Computer Engineering Programming 2

- 제1장 프로그래밍의 개념
- 제2장 프로그램 작성 방법
- Linux 환경에서의 프로그래밍

Lecturer: JUNBEOM YOO
jbyoo@konkuk.ac.kr

※ 본 강의자료는 생능출판사의 "PPT 강의자료"를 기반으로 제작되었습니다.

제1장 프로그래밍의 개념

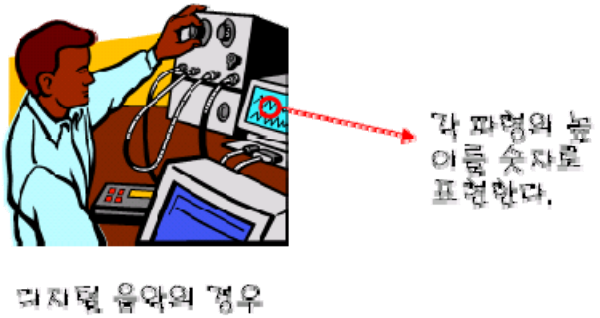
컴퓨터란?

Q) 컴퓨터(computer)는 무엇인가?

A) 컴퓨터는 기본적으로 계산(compute)하는 기계

Q) 컴퓨터를 이용하여 데이터를 처리하려면 반드시 데이터가 숫자 형태이어야 한다. 왜?

A) 컴퓨터는 숫자 계산을 하기 때문에 데이터는 숫자로 표시되어야 한다.



디지털과 아날로그

Q) 디지털은 무엇인가?

A) 모든 데이터를 숫자 형태로 표시하는 것
데이터를 표현, 처리, 저장, 전송하는데 숫자를 사용

Q) 아날로그는 무엇인가?

A) 연속적인 값으로 데이터를 표현
데이프와 같은 아날로그 매체에 데이터를 저장

오디오 테이프는 아날로그

mp3는 디지털

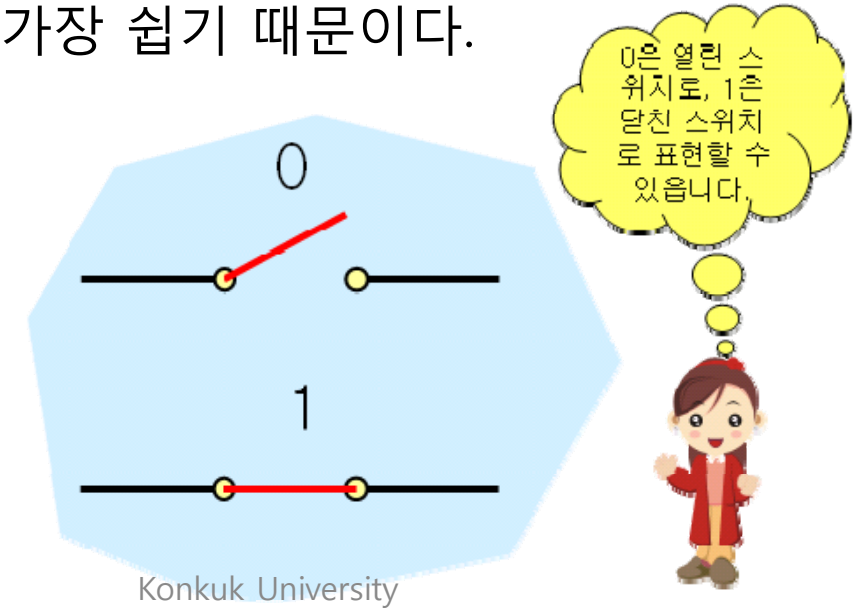
컴퓨터의 계산 능력

Q) 컴퓨터가 계산을 빠르게 할 수 있는 이유는?

A) 전자 회로를 이용하여 계산을 하기 때문이다.

Q) 전자 회로를 이용하여 계산을 하려면 어떤 진법을 사용?

A) 이진수를 사용하여야 한다. 이진수가 전자 회로로 나타내기가 가장 쉽기 때문이다.



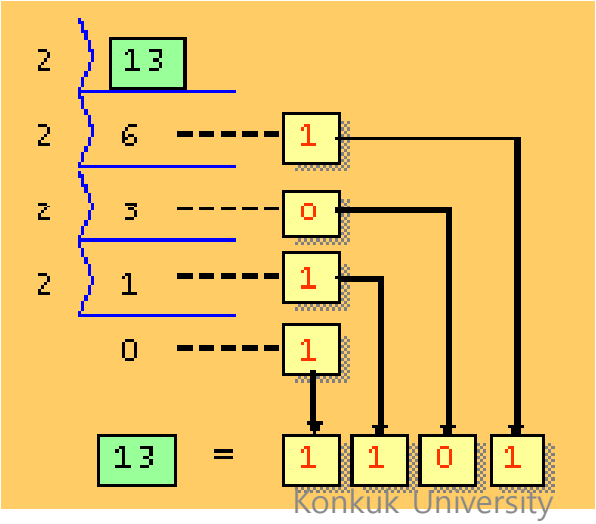
이진수

Q) 이진수는 십진수와 무엇이 다른가?

A) 이진수는 0과 1로만 구성되어 있다.

Q) 십진수를 이진수로 바꾸려면?

A) 십진수를 이진수로 바꾸려면 십진수를 2로 나누고 나머지를 기록하는 작업을 몫이 0이 될 때까지 되풀이하면 된다.



비트와 바이트

Q) 비트(bit)란 무엇인가?

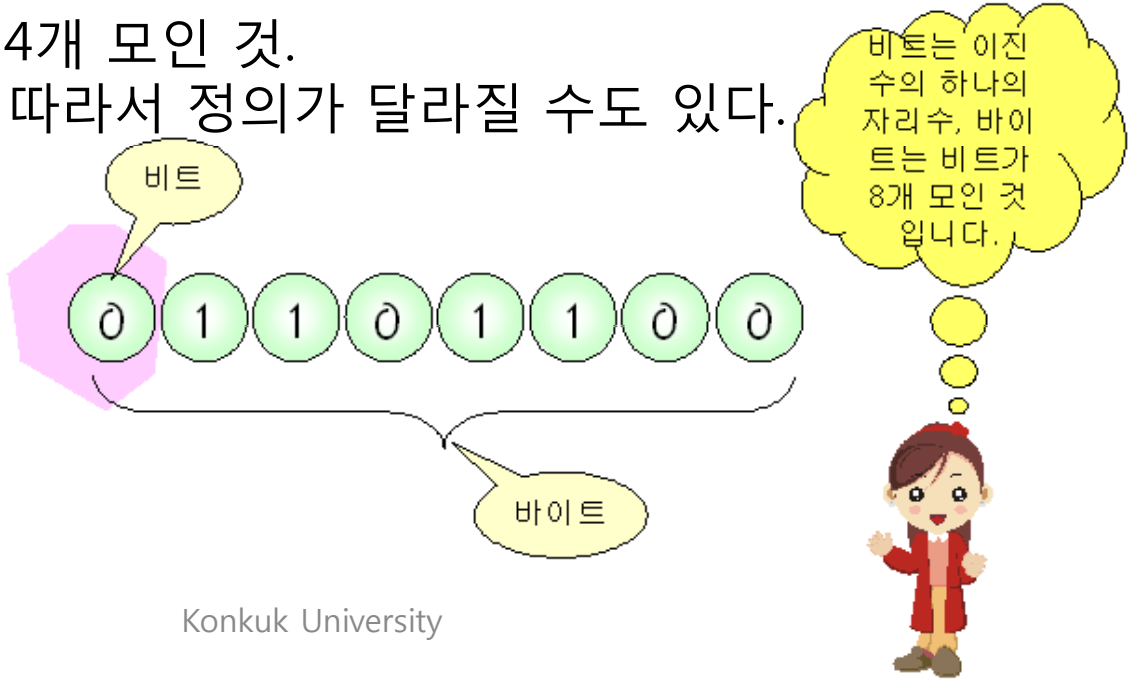
A) 이진수의 하나의 자리수를 의미한다.
바이너리 디지트(binary digit)의 약자이다.

Q) 바이트(byte)란 무엇인가?

A) 8개의 비트를 모은 것을 바이트라고 한다.

Q) 워드(word)란 무엇인가?

