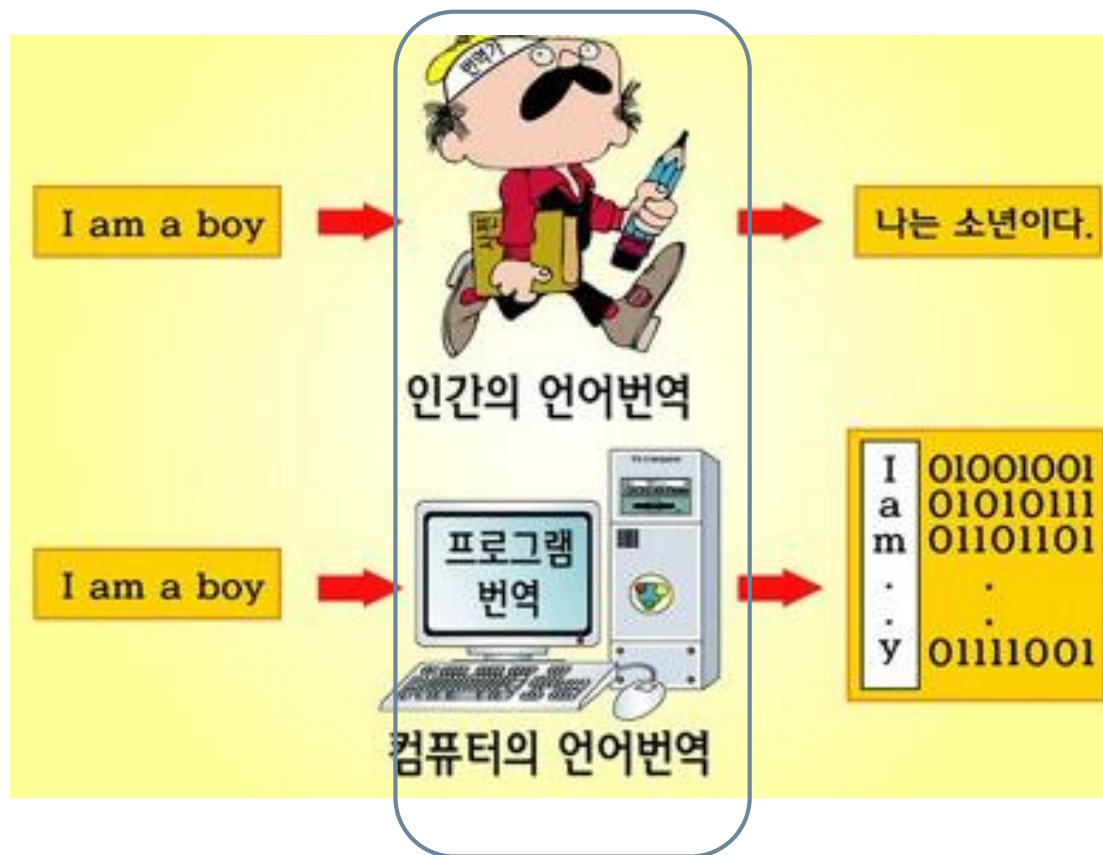




CH01 프로그래밍 기초



프로그래밍 언어와 문법이란?



문법



Bit 와 Byte

비트 (binary digit, bit)

0과 1, 두 가지 값만
가질 수 있는 측정 단위



0 OFF FALSE



1 ON TRUE

바이트 (Byte)

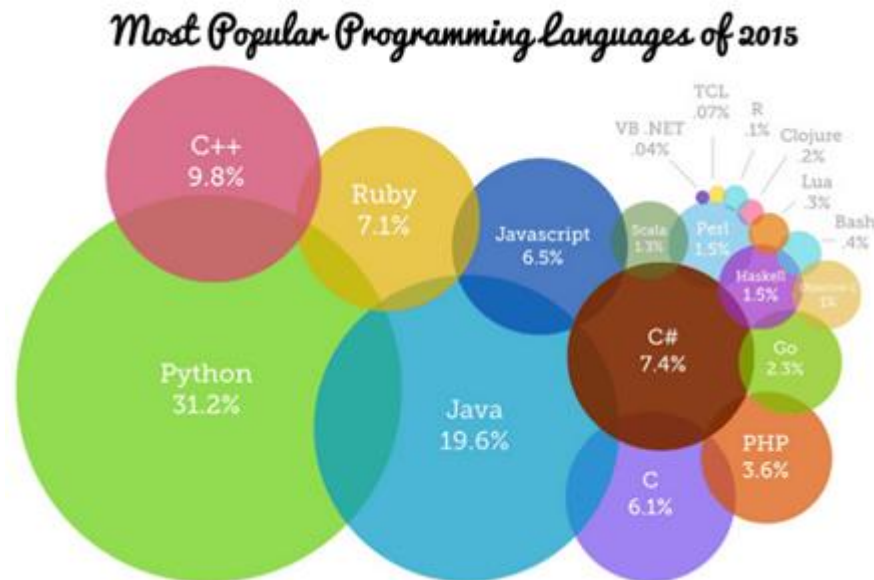
여덟 개의 비트로 구성된
데이터의 양을 나타내는 단위





프로그래밍 언어의 종류

- 많이 사용되는 언어들에는 '파이썬', '자바', 'C#', 'BASIC' 등이 있다.





프로그래밍 언어의 종류

언어	장점	단점
C, C++	빠른 속도 하드웨어 제어 가능	복잡한 문법 메모리 관리 부담
Python	사용하기 쉬운 문법 메모리 관리 쉬움 광범위한 라이브러리 인터프리터 언어	C++보다 속도 느림
Java	플랫폼 독립성 메모리 관리가 쉬움 풍부한 생태계	느린 속도 Python 보다 복잡한 문법 메모리 소모가 많음



프로그래밍 언어 문법

프로그래밍 문법

기본 문법

객체 관련 문법



프로그램 개발 과정

요구사항 분석

무엇을
만들 것인가를
결정한다.



설계

알고리즘을
설계한다.



구현

개발 도구를
사용하여
소스 코드를
작성한다.



테스팅

여러가지
경우에 대하여
실행하여 본다.



유지보수

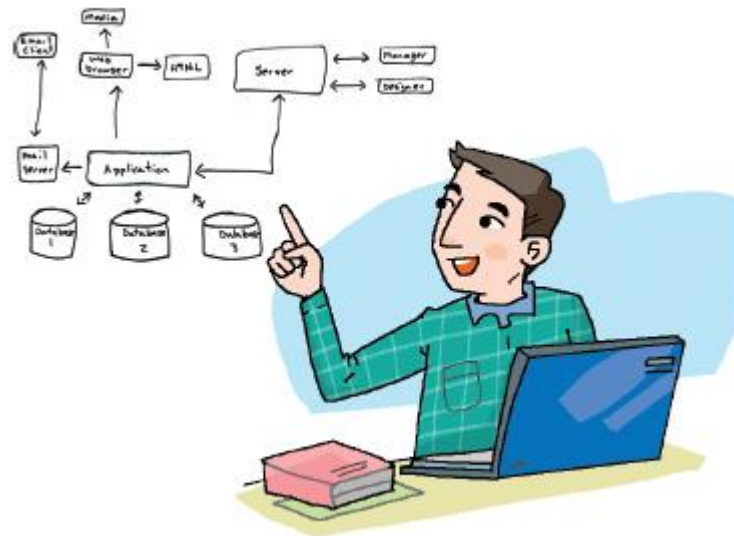
사용자의 추가
요구사항을
반영한다.





