ sudo apt update

# 기본 빌드 도구를 설치합니다(gcc가 포함되어 있습니다).
$ sudo apt-get install build-essential
```

C++ 프로그램 실행

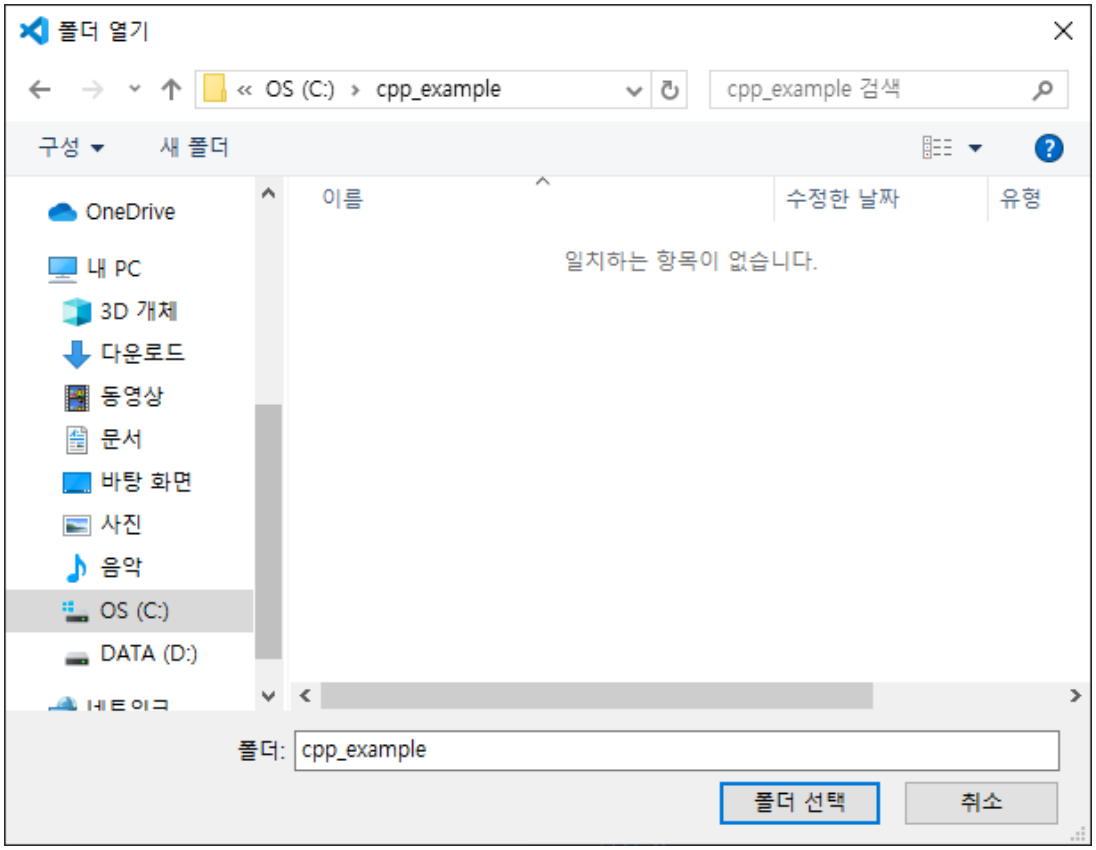
2개의 요소를 모두 설치했으므로, 이제 코드를 작성하고 실행하는 방법을 알아보겠습니다.

- Visual Studio Code로 폴더와 터미널 열기

  일단 Visual Studio Code를 실행합니다. 이어서 [파일]-[폴더 열기]를 클릭해서 [폴더 열기] 대화 상자를 표시합니다.

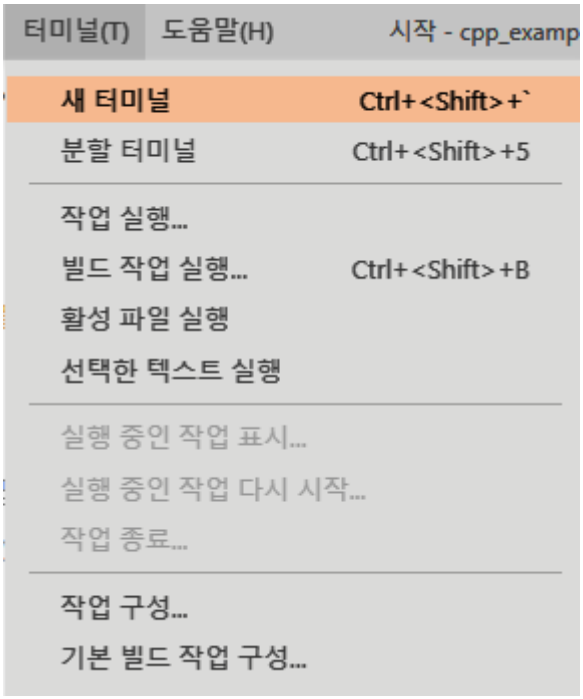
폴더 열기

[폴더 열기] 대화 상자가 나오면, C++ 코드를 작성할 폴더를 열고, [폴더 선택] 버튼을 클릭합니다. 여기서는 C:₩cpp_example이라는 폴더를 만들고 열었습니다.



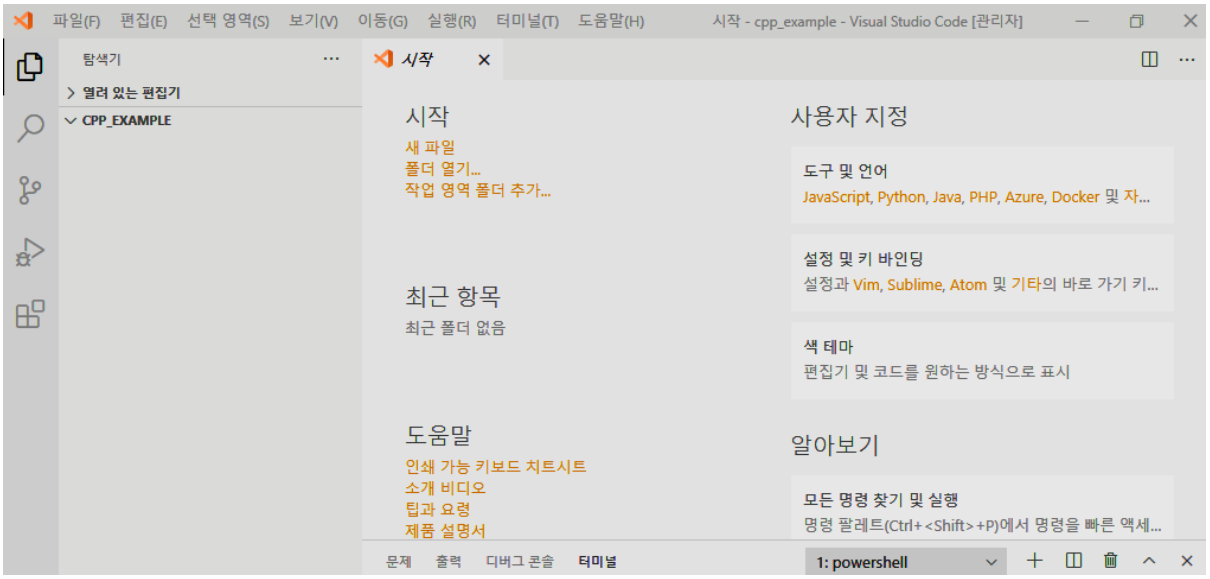
[폴더 열기] 대화 상자

이어서 [터미널]-[새 터미널] 버튼을 눌러서, [터미널] 패널을 엽니다.



새 터미널 열기

여기까지 완료했다면, Visual Studio Code의 화면이 다음과 같이 구성되어 있을 것입니다. 폴더를 열고 터미널 화면을 열었다면, 다른 폴더를 열기 전까지는 현재와 같은 화면 구성이 다음 실행 때도 유지됩니다.