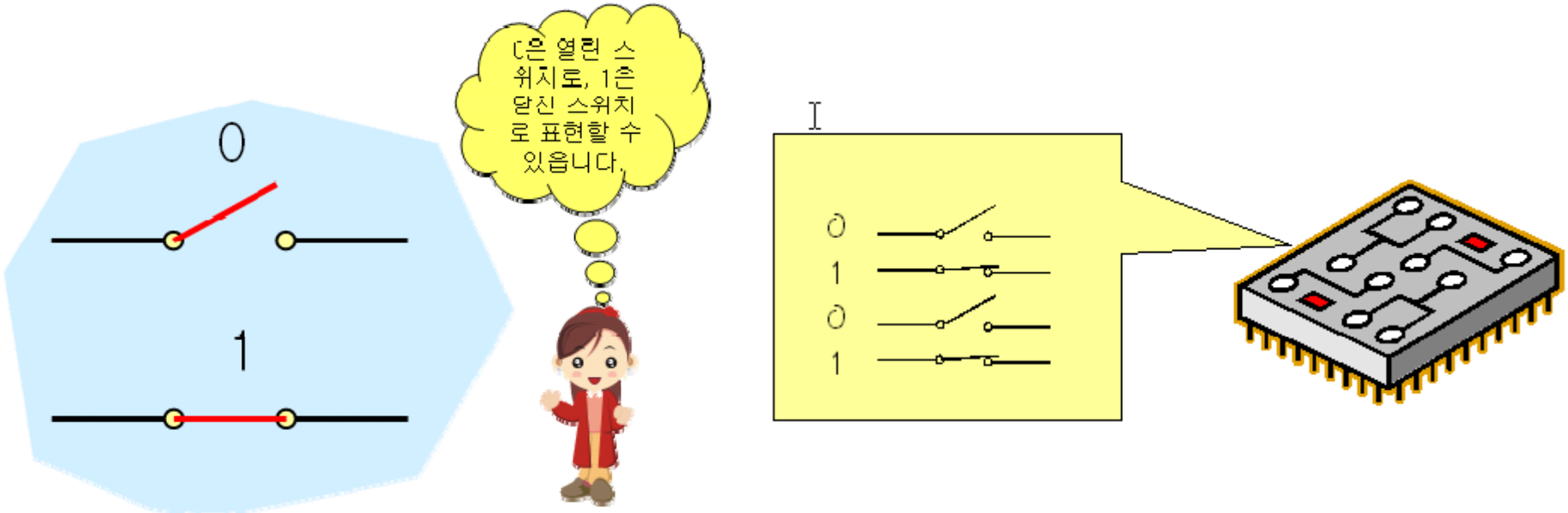
A) 바이트가 4개 모인 것.
시스템에 따라서 정의가 달라질 수도 있다.



비트의 표현

Q) 비트는 컴퓨터 내부에서 구체적으로 어떻게 표현되는가?

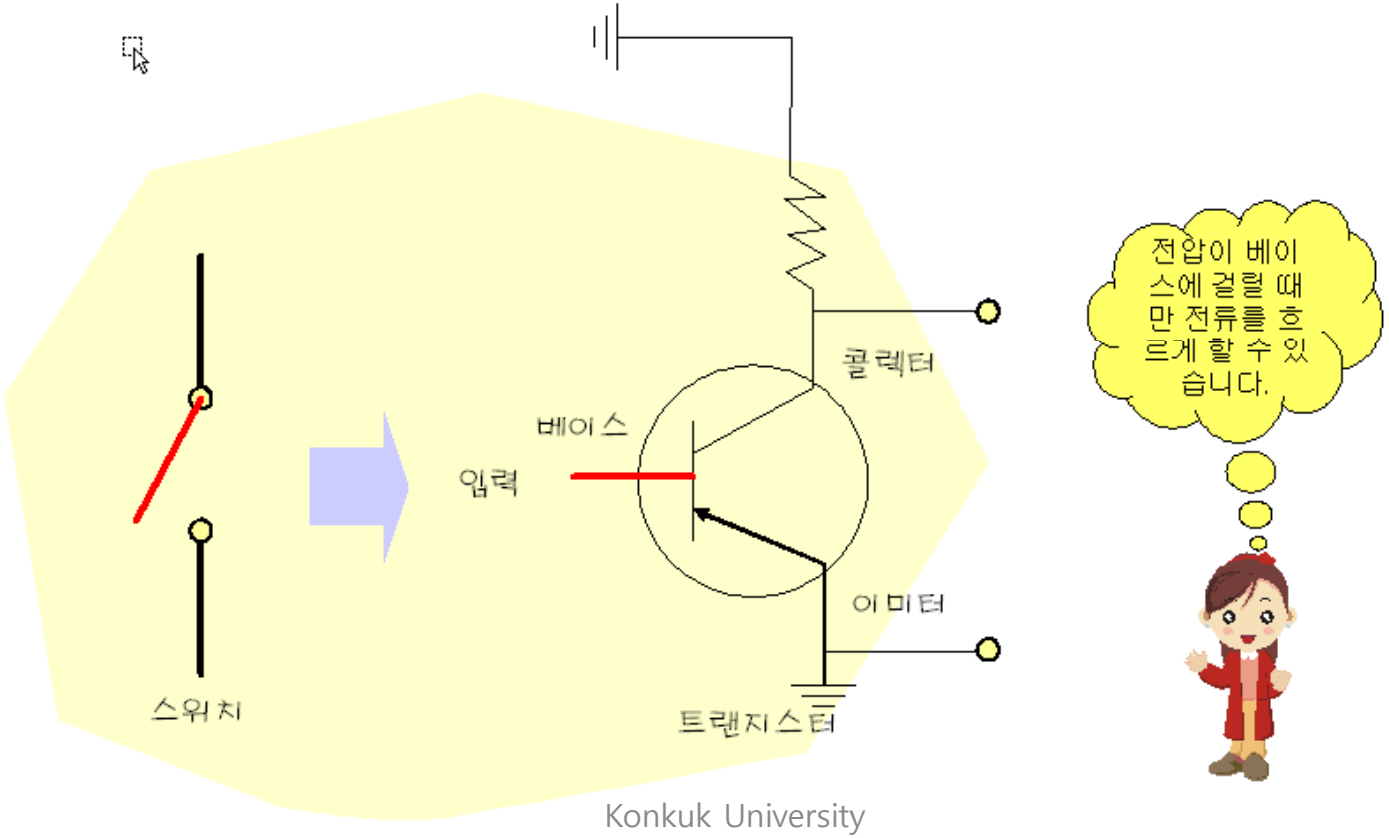
A) 이진수의 0은 열린 스위치로, 이진수의 1은 닫친 스위치로 표현된다.



스위치의 구현

Q) 스위치는 컴퓨터 내부에서 구체적으로 어떻게 구현되는가?

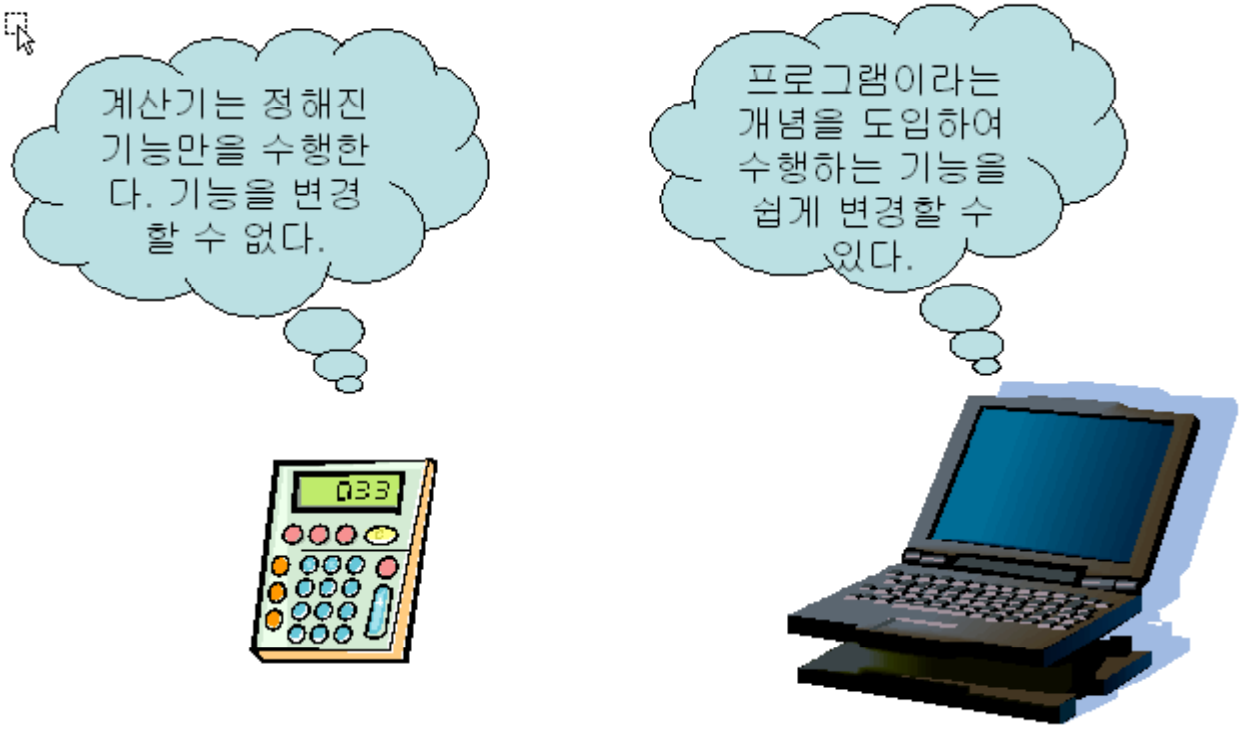
A) 하나의 트랜지스터는 소형 전자 스위치의 역할을 한다.