설계

- 문제를 해결하는 알고리즘을 개발하는 단계
- 순서도와 의사 코드를 도구로 사용
- 알고리즘은 프로그래밍 언어와는 무관
- 알고리즘은 원하는 결과를 얻기 위하여 밟아야 하는 단계에 집중적으로 초점을 맞추는 것





소스 작성

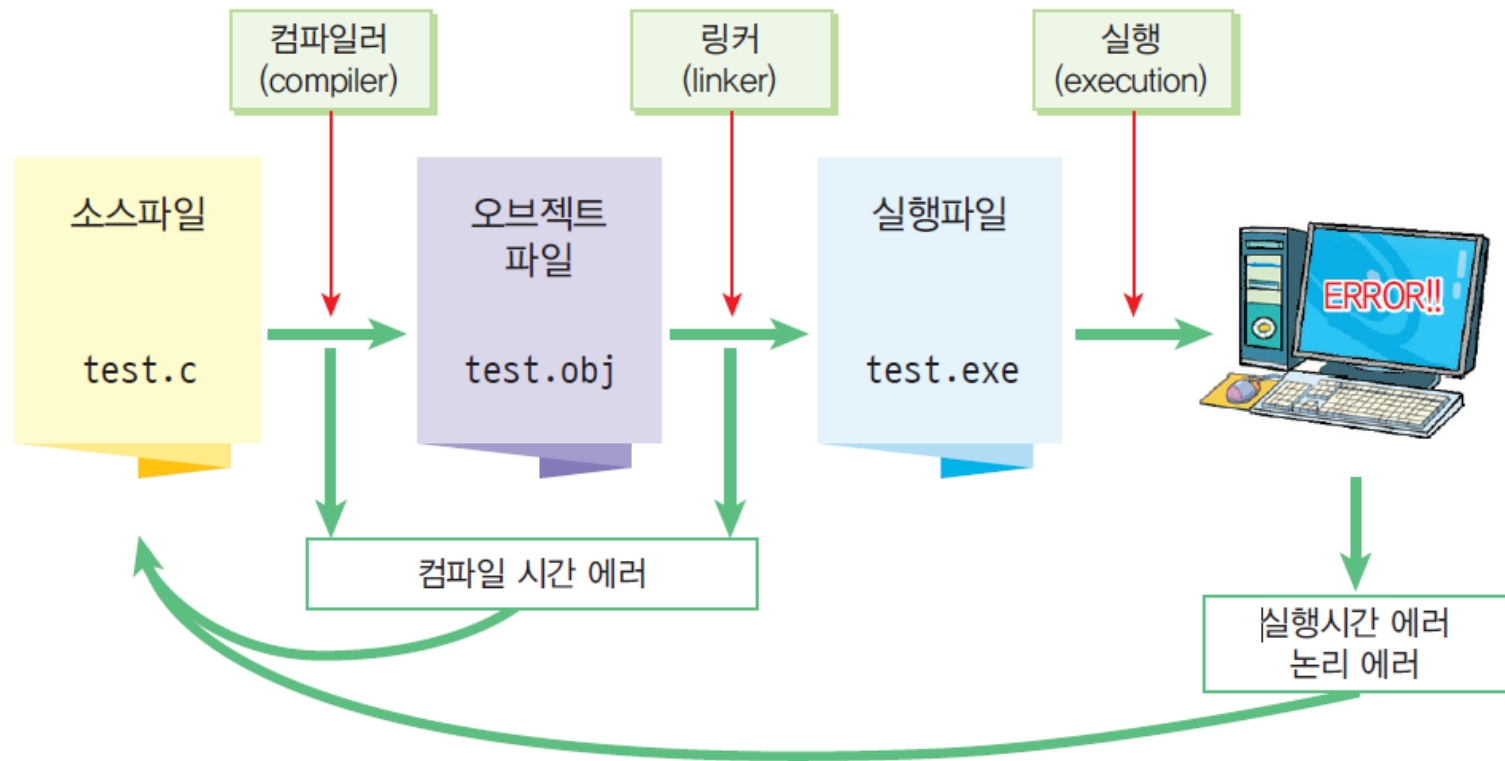
- 알고리즘의 각 단계를 프로그래밍 언어를 이용하여 기술
- 알고리즘을 프로그래밍 언어의 문법에 맞추어 기술한 것을 *소스 프로그램(source program)*이라고 한다.
- 소스 프로그램은 주로 텍스트 에디터나 통합 개발 환경을 이용하여 작성한다.
- 소스 파일 이름: (예) `test.c`



```
int main(void)
{
    printf("Hello World!");
}
```