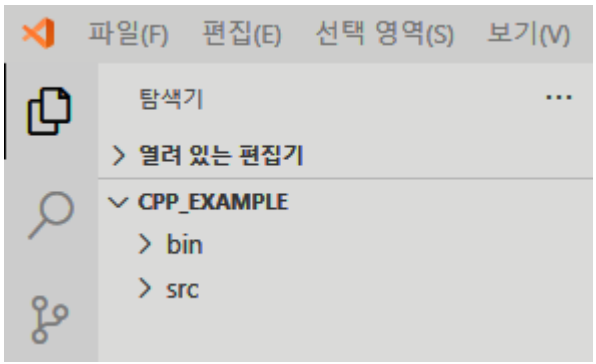


Visual Studio Code 기본 상태

■ 프로젝트 폴더 구성

그럼 프로젝트 폴더를 구성하겠습니다. 터미널에 'mkdir bin ; mkdir src'라고 입력해서 bin 폴더와 src 폴더를 생성합니다.

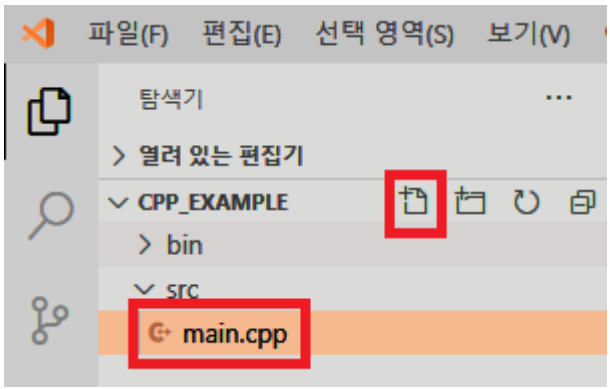
```
# mkdir 명령어로 폴더를 만들 수 있음
# bin 폴더와 src 폴더를 생성
# PowerShell에서 2개 이상의 명령어를 실행하고 싶다면 ; 기호로 연결
> mkdir bin ; mkdir src
```



생성된 폴더

■ 파일 작성

이어서 src 폴더 아래에 main.cpp 파일을 생성합니다. src 폴더를 선택하고, 폴더 이름 오른쪽의 [파일 생성] 아이콘을 클릭합니다. 다음으로 파일 이름을 main.cpp로 지정합니다.



main.cpp 파일 생성하기

이어서 main.cpp 파일에 다음과 같은 코드를 입력해봅시다. C++ 코드와 관련된 자세한 내용은 책에서 계속 다루므로, 코드와 관련된 자세한 설명은 생략하겠습니다. 이 코드는 단순하게 'Hello World!'라는 문자열을 출력하는 코드라고 생각해주세요.