컴퓨터의 정의

Q) 그렇다면 계산만 빠르게 할 수 있으면 컴퓨터인가?

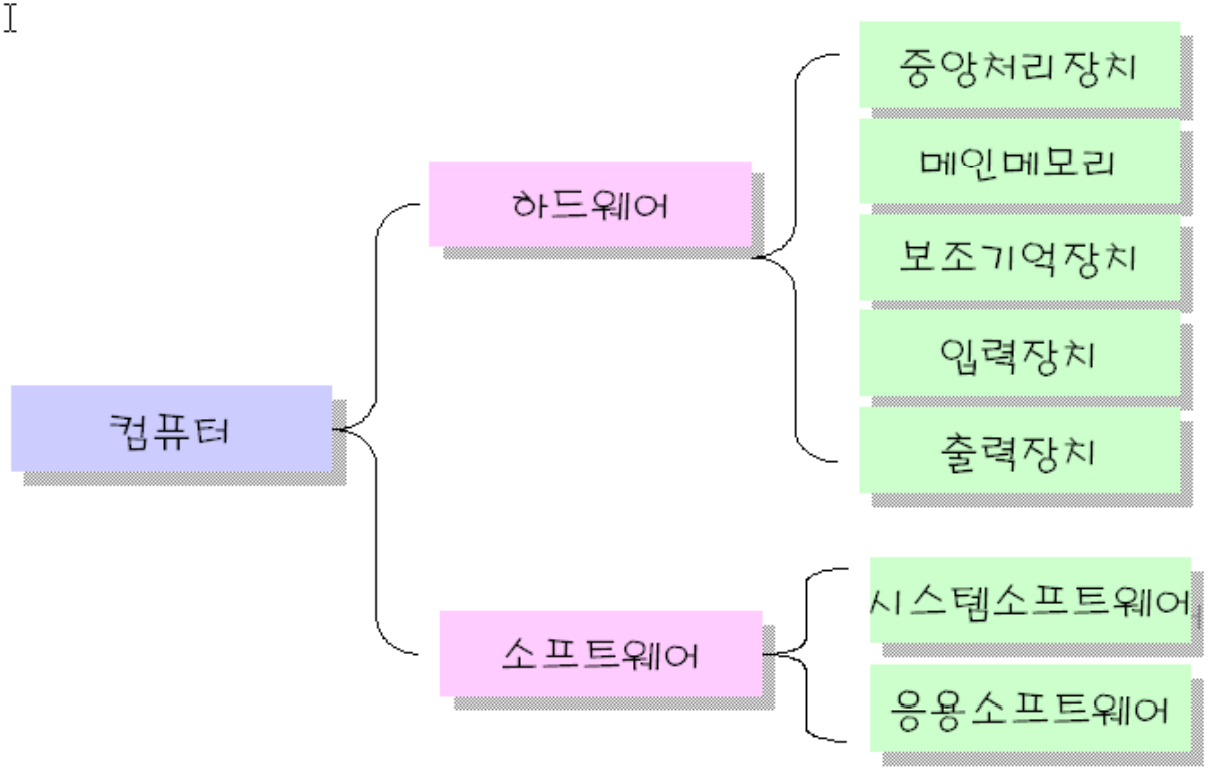
A) 현대적인 의미에서의 컴퓨터는 명령어들의 리스트에 따라 데이터를 처리하는 기계라고 할 수 있다



컴퓨터의 구성 요소

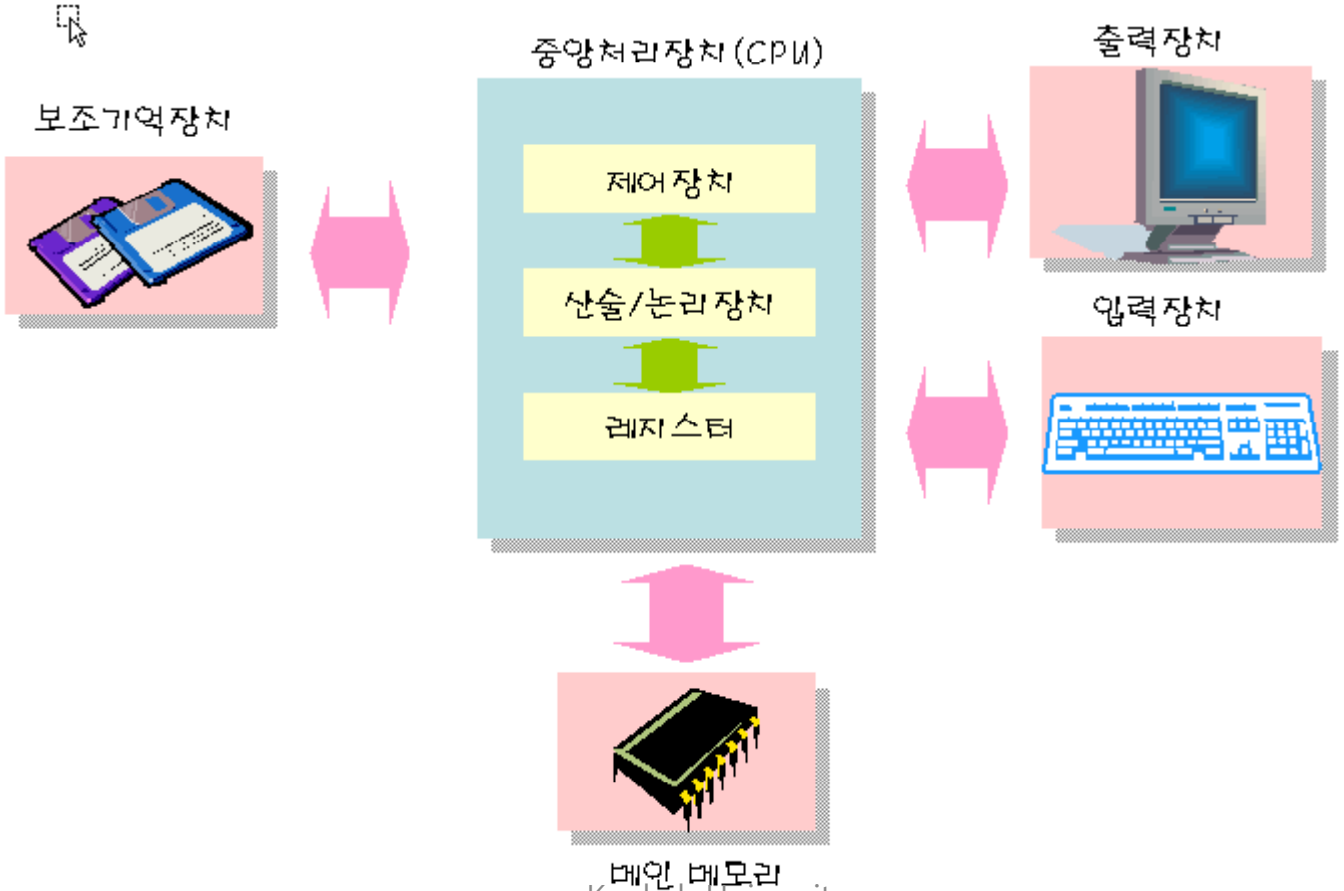
Q) 컴퓨터의 구성 요소를 크게 2가지로 분류하면?