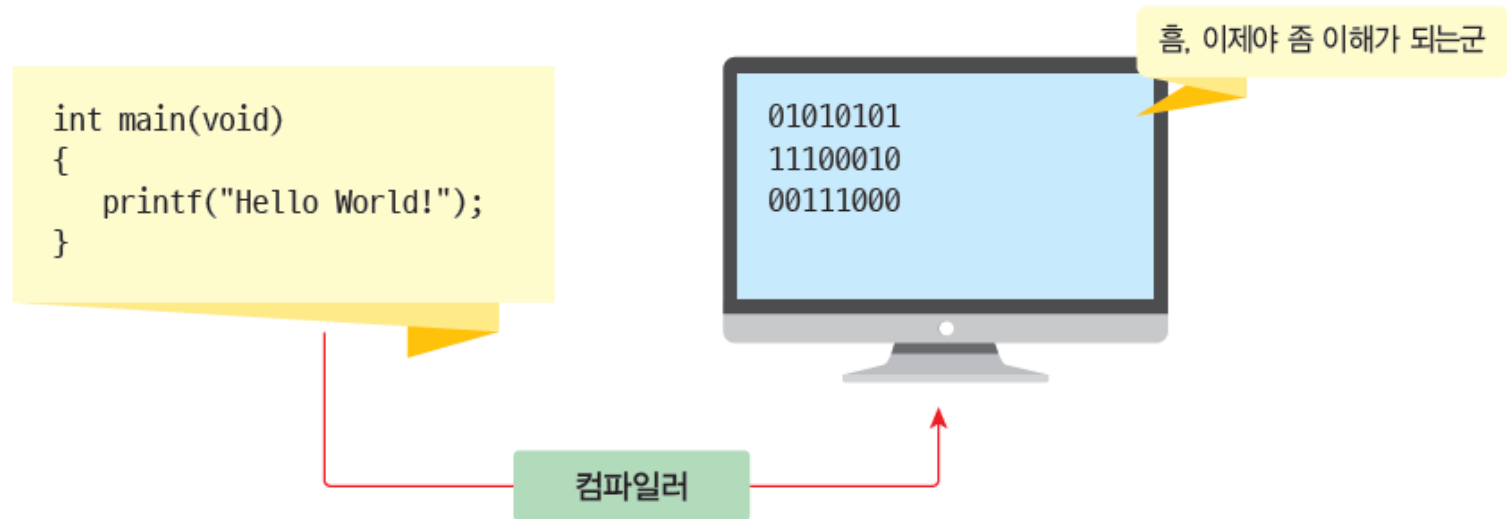
C, C++ 절차





컴파일

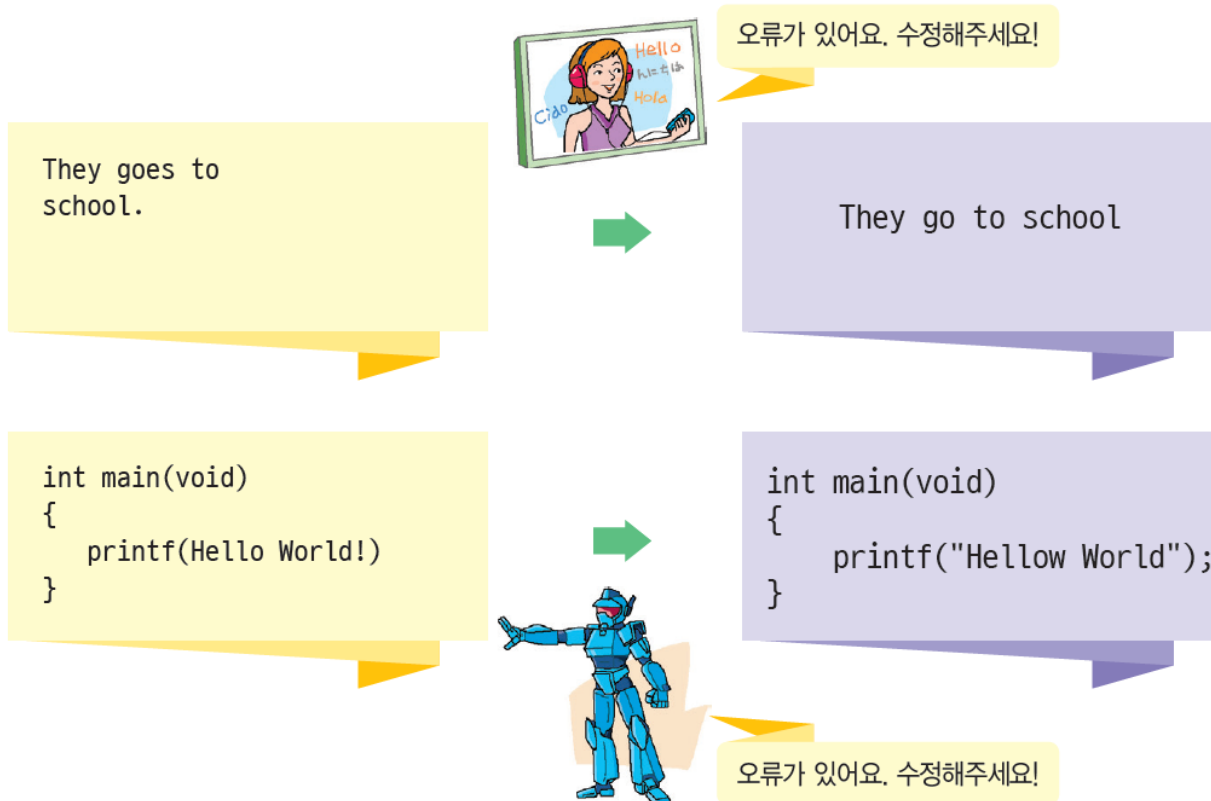
- 소스 프로그램을 오브젝트 파일로 변환하는 작업
- 오브젝트 파일 이름: (예) test.obj





컴파일 오류

- 컴파일 오류(compile error): 문법 오류
 - (예) He go to school;





링크

- 컴파일된 목적 프로그램을 라이브러리와 연결하여 실행 프로그램을 작성하는 것
- 실행 파일 이름: (예) test.exe
- *라이브러리(library)*: 프로그래머들이 많이 사용되는 기능을 미리 작성해 놓은 것
 - (예) 입출력 기능, 파일 처리, 수학 함수 계산
- 링크를 수행하는 프로그램을 *링커(linker)*라고 한다.



오브젝트 파일

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    printf("Hello World!");
```