```
#include <iostream>

int main() {
   std::cout << "Hello World!" << std::endl;
   return 0;
}
```

코드를 모두 입력했다면, [파일]-[저장](또는 ctrl + s 키)을 눌러서 저장해주세요. 처음 에디터를 사용할 때 가장 많이 실수하는 부분이 파일을 저장하지 않고 실행하는 것입니다.

■ 기본 컴파일

이어서 다음 명령어를 입력하면 gcc 컴파일러로 main.cpp 파일을 실행할 수 있습니다. > 는 입력하는 부분을 나타내는 것이지 실제로 입력하는 명령어가 아닙니다. 코드가 실행되면 'Hello World!'라는 글자를 출력합니다.

```
> gcc ./src/main.cpp -o ./bin/main -lstdc++ ; ./bin/main
Hello World!
```

명령어를 하나하나 분석해보면 다음과 같습니다. 여러 번 입력하다 보면 금방 익숙해질 것입니다.

```
gcc                                  # gcc 명령어를 사용
./src/main.cpp                       # ./src/main.cpp 파일을 사용하여
-o ./bin/main                        # 출력 파일로 ./bin/main을 생성
-lstdc++                             # c++ 기본 표준을 사용

;                                    # 그리고
```

```
./bin/main                                    # ./bin/main을 실행
```

g++ 명령어를 사용하면 코드를 조금 더 단순하게 입력할 수 있습니다. g++ 명령어는 gcc c++ 컴파일러를 의미하므로, -lstdc++ 옵션을 추가로 입력하지 않아도 됩니다. 내부적으로는 같은 동작을 합니다.