A) 컴퓨터는 기본적으로 하드웨어와 소프트웨어로 구분



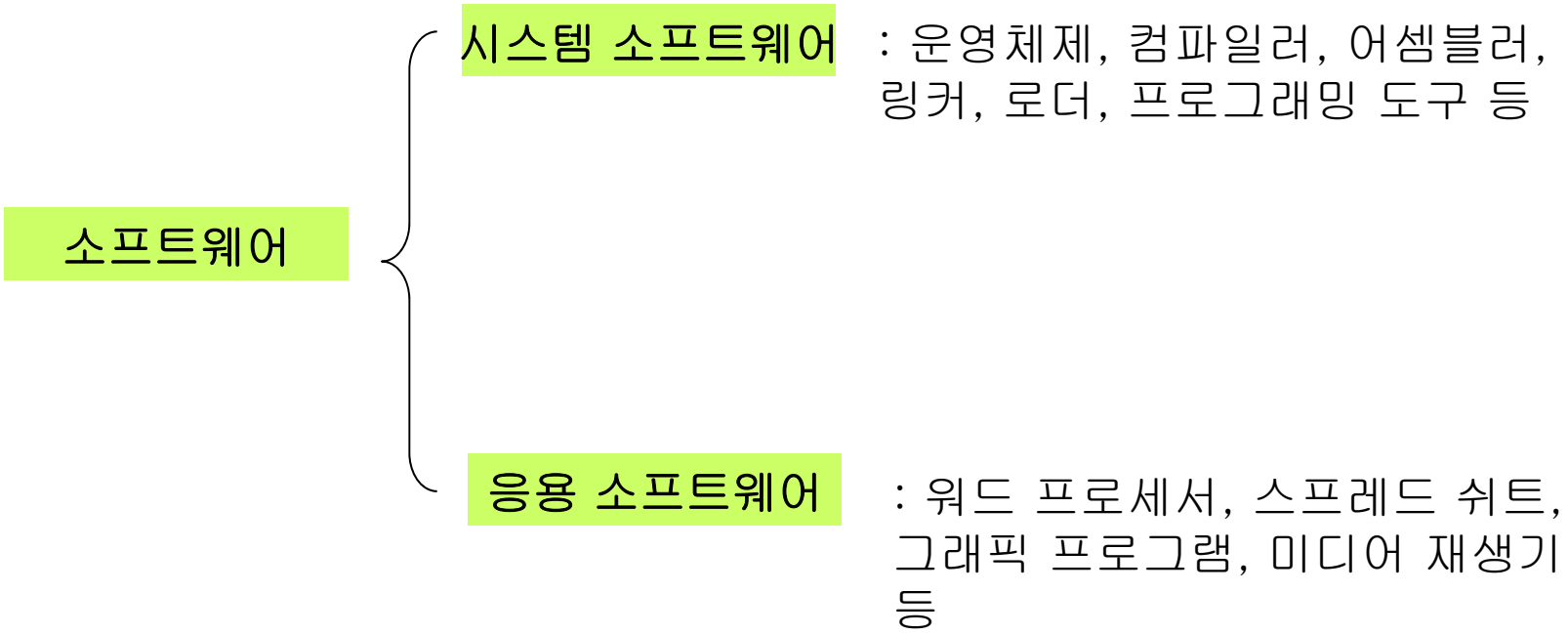
하드웨어

Q) 하드웨어는 어떻게 분류할 수 있는가?



소프트웨어

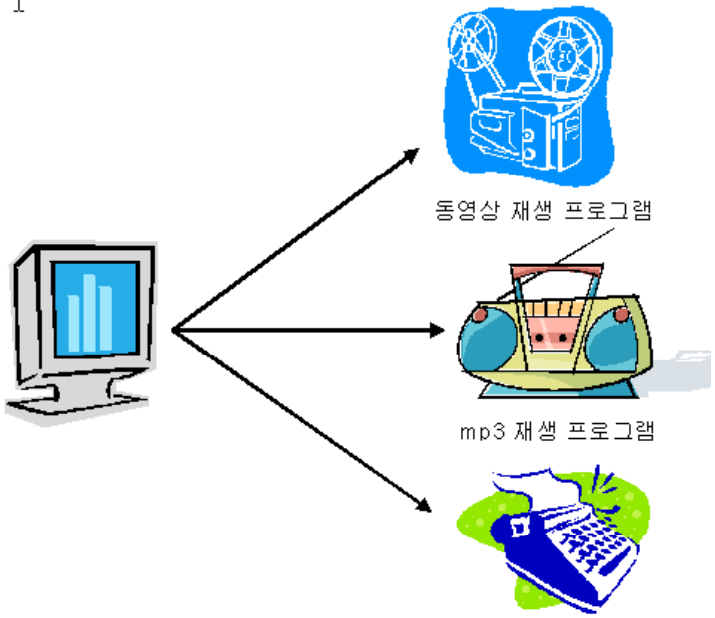
Q) 소프트웨어는 어떻게 분류할 수 있는가?



프로그램

Q) 왜 컴퓨터에서는 가전제품처럼 프로그램 설치 없이 바로 동작되도록 하지 않고 불편하게 사용자가 프로그램을 설치하게 하였을까 ?

A) 컴퓨터를 범용적인 기계로 만들기 위해서이다. 컴퓨터는 프로그래만 바꾸어주면 다양한 작업을 할 수 있다.

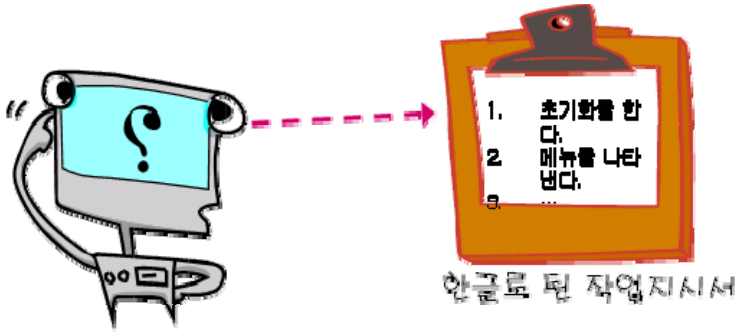


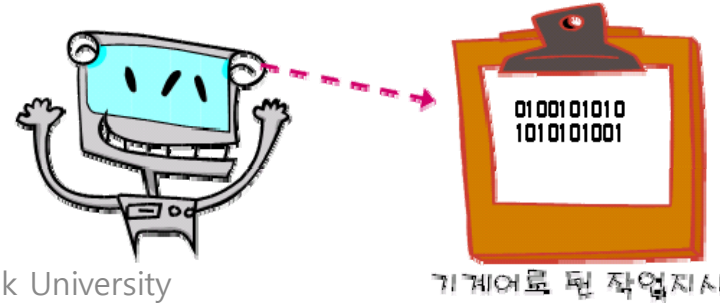
기계어

Q) 컴퓨터가 이해할 수 있는 언어는 어떤 것인가?

A) 컴퓨터가 알아듣는 언어는 한가지이다. 즉 0과 1로 구성되어 있는 "001101110001010..."과 같은 기계어이다.

A) 컴퓨터는 모든 것을 0과 1로 표현하고 0과 1에 의하여 내부 스위치 회로들이 ON/OFF 상태로 변경되면서 작업을 한다.

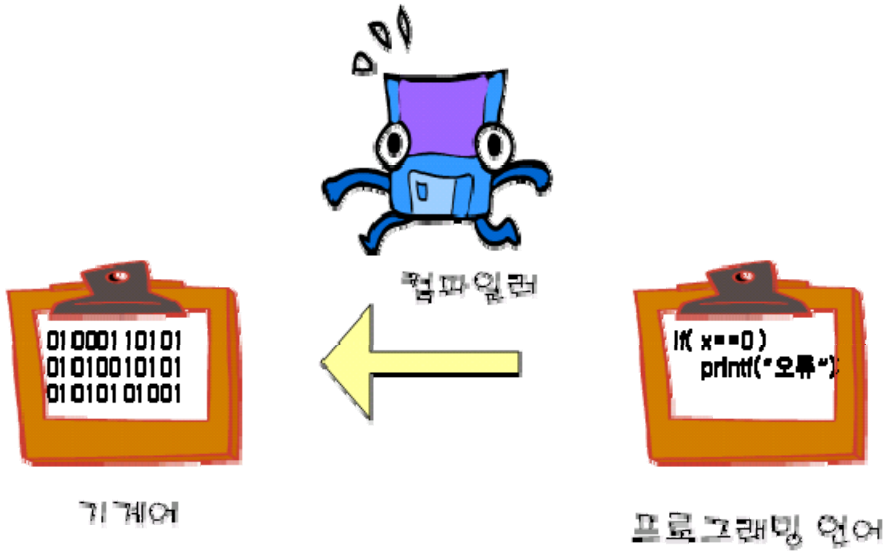




프로그래밍 언어의 필요성

Q) 그렇다면 인간이 기계어를 사용하면 어떤가?

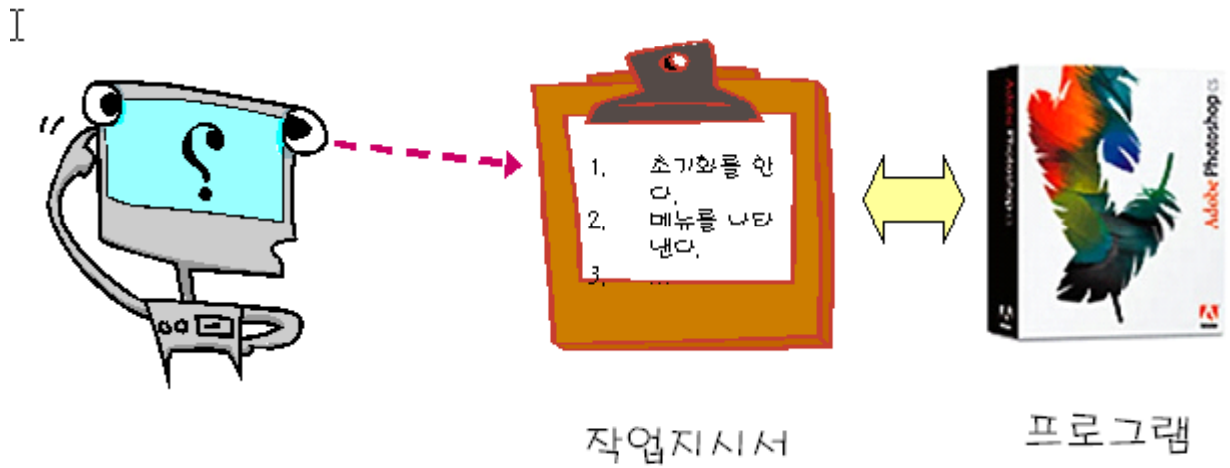
- 기계어를 사용할 수는 있으나 이진수로 프로그램을 작성하여야 하기 때문에 아주 불편하다.
- 프로그래밍 언어는 자연어와 기계어 중간쯤에 위치
- 컴파일러가 프로그래밍 언어를 기계어로 통역



프로그램의 역할

Q) 컴퓨터에서 프로그램이 하는 일은 무엇인가?