```
    return 0;
```

```
}
```

소스 파일



```
010101000001111101010
```

```
101010100000111110101
```

```
010101010000011111010
```

```
101010101000001111101
```

```
010101010100000111110
```

```
101001010100000111110
```

```
10101...
```

오브젝트 파일



실행 및 디버깅

- 실행 시간 오류(run time error):
 - 0으로 나누는 것
 - 잘못된 메모리 주소에 접근하는 것
- 논리 오류(logical error): 문법은 틀리지 않았으나 논리적으로 정확하지 않는 것
 - (예)

- ① 그릇1과 그릇2를 준비한다.
- ② 그릇1에 밀가루, 우유, 계란을 넣고 잘 섞는다.
- ③ 그릇2를 오븐에 넣고 30분 동안 350도로 굽는다.

실수로 빈그릇을 오븐에 넣는다면
논리적인 오류입니다.





디버깅

- 소스에 존재하는 오류를 잡는 것





통합 개발 환경

- 통합 개발 환경 (IDE: integrated development environment) = 에디터 + 컴파일러 + 디버거





통합 개발 환경의 예

- 비주얼 스튜디오: 마이크로소프트
- 이클립스(eclipse): 오픈 소스 프로젝트
- Dev-C++: 오픈 소스 프로젝트





비주얼 스튜디오 버전

- 커뮤니티(Visual Studio Community) 버전은 “기업 외 응용 프로그램 빌드 개발자를 위한 완벽한 기능의 확장 가능한 무료 도구”이다.
- 프로페셔널 버전(Visual Studio Professional)은 “개별 개발자 또는 소규모 팀을 위한 전문적인 개발자 도구 및 서비스”라고 되어 있다.
- 엔터프라이즈 버전(Visual Studio Enterprise)은 “고급 테스트 및 DevOps를 포함해서 어떠한 크기나 복잡한 프로젝트까지 개발 팀을 위한 고급 기능이 포함된 엔터프라이즈급 솔루션”라고 표시되어 있다.



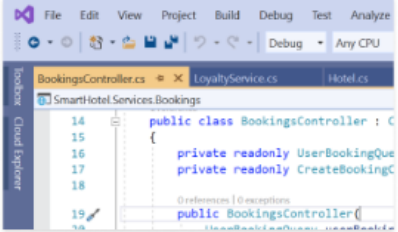
비주얼 스튜디오 설치

Visual Studio IDE, 코드 편집기, / x +

visualstudio.microsoft.com/ko/

모든 개발자를 위한 최상의 도구

Visual Studio



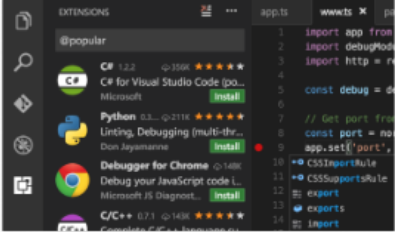
코딩, 디버그, 테스트 및 모든 플랫폼에 배포할 수 있는 완전한 기능을 갖춘 IDE

Visual Studio 다운로드

Community 2019 ↓

Professional 2019 ↓

Visual Studio Code



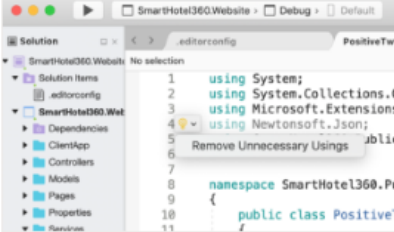
모든 OS 편집 및 디버깅

Visual Studio Code를 사용하면 다음에 동의하는 것입니다. [라이선스 및 개인정보처리방침](#)

Visual Studio 코드 다운로드

[자세히 보기 >](#)

Visual Studio for Mac



.NET을 사용하여 iOS, Android 및 웹용 앱 및 게임 개발

[라이선스 활성화에 대한 자세한 정보](#)

Visual Studio for Mac 다운로드

[자세히 보기 >](#)

<https://visualstudio.microsoft.com/ko/thank-you-downloading-visual-studio/?sku=Community&rel=16>

컴퓨터



비주얼 스튜디오 설치

Visual Studio를 다운로드해 주셔서 감사합니다.

다운로드가 곧 시작됩니다. 다운로드가 시작되지 않은 경우에는 [여기를 클릭하여 다시 시도하세요](#)

지금 개발을 시작하세요.

- Visual Studio 설치에 대한 지원 받기
- 다음 C++ 자습서에 따라 지금 데스크톱 앱 만들기
- C++ 빠른 시작 가이드를 사용하여 데스크톱 앱 개발 시작하기

Visual Studio 둘러보기 >

vs_community_1....exe



비주얼 스튜디오 설치

설치 중 — Visual Studio Community 2019 — 16.8.3

워크로드 개별 구성 요소 언어 팩 설치 위치

웹 및 클라우드 (4)

ASP.NET 및 웹 개발
Docker 지원이 포함된 ASP.NET Core, ASP.NET, HTML/JavaScript 및 컨테이너를 사용하여 웹 애플리케이션

Python 개발
Python에 대한 편집, 디버깅, 대화형 개발 및 소스 제어입니다.

Azure 개발
.NET Core 및 .NET Framework를 사용하여 클라우드 앱을 개발하고 리소스를 만들기 위한 Azure SDK, 도구 및 프로젝트

Node.js 개발
비동기 이벤트 구동 JavaScript 런타임인 Node.js를 사용하여 확장 가능한 네트워크 애플리케이션을 빌드합니다.

데스크톱 및 모바일 (5)

.NET 데스크톱 개발
.NET Core and .NET Framework와 함께 C#, Visual Basic 및 F#를 사용하여 WPF, Windows Forms 및 콘솔 애플리케이션

C++를 사용한 데스크톱 개발
MSVC, Clang, CMake 또는 MSBuild 등 선택한 도구를 사용하여 Windows용 최신 C++ 앱을 빌드합니다.

유니버설 Windows 플랫폼 개발
C#, VB 또는 C++(선택 사항)를 사용하여 유니버설 Windows 플랫폼용 애플리케이션을 만듭니다.

.NET을 사용한 모바일 개발
Xamarin을 사용하여 iOS, Android 또는 Windows용 플랫폼 간 애플리케이션을 빌드합니다.

위치
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community 변경...

계속하면 선택한 Visual Studio 버전에 대한 [라이선스](#)에 동의하는 것입니다. Microsoft는 Visual Studio와 함께 다른 소프트웨어를 다운로드할 수 있는 기능도 제공합니다. 이 소프트웨어는 [타사 고지 사항](#) 또는 해당 라이선스에 명시된 대로 별도로 사용이 허가됩니다. 계속하면 이러한 라이선스에도 동의하는 것입니다.