```
> g++ ./src/main.cpp -o ./bin/main ; ./bin/main
```

역자 코드를 실행할 때마다 위의 명령어를 계속 입력하는 것은 굉장히 귀찮은 일입니다. 한 번 입력한 코드는 키보드의 위 방향키(↑)를 눌러서 다시 읽어 들일 수 있습니다. 따라서 코드를 편집한 뒤, 터미널에 커서를 놓고 ↑ + Enter 과정을 거치면, 코드를 쉽게 다시 실행할 수 있습니다.

■ 분할 컴파일

책의 본문에서는 파일을 여러 개 만들어서 컴파일하기도 합니다. 파일이 여러 개 있을 때는 어떻게 해야 하는지 간단하게 알아보겠습니다. src 폴더에 test.h와 test.cpp라는 파일을 추가합니다. 참고로 bin 폴더 아래의 main.exe는 위의 명령어를 입력하면서 자동으로 만들어진 파일입니다.

```
├──bin
|     main.exe
└──src
     main.cpp
     test.cpp
     test.h
```

이어서 각각의 파일에 다음과 같은 코드를 입력해주세요. test.h 파일에 아래와 같은 코드를 입력합니다.

```
#include <string>

std::string test();
```

test.cpp 파일에 아래와 같은 코드를 입력합니다.

```
#include "test.h"

std::string test() {
   return "Hello Compile";
}
```

main.cpp 파일에 아래와 같은 코드를 입력합니다.

```
#include <iostream>
#include "test.h"

int main() {
   std::cout << test() << std::endl;
   return 0;
}
```

이어서 다음과 같은 명령어를 입력하면 코드를 실행할 수 있습니다.

```
> g++ ./src/main.cpp ./src/test.cpp -o ./bin/main ; ./bin/main
Hello Compile
```

명령어를 분석해보면 다음과 같습니다. 명령어를 분석해보면 그렇게 어려운 내용은 없으므로, 영어 단어처럼 외우는 것을 추천합니다.

```
g++                          # gcc 명령어를 사용
./src/main.cpp ./src/test.cpp # main.cpp와 test.cpp 파일을 사용 후
-o ./bin/main                # 출력 파일로 ./bin/main을 생성

;                            # 그리고

./bin/main                   # ./bin/main을 실행
```

■ 일괄 분할 컴파일 명령어

이전에 작성했던 명령어를 정리해보면, 다음과 같이 g++과 -o 옵션 사이에 파일들을 여러 개 입력하기만 하면 되었습니다. 따라서 이 부분에 파일 목록이 자동으로 들어가게 명령어를 작성하면, 범용적으로 활용할 수 있는 분할 컴파일 명령어를 만들 수 있습니다.

```
g++ <파일 목록> -o ./bin/main ; ./bin/main
```

윈도우 10의 PowerShell을 기준으로, Get-ChildItem 명령어를 활용하면, 원하는 문자열을 가진 파일 목록을 읽어 들일 수 있습니다.

```
> Get-ChildItem ./src/*.cpp
    디렉터리: C:₩cpp_example₩src
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2020-07-10     오전 5:47            108 main.cpp
-a----        2020-07-10     오전 5:48             73 test.cpp
```

그리고 여기에서 파일의 이름만 추출하려면, 뒤에 '| % {$_.FullName}'을 입력하면 됩니다. 이 책은 파워쉘 등의 명령어를 다루는 책은 아니므로 자세한 설명은 하지 않겠습니다.

```
> Get-ChildItem ./src/*.cpp | % {$_.FullName}
C:₩cpp_example₩src₩main.cpp
C:₩cpp_example₩src₩test.cpp
```

따라서 이를 활용해서 다음과 같은 명령어를 작성해서 활용하면, 코드를 쉽게 실행할 수 있습니다. 어떤 의미의 명령어인지 간단하게 이해해보고, 메모장 등에 정리해서 C++ 프로그램을 실행할 때 활용하기 바랍니다.

```
g++ (Get-ChildItem ./src/*.cpp | % {$_.FullName}) -o ./bin/main -std=c++17 -lm; ./bin/main
```

역자의 팁! **우분투와 맥의 경우**

우분투와 맥 등을 사용하고 있다면 다음과 같은 명령어로 같은 작업을 할 수 있습니다.

```
g++ $(find src/ -name *.cpp -o -name *.c) -o bin/main -g -std=c++17 -lm && ./bin/main
```