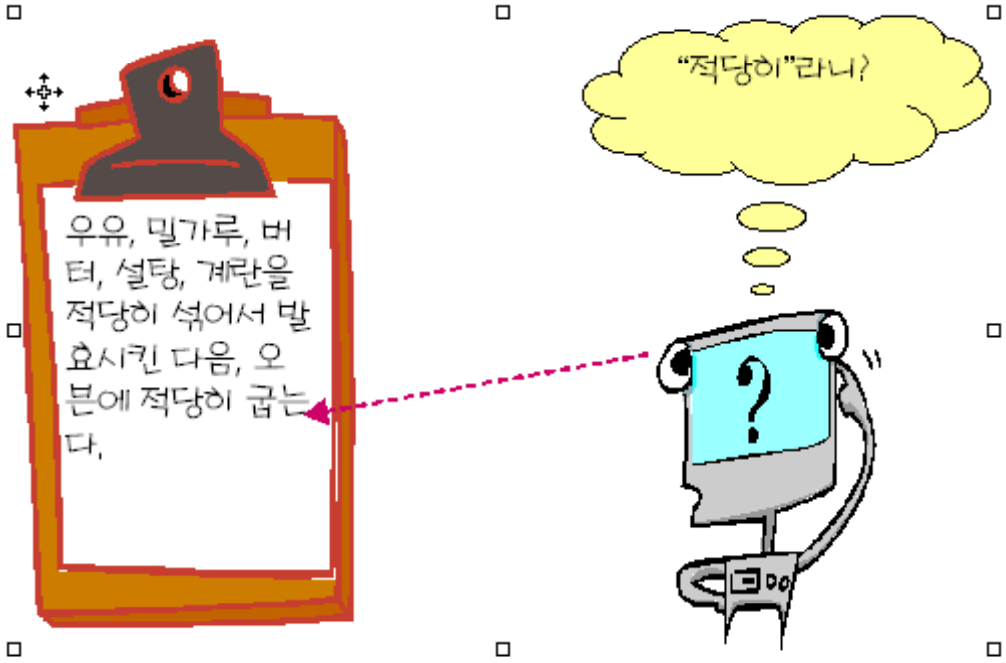
A) 프로그램이란 우리가 하고자 하는 작업을 컴퓨터에게 전달하여 주는 역할을 한다.

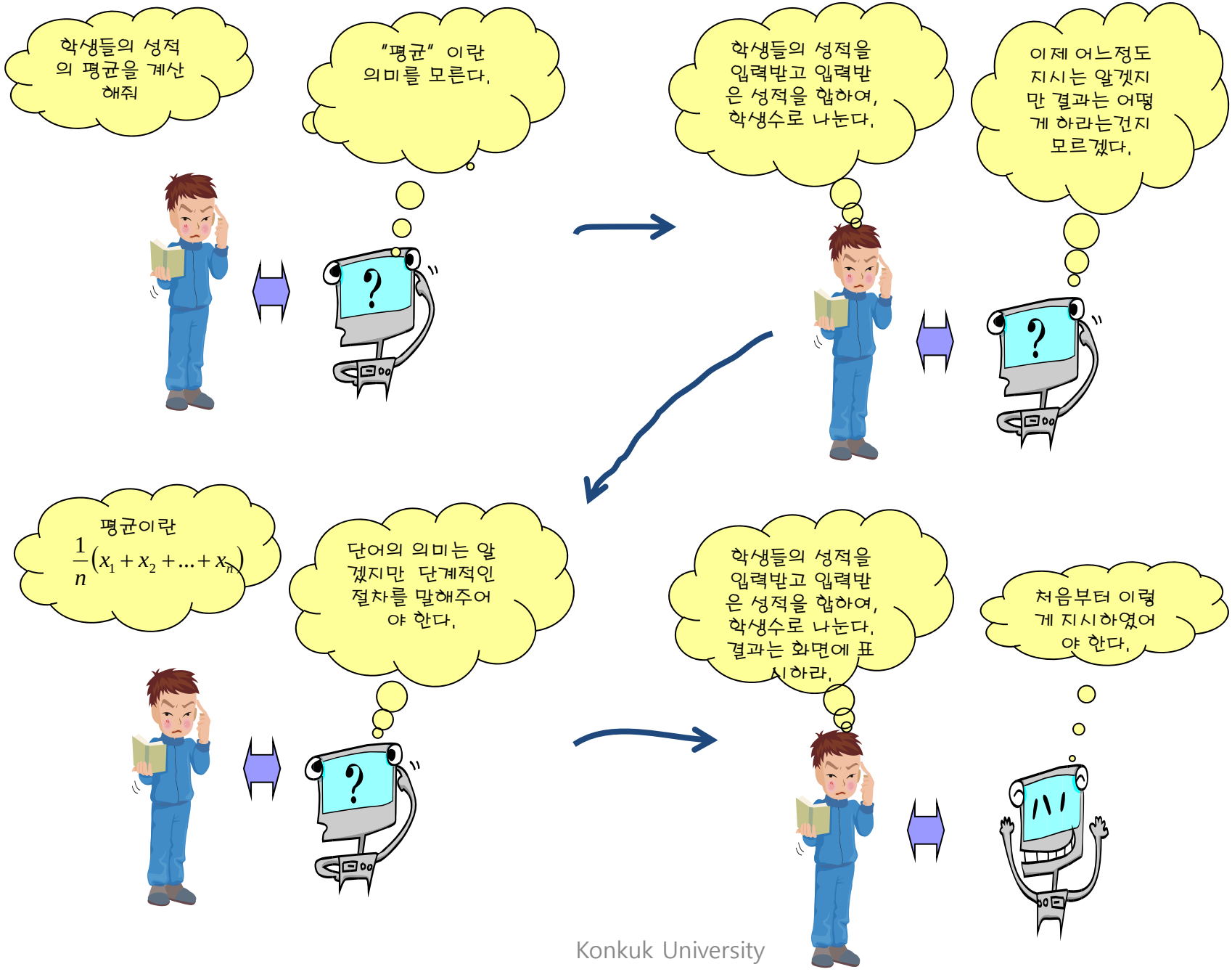


작업을 지시하는 방법

Q) 컴퓨터에게 적당히 작업을 시킬 수 있을까?

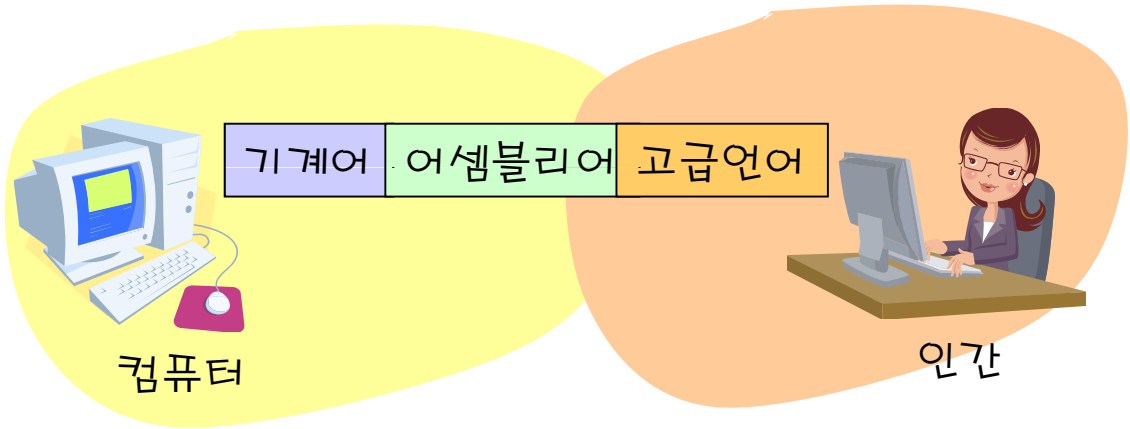
A) 상식이나 지능이 없기 때문에 아주 자세하고 구체적으로 일을 지시하여야 한다.