필요한 총 공간 7.38GB

다운로드하는 동안 설치 설치

설치 세부 정보

> Visual Studio 핵심 편집기

✓ C++를 사용한 데스크톱 개발 포함됨

✓ C++ 핵심 데스크톱 기능

✓ IntelliCode

옵션

✓ MSVC v142 - VS 2019 C++ x64/x86 빌드 도구(v...

✓ Windows 10 SDK(10.0.18362.0)

✓ Just-In-Time 디버거

✓ C++ 프로파일링 도구

✓ Windows용 C++ CMake 도구

✓ 최신 v142 빌드 도구용 C++ ATL(x86 및 x64)

✓ Test Adapter for Boost.Test

✓ Test Adapter for Google Test

✓ Live Share

✓ C++ AddressSanitizer(실험적)

☐ 최신 v142 빌드 도구용 C++ MFC(x86 및 x64)

☐ v142 빌드 도구용 C++/CLI 지원(14.28)

☐ v142 빌드 도구용 C++ 모듈(x64/x86 - 실험적)

☐ Windows용 C++ Clang 도구(10.0.0 - x64/x86)






비주얼 스튜디오 시작

Visual Studio 2019

최근 파일 열기(R)

오늘

-  **Project7.sln** 2020-07-26 오후 10:00
C:\Users\kim\source\repos\Project7
-  **Project6.sln** 2020-07-26 오후 5:55
C:\Users\kim\source\repos\Project6
-  **Project3.sln** 2020-07-26 오전 11:54
C:\Users\kim\source\repos\Project3

어제

-  **ConsoleApplication2.sln** 2020-07-25 오후 4:22
C:\Users\kim\source\repos\ConsoleApplication2
-  **Project5.sln** 2020-07-25 오후 4:10
C:\Users\kim\source\repos\Project5
-  **Project4.sln** 2020-07-25 오후 3:10
C:\Users\kim\source\repos\Project4

시작



리포지토리 복제(C)

GitHub 또는 Azure DevOps 같은 온라인 리포지토리에서 코드 가져오기



프로젝트 또는 솔루션 열기(P)

로컬 Visual Studio 프로젝트 또는 .sln 파일 열기



로컬 폴더 열기(F)

폴더 내에서 탐색 및 코드 편집



새 프로젝트 만들기(N)

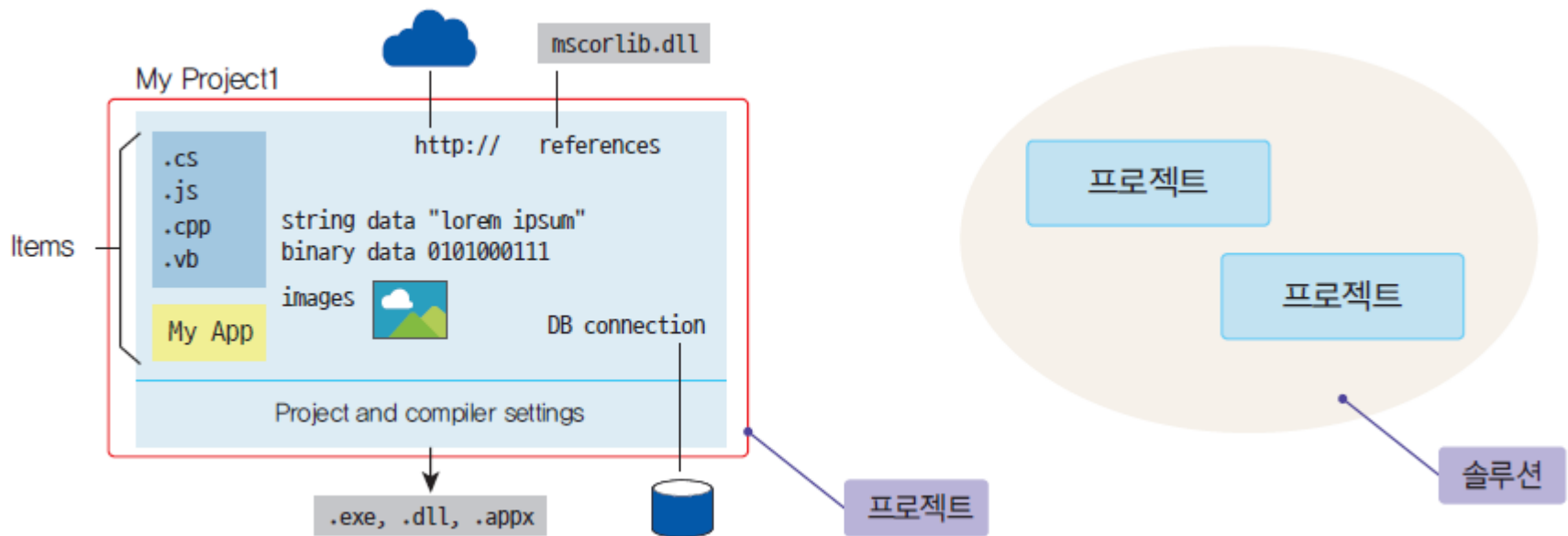
시작하려면 코드 스캐폴딩과 함께 프로젝트 템플릿을 선택하세요.

[코드를 사용하지 않고 계속\(W\) →](#)



워크스페이스와 프로젝트

- **솔루션(solution)**; 문제 해결에 필요한 프로젝트가 들어 있는 컨테이너
- **프로젝트(project)**; 하나의 실행 파일을 만드는데 필요한 여러 가지 항목들이 들어 있는 컨테이너







프로젝트 생성하기

새 프로젝트 만들기

최근 프로젝트 템플릿(R)

빈 프로젝트

C++

템플릿 검색(Alt+S)(S)

모든 언어(L)

모든 플랫폼(P)

모든 프로젝트 형식(T)



빈 프로젝트

Windows용 C++를 사용하여 처음부터 시작합니다. 시작 파일을 제공하지 않습니다.

콘솔

C++

Windows



콘솔 앱

Windows 터미널에서 코드를 실행합니다. 기본적으로 "Hello World"를 출력합니다.

콘솔

C++

Windows



Windows 데스크톱 마법사

마법사를 사용하여 고유한 Windows 앱을 만드세요.

콘솔

C++

데스크톱

라이브러리

Windows



Windows 데스크톱 애플리케이션

Windows에서 실행되는 그래픽 사용자 인터페이스를 사용하는 애플리케이션용 프로젝트입니다.

C++

데스크톱

Windows



공유 항목 프로젝트

공유 항목 프로젝트는 여러 프로젝트 간에 파일을 공유하는 데 사용됩니다.

Android

콘솔

C++

데스크톱

게임

iOS

라이브러리

Linux

모바일

UWP

Windows

다음(N)



프로젝트 생성하기

새 프로젝트 구성

빈 프로젝트 콘솔 C++ Windows

프로젝트 이름(N)

hello

위치(L)

C:\Users\Wkim\source\repos

솔루션 이름(M) ⓘ

hello

☒ 솔루션 및 프로젝트를 같은 디렉터리에 배치(D)

(1) hello로 입력

(2) 원하는 디렉토리 선택

뒤로(B)

만들기(C)



소스 파일 생성하기

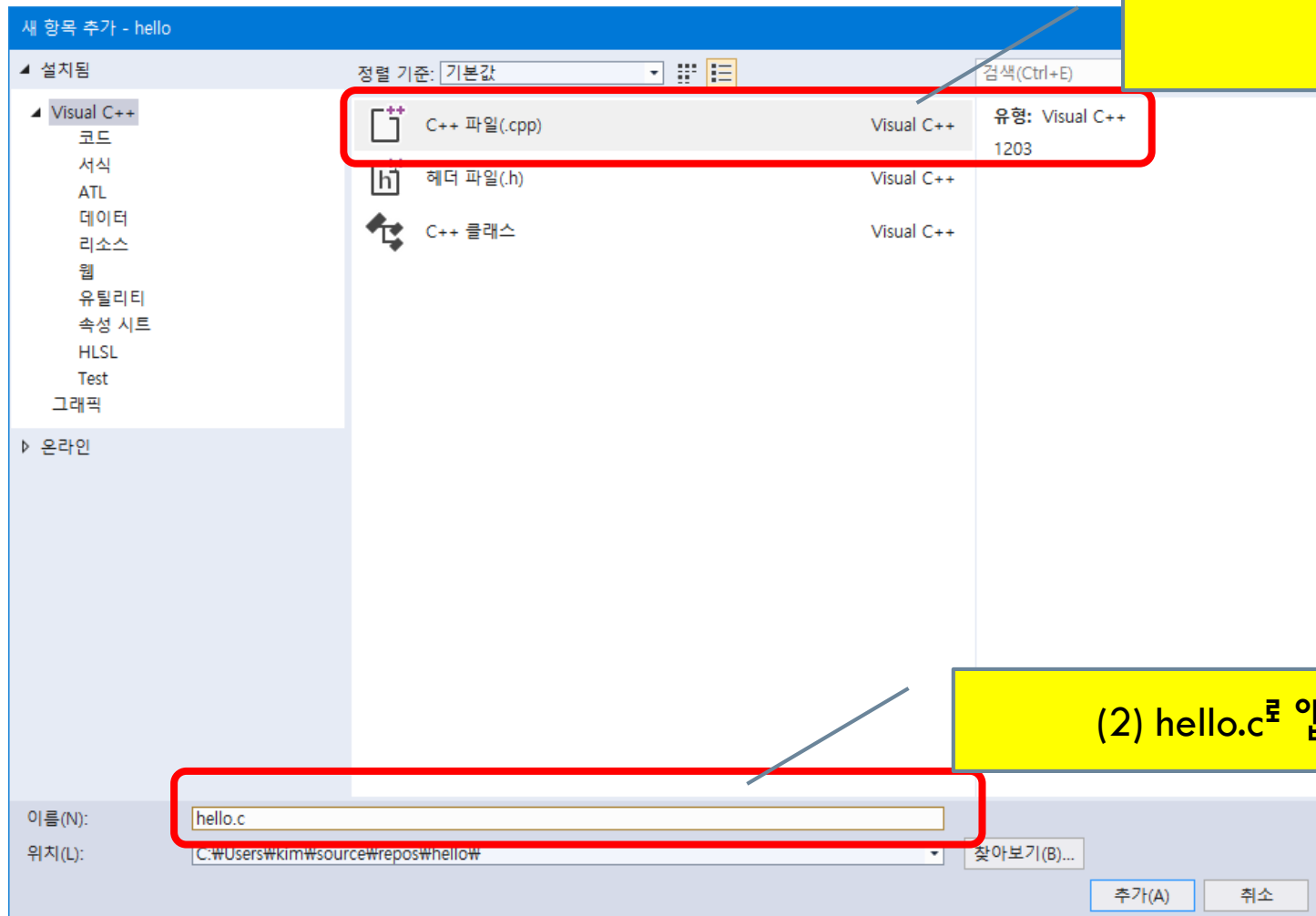
The screenshot shows the Visual Studio IDE interface. The 'Source Files' menu is open, and the 'Add' option is highlighted with a red rectangle. A yellow arrow points to the 'Add' option. The 'Add' option is located in the 'Source Files' menu, which is part of the 'View' menu. The 'Add' option is located in the 'Source Files' menu, which is part of the 'View' menu. The 'Add' option is located in the 'Source Files' menu, which is part of the 'View' menu.

이 항목은 미리 보기를 지원하지 않습니다.

소스 제어에 추가



소스 파일 생성하기





프로그램 입력

Visual Studio IDE interface showing a C program named 'hello.c' with the following code:

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     printf("Hello World!");
6     return;
7 }
```