학생들의 성적의 평균을 계산해줘
"평균" 이란 의미를 모른다.
학생들의 성적을 입력받고 입력받은 성적을 합하여, 학생수로 나눈다.
이제 어느정도 지시는 알겟지만 결과는 어떻게 하라는건지 모르겠다.
평균이란 $\frac{1}{n}(x_1 + x_2 + \ldots + x_n)$
단어의 의미는 알겠지만 단계적인 절차를 말해주어야 한다.
학생들의 성적을 입력받고 입력받은 성적을 합하여, 학생수로 나눈다, 결과는 화면에 표시하라.
처음부터 이렇게 지시하였어야 한다.

프로그래밍 언어의 분류

- 기계어(machine language)
- 어셈블리어(assembly language)
- 고급 언어(high-level language)



기계어

- 특정 컴퓨터의 명령어(instruction)를 이진수로 표시한 것
- 0과 1로 구성
- 하드웨어에 종속

```
00001111 10111111 01000101 11111000
00001111 10111111 01001101 11111000
00000011 10100001
01100110 10001001 01000101 11111010
```

어셈블리어

- CPU의 명령어들을 이진수가 아닌 영어의 약자인 기호로 표기
- 기계어보다는 더 높은 수준에서 프로그램을 작성하는 것을 가능
- 기호와 CPU의 명령어가 일대일 대응
- 어셈블러(assembler): 기호를 이진수로 변환하는 프로그램

```
MOV AX, MIDSCORE
MOV CX, FINALSCORE
ADD AX CX
MOV TOTALSCORE, AX
```

고급언어

- 특정한 컴퓨터의 구조나 프로세서에 무관하게, 독립적으로 프로그램을 작성할 수 있는 언어
- C, C++, JAVA, FORTRAN, PASCAL
- 컴파일러: 고급 언어 문장을 기계어로 변환하는 프로그램

```
TotalScore = MidScore + FinalScore;
```

C

- 1970년대 초 AT&T의 Dennis Ritchie 에 의하여 개발
- B언어->C언어
- UNIX 운영 체제 개발에 필요해서 만들어짐
- 처음부터 전문가용 언어로 출발

Ken Thomson과 Dennis Ritchie가 클린턴 대통령으로부터 National Medal of Technology상을 받는 장면

C언어의 버전

- K & R C
  - 1978년 “C Programming Language” 책 출간
  - 비공식적인 명세서 역할
- ANSI C
  - 1983년 ANSI(American National Standards Institute)는 X3J11이라는 위원회에 의한 표준
- C99
  - 1999년에 ISO에 의한 표준
  - C++에서 사용되는 특징 추가
  - 아직 마이크로소프트는 지원하지 않음(이유: C++에 집중)

C언어의 특징

- 간결하다.
- 효율적이다.
- C 언어는 하드웨어를 직접 제어하는 하는 저수준의 프로그래밍도 가능하고 고수준의 프로그래밍도 가능하다.
- C언어는 이식성이 뛰어나다.
- 초보자가 배우기가 어렵다.

C언어의 특징

C 언어에는 꼭 필요한 기능만이 들어 있고 모든 표기법도 아주 간결하게 되어 있다.

간결하다.

C로 작성된 프로그램이 크기가 작으며 실행 속도가 빠르고 메모리를 효과적으로 사용한다. C언어는 거의 어셈블리 언어 수준의 효율성을 가진다.

효율적이다.

항상 모든 자유에는 책임이 따르듯이 하드웨어를 제어하기 위하여 꼭 필요한 요소인 포인터 등을 잘못 사용하는 경우가 많다.

배우기는 어렵다.

C language

이식성이 뛰어나다.

저수준과 고수준이 모두 가능하다.

한번 작성된 프로그램을 다른 CPU를 가지는 하드웨어로 쉽게 이식할 수 있다. PC에서 개발된 프로그램도 컴파일만 다시 하면 슈퍼 컴퓨터에서 수행시킬 수 있다.

C 언어는 운영체제를 만들었던 언어이니 만큼, 어셈블리 언어 만큼의 구체적인 하드웨어 제어가 가능하다. 반면에 하향식(top-down) 설계, 구조화 프로그래밍, 모듈화 설계 등의 소프트웨어 공학의 다양한 기법들을 적용할 수 있다.

C언어의 미래

Q) 앞으로도 C언어는 사용될 것인가?

- C언어는 C++와 JAVA의 공통적인 부분이다.
- 임베디드 시스템에서는 C언어가 많이 사용된다.

엠베디드 시스템: 엠베디드 시스템이란 특수 목적의 시스템으로 컴퓨터가 장치 안에 MP3 플레이어, 핸드폰등이 여기에 속한다.

알고리즘

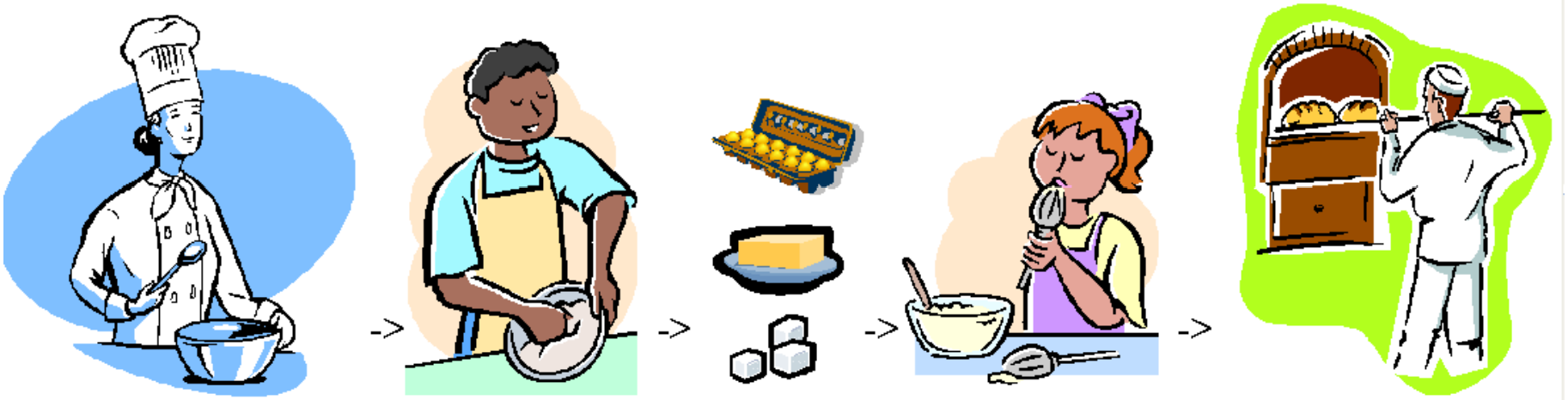
Q) 오븐의 사용법만 배우고 음식 재료만 있으면 누구나 요리가 가능한가?

A) 요리법을 알아야 한다.

- 프로그램이 요리와 같다면 알고리즘은 요리법에 해당한다.
- 알고리즘(algorithm): 문제를 해결하는 절차(방법)

빵을 만드는 알고리즘

① 빈 그릇을 준비한다.
② 이스트를 밀가루, 우유에 넣고 저어준다.
③ 버터, 설탕, 계란을 추가로 넣고 섞는다.
④ 따뜻한 곳에 놓아두어 발효시킨다
⑤ 170~180도의 오븐에서 굽는다



1부터 10까지의 합을 구하는 알고리즘

① 1부터 10까지의 숫자를 직접 하나씩 더한다.

$1 + 2 + 3 + \ldots + 10 = 55$

·② 두수의 합이 10이 되도록 숫자들을 그룹핑하여 그룹의 개수에 10을 곱하고 남은 숫자 5를 더한다.

(0 + 10) = 10
(1 + 9) = 10
(2 + 8) = 10
(3 + 7) = 10
(4 + 6) = 10
5

10*5=50 + 5 = 55

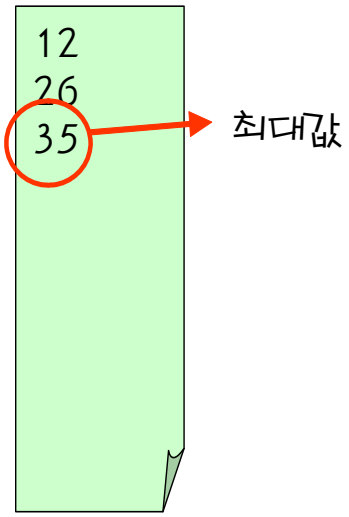
③ 공식을 이용하여 계산할 수도 있다.

$10*(1+10)/2 - 55$

알고리즘의 기술

1. 영어와 국어와 같은 자연어
2. 순서도(flowchart)
3. 의사 코드(pseudo-code)

- 예제: 숫자들의 리스트에서 최대값을 구하는 문제



노트

자연어

- 자연어 (natural language)는 인간이 사용하는 언어
- 단어들을 명백하게 정의해야 한다.