The code is highlighted with a red box. A yellow callout box points to the code with the text: **철자에 주의하여 입력** (Enter with attention to spelling).

The output window shows the following text:

```
출력 보기 선택(S): 빌드
다시 빌드 시작...
1>----- 모두 다시 빌드 시작: 프로젝트: hello, 구성: Debug Win32 -----
1>hello.c
1>hello.vcxproj -> C:\Users\kim\source\repos\hello\Debug\hello.exe
===== 모두 다시 빌드: 성공 1, 실패 0, 생략 0 =====
```

The bottom status bar indicates '준비' (Ready).



기호는 정확하게 입력

문장의 끝에는 세미콜론



컴파일하기

The screenshot shows the Visual Studio IDE with the 'Build' menu open. A yellow lightning bolt points to the 'Build' menu. The 'Build' menu is highlighted with a red box, and the 'Build' option is also highlighted with a red box. The output window at the bottom shows the build process for the 'hello' project.

Build Menu Options:

- 솔루션 빌드(B) Ctrl+Shift+B
- 솔루션 다시 빌드(R)
- 솔루션 정리(C)
- 솔루션의 전체 프로그램 데이터베이스 파일 빌드
- 솔루션에서 코드 분석 실행(Y) Alt+F11
- hello 빌드(U) Ctrl+B
- hello 다시 빌드(E)
- hello 정리(N)
- hello에서 코드 분석 실행(A)
- 프로젝트만(J)
- 일괄 빌드(T)...
- 구성 관리자(O)...
- 컴파일(M) Ctrl+F7
- 파일에서 코드 분석 실행(F) Ctrl+Shift+Alt+F7

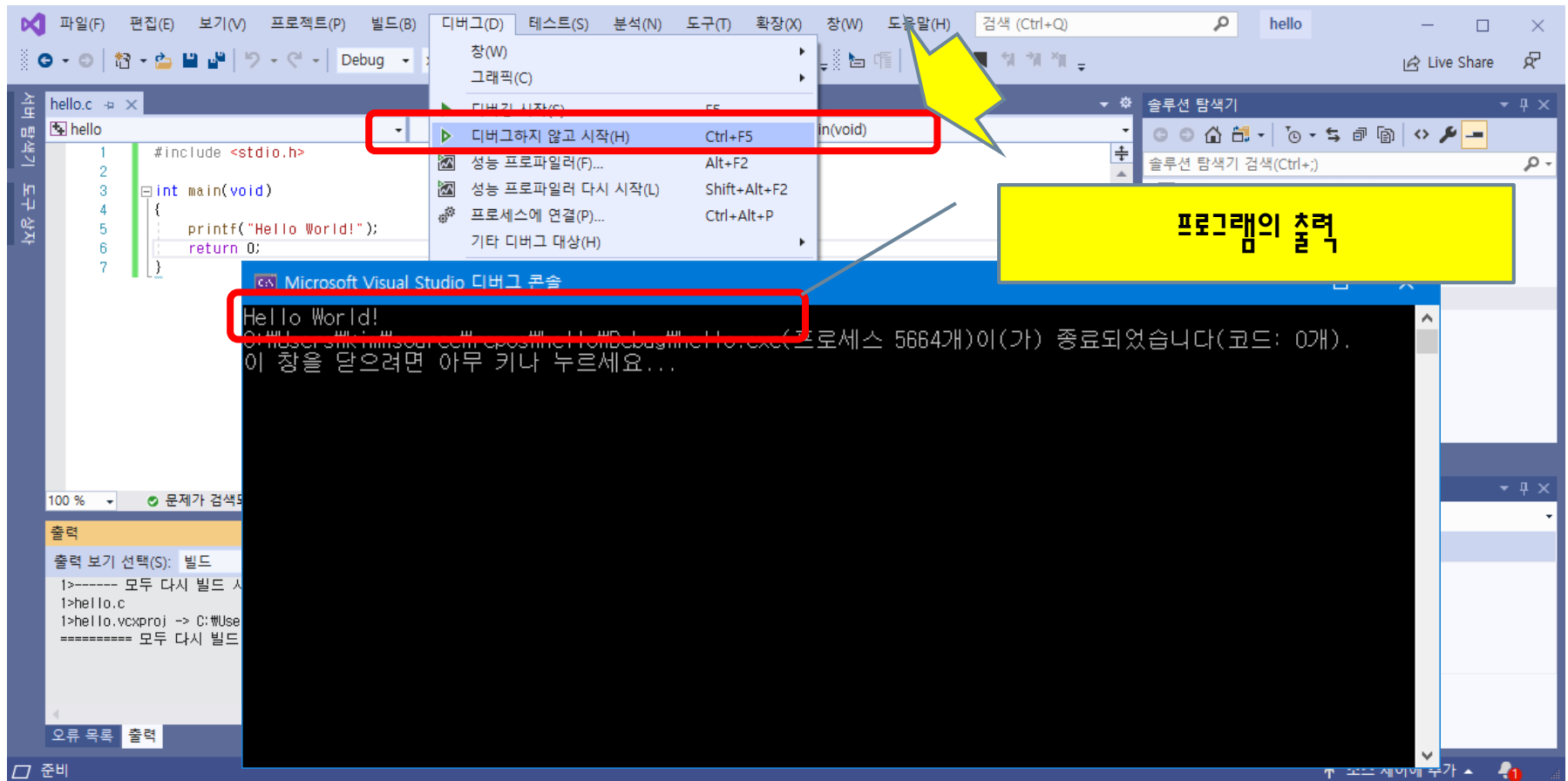
Output Window:

```
출력 보기 선택(S): 빌드
1>----- 모두 다시 빌드 시작: 프로젝트: hello, 구성: Debug Win32 -----
1>hello.c
1>hello.vcxproj -> C:\Users\kim\source\repos\hello\Debug\hello.exe
===== 모두 다시 빌드: 성공 1, 실패 0, 생략 0 =====
```

모두 다시 빌드했습니다.