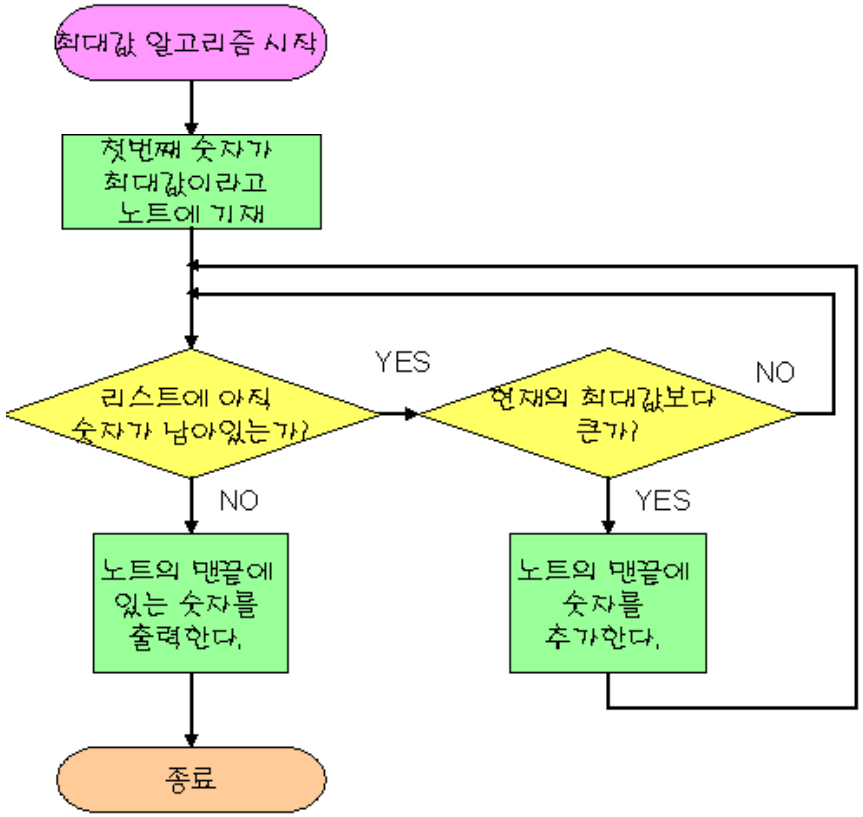
1. 리스트의 첫 번째 숫자가 가장 크다고 가정하자.

2. 리스트의 남아있는 숫자들이 하나씩 조사하여 현재의 최대값보다 크면 노트에 적는다.

3. 모든 숫자들을 전부 조사된 후에 노트에 가장 나중에 적힌 숫자가 최대값이 된다.

순서도

- 프로그램에서의 논리순서 또는 작업순서 등을 그래픽으로 표현하기 위한 형식
- 알고리즘이 복잡하면 기술하기가 힘들어진다.



의사 코드

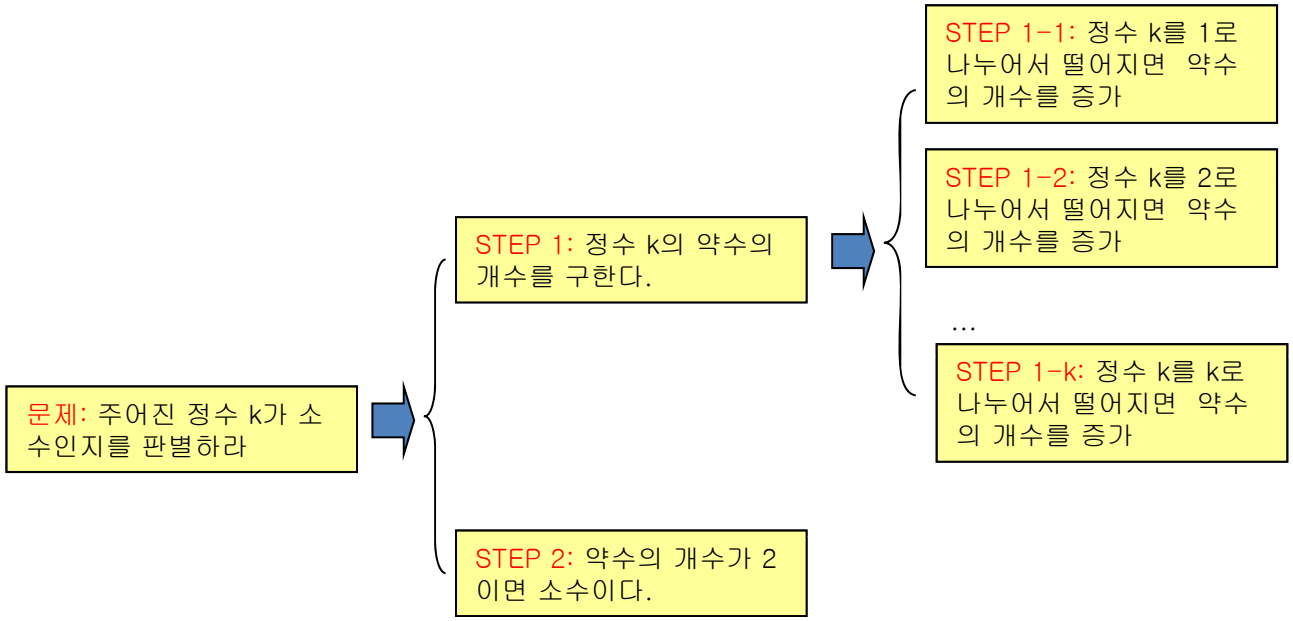
- 자연어보다는 더 체계적이고 프로그래밍 언어보다는 덜 엄격한 언어로서 알고리즘의 표현에 주로 사용되는 코드

```
알고리즘 GetLargest
  입력: 숫자들의 리스트 L.
  출력: 리스트에서 가장 큰 값


  largest ←L[0]
  for each n in L do
    if n > largest  then
      largest ← n
  return largest
```

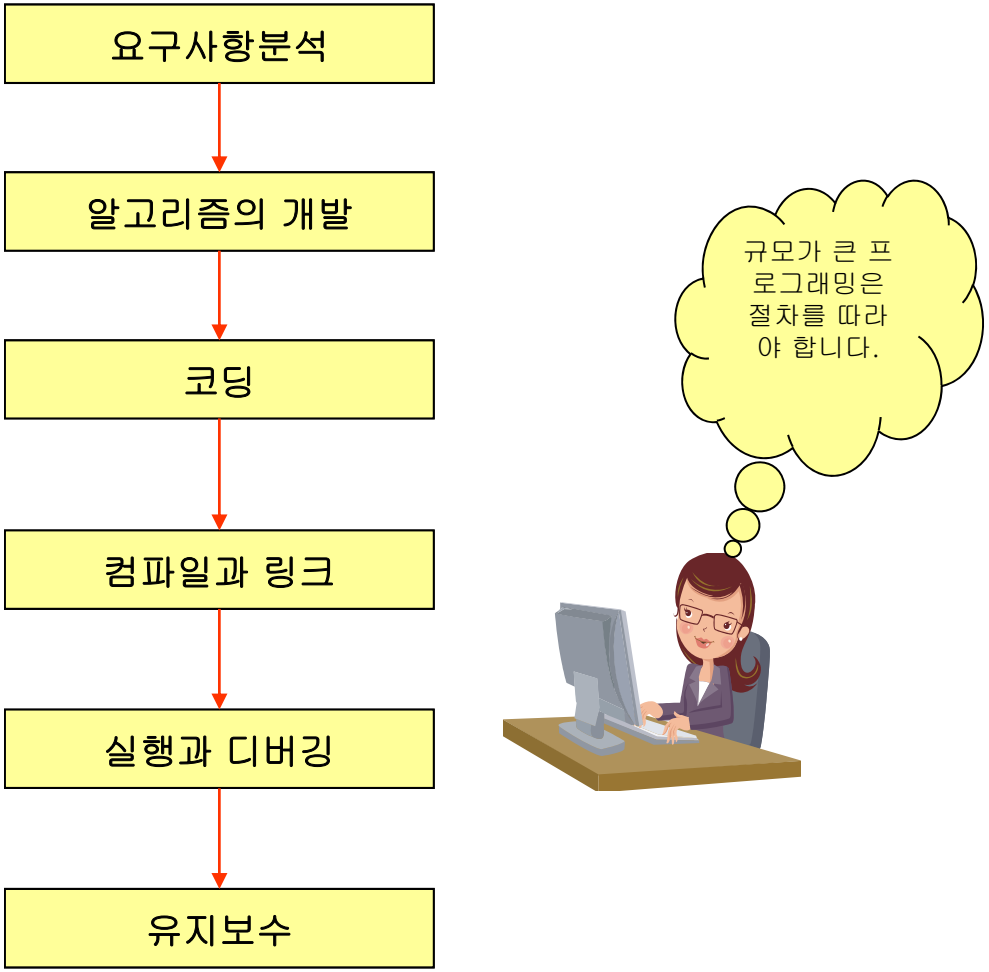
알고리즘을 만드는 방법

문제를 한 번에 해결하려고 하지 말고 더 작은 크기의 문제들로 분해한다.
문제가 충분히 작아질 때까지 계속해서 분해한다.