프로그램 실행 하기





첫 번째 프로그램의 설명

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    printf("Hello World!");
```

```
    return 0;
```

```
}
```



Hello World!

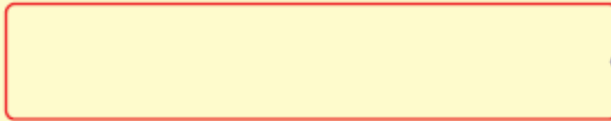


작업을 적어주는 위치

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```



```
    return 0;
```

```
}
```

여기다 원하는 작업을
수행하는 문장을 적어준다.

프로그램



간략한 소스 설명

```
#include <stdio.h>
```

헤더파일을 포함한다.

```
int main(void)
```

메인 함수 시작

```
{
```

```
    printf("Hello World!");
```

화면에 "Hello World!"를 출력

```
    return 0;
```

외부로 0값을 반환

```
}
```

메인 함수 종료

프로그램



헤더 파일 포함

- `#include`는 소스 코드 안에 특정 파일을 현재의 위치에 포함

- 주의!: 전처리기 지시자 문장 끝에는 세미콜론(;)을 붙이면 안 된다.

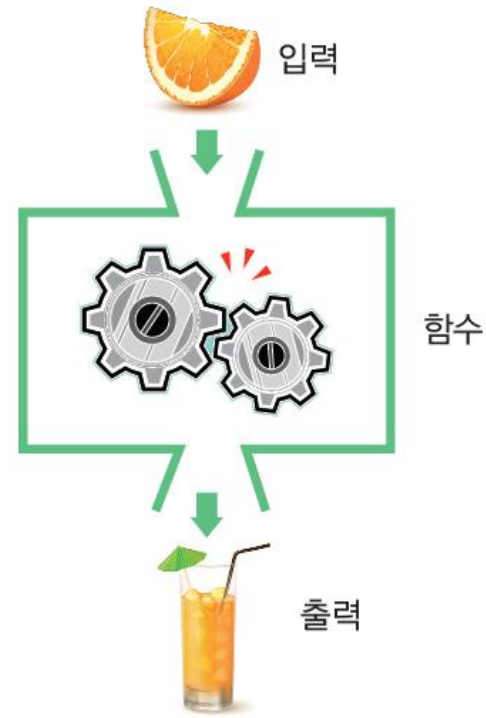
`#include <stdio.h>`

- 헤더 파일(header file): 컴파일러가 필요로 하는 정보를 가지고 있는 파일
- stdio.h: standard input output header file



함수

- 함수(function): 특정한 작업을 수행하기 위하여 작성된 독립적인 코드
- (참고) 수학적 함수 $y = x^2 + 1$
- 프로그램 = 함수의 집합





함수의 간단한 설명

함수의 출력 타입

int

함수의 이름

main

함수의 입력 타입

(void)

{

함수의 시작

```
printf("Hello World");  
return 0;
```

함수의 몸체

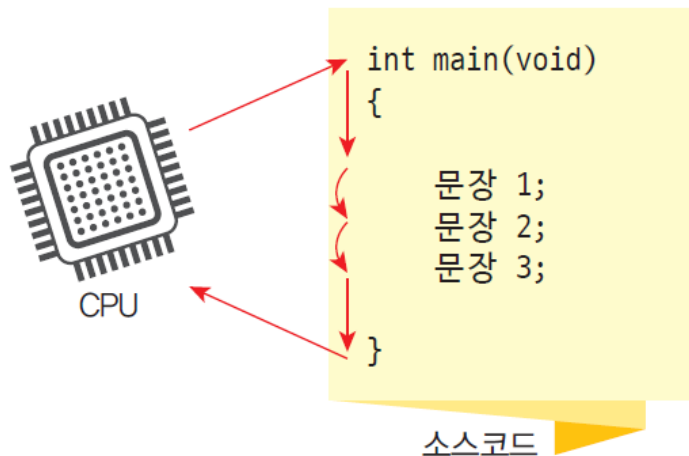
}

함수의 끝



문장(명령문)

- 함수는 여러 개의 문장으로 이루어진다.
- 문장들은 순차적으로 실행된다.
- 문장의 끝에는 반드시 ;이 있어야 한다.



소스 코드의 문장들은
기본적으로 차례대로
실행됩니다.



printf() 호출

- printf()는 컴파일러가 제공하는 함수로서 출력을 담당한다

printf("Hello World!");



Hello World!

- 큰따옴표 안의 문자열이 화면에 출력된다.



함수의 반환값

- return은 함수의 결과값을 외부로 반환

return 0;

- 반환값은 0



장간 점검

1. 문장의 끝에 추가하여야 하는 기호는?
2. C프로그램에 반드시 있어야 하는 함수는?
3. printf()가 하는 기능은 무엇인가?





응용 프로그램 #1

- 다음과 같은 출력을 가지는 프로그램을 제작하여 보자.





첫 번째 버전

- 문장들은 순차적으로 실행된다는 사실 이용

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    printf("Hello World!");
```

```
    printf("Kim ChulSoo");
```

```
    return 0;
```

```
}
```

2개의 문장은 순차적으로
실행된다

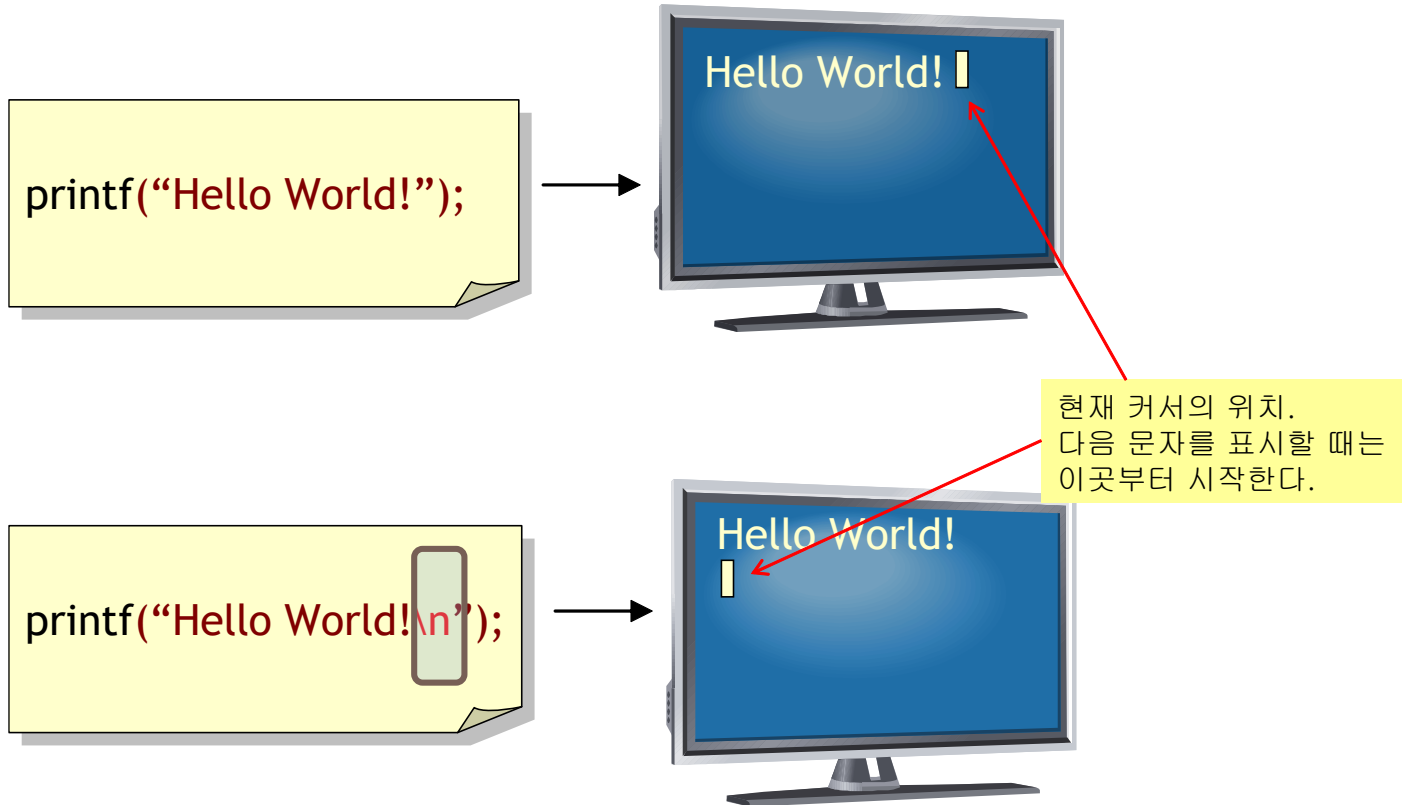


Hello World! Kim ChulSoo



줄바꿈 문자

- 줄바꿈 문자인 `\n`은 화면에서 커서는 다음줄로 이동하게 한다.





잘바꿈 문자 2개를 사용하면?

```
printf("Hello \nWorld! \n");
```





변경된 프로그램

- 줄바꿈 문자를 추가하면 우리가 원하던 결과가 된다.



```
#include <stdio.h>

int main(void)
{

    printf("Hello World!\n");
    printf("Kim ChulSoo \n");

    return 0;
}
```





주석(comment)

```
/* 두 개의 숫자의 합을 계산하는 프로그램 */
```

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int x; // 첫 번째 정수를 저장할 변수
```

```
    int y; // 두 번째 정수를 저장할 변수
```

```
    int sum; // 두 정수의 합을 저장하는 변수
```

```
    x = 100;
```

```
    y = 200;
```

```
    sum = x + y;
```

```
    printf("두수의 합 : % d", sum);
```

```
    return 0;
```

```
}
```

주석은 코드를
설명하는 글입니다.





3가지 방법의 주석

- /* 한 줄로 된 주석 */
- /* -----
저자: 홍길동
날짜: 2020.3.4
여러 줄로 이루어진 주석
----- */
- // 여기서부터 줄의 끝까지 주석



주석의 중요성

- 다른 사람이 프로그램을 보았을 때, 주석이 있다면 훨씬 쉽게 프로그램의 내용을 알 수 있다. 많은 시간이 흘렀다면, 만든 사람이라고 하더라도 내용을 잘 기억할 수 없다.
- 좋은 주석은 코드를 반복하거나 코드를 설명하지 않는 것이다. 주석에는 코드를 작성한 의도를 명확히 나타내어야 한다.



들여쓰기

- 들여쓰기(indentation)
 - 어느 부분에 속한 부분들을 나타냄

```
#include <stdio.h>
```

빈줄을 넣어서 의미별로 구별을 한다.

```
int main(void)
```

```
{
```

```
    int x;
```

```
    int y;
```

```
    int sum;
```

// 첫 번째 정수를 저장할 변수

// 두 번째 정수를 저장할 변수

// 두 정수의 합을 저장하는 변수

프로그램의 의도를
주석으로 설명한다.

```
    ...
```

같은 내용의 처리이면 들여쓰기를 한다.

```
    return 0;
```

```
}
```



주석과 들여 쓰기가 어렵다면..

```
#include <stdio.h>
int main(void) {
    int x; int y; int sum;    x = 100; y = 200; sum = x
    + y;    printf("두수의 합: %d", sum); return 0;
}
```

실행은 되지만 무슨 처리를 하고 있는
프로그램인지 알기가 힘들고 또한 들여
여쓰기가 안 되어 있어서 같은 수준
에 있는 문자들을 구분하기 힘듭니다.





함수

- 함수(function): 특정 기능을 수행하는 처리 단계들을 괄호로 묶어서 이름을 붙인 것
- 함수는 프로그램을 구성하는 기본적인 단위(부품)
- 코드 안에 () 가 들어가 있는것은 모두 함수

```
#include <stdio.h>
```

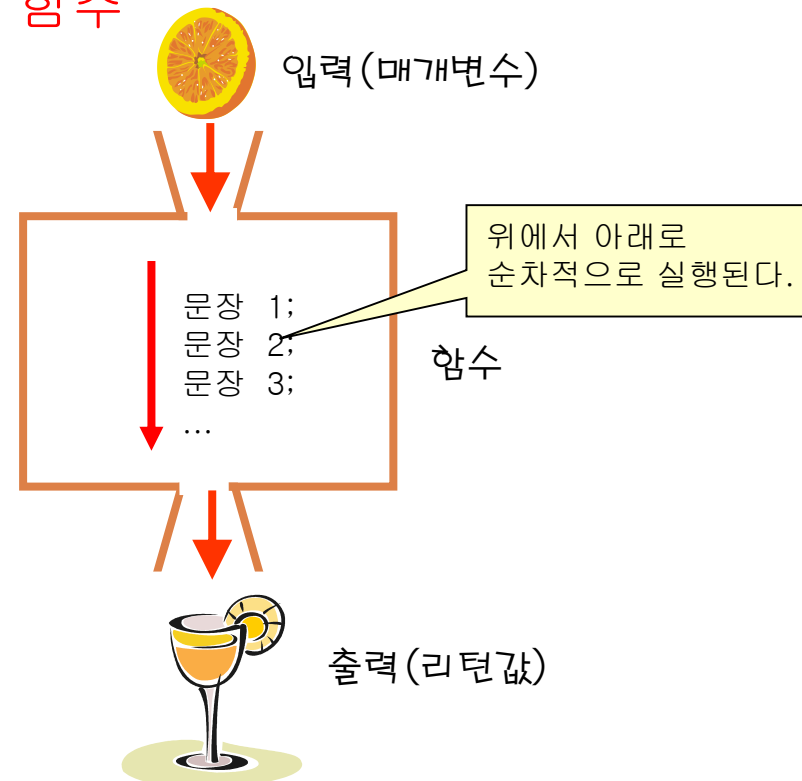
```
int main(void) {
```

```
    ...
```

```
    ...
```

```
    return 0;
```

```
}
```





함수의 구조