(예) 주어진 정수 k가 소수(prime)인가 아닌가를 판별하는 알고리즘을 만들어보자.



프로그램 개발 과정


요구사항분석
알고리즘의 개발
코딩
컴파일과 링크
실행과 디버깅
유지보수
규모가 큰 프로그래밍은 절차를 따라야 합니다.

요구 사항 분석

- 프로그래머는 사용자들의 요구사항을 만족시키기 위하여 프로그램을 작성
- (예) 3년 이상 근무한 직원들의 리스트 출력
  - 정규직만 or 계약직 포함
  - 기준이 되는 날짜가 오늘?
- 요구 사항 명세서: 사용자의 요구 조건을 만족하도록 소프트웨어가 갖는 기능 및 제약 조건, 성능 목표 등을 포함

알고리즘의 개발

- 핵심적인 부분
- 어떤 단계를 밟아서 어떤 순서로 작업을 처리할 것인지를 설계
- 순서도와 의사 코드를 도구로 사용
- 알고리즘은 프로그래밍 언어와는 무관
- 알고리즘은 원하는 결과를 얻기 위하여 밟아야 하는 단계에 집중적으로 초점을 맞추는 것

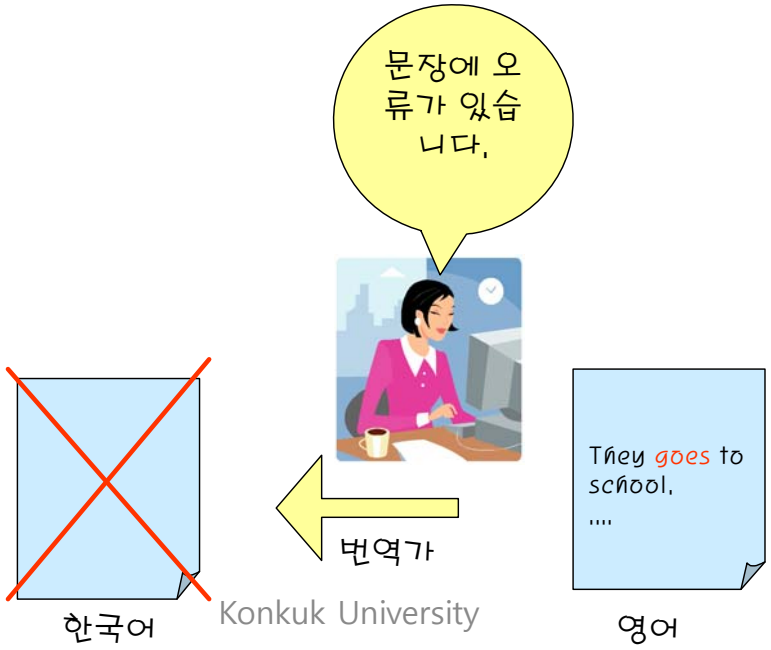
코딩

- 알고리즘의 각 단계를 프로그래밍 언어를 이용하여 기술
- 어떤 프로그래밍 언어로도 가능
- 알고리즘을 프로그래밍 언어의 문법에 맞추어 기술한 것을 *소스 프로그램(source program)*
- 소스 프로그램은 주로 텍스트 에디터나 통합 개발 환경을 이용하여 작성
- (Q) 알고리즘 개발과 코딩 중 어떤 것이 더 어려울까?

  (A) 알고리즘 개발이 더 창의적인 작업이고 더 어렵다

컴파일

- 소스 프로그램을 목적 프로그램으로 변환하는 작업
- 컴파일러가 수행
- 컴파일 오류: 문법 오류
  - (예) He go to school;
- 오류가 발생하면 소스 프로그램을 수정한 후에 다시 컴파일



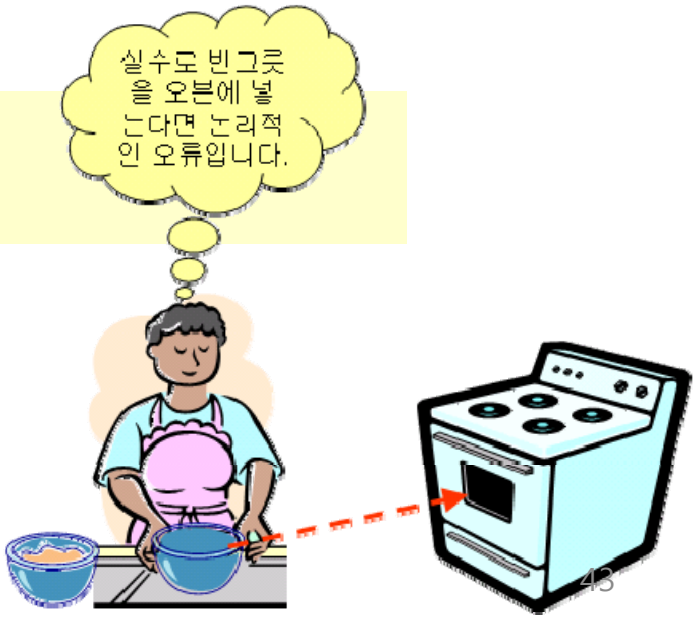
링크

- 컴파일된 목적 프로그램을 라이브러리와 연결하여 실행 프로그램을 작성하는 것
- *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.

실행 및 디버깅

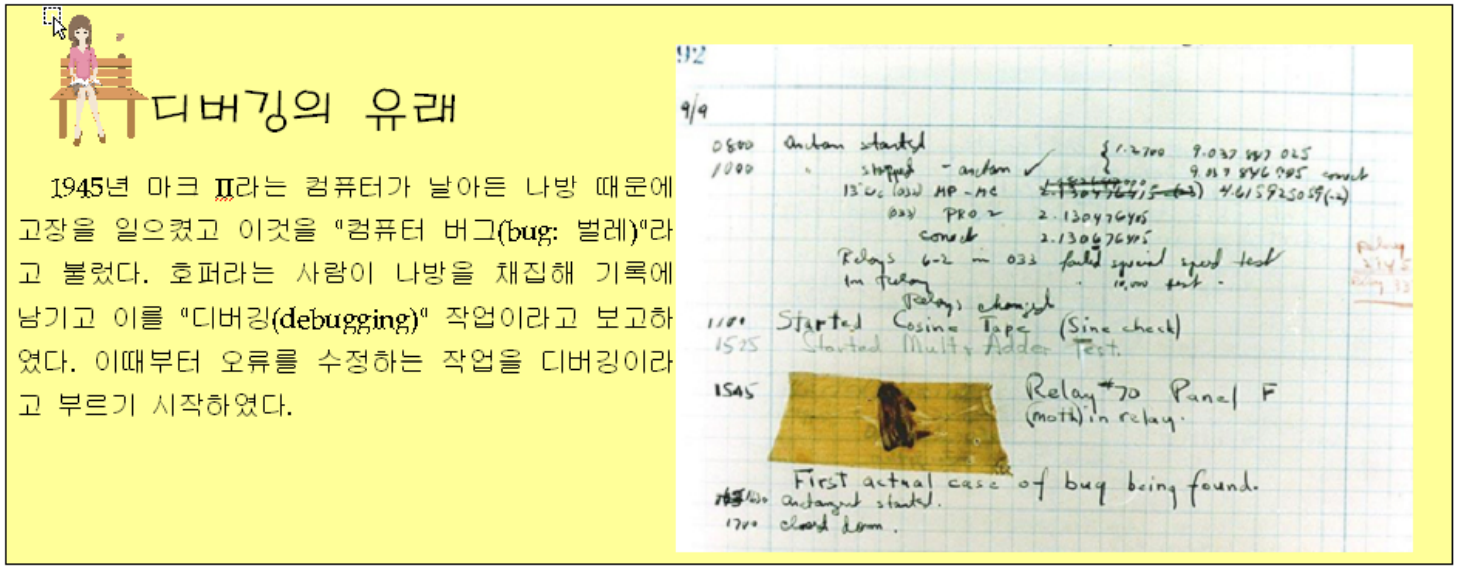
- 실행 파일: 실행 파일은 윈도우즈에서는 .exe라는 확장자
- 실행 시간 오류(run time error):
  - 0으로 나누는 것
  - 잘못된 메모리 주소에 접근하는 것
- 논리 오류(logical error):
  - 문법은 틀리지 않았으나 논리적으로 정확하지 않는 것
  - (예)

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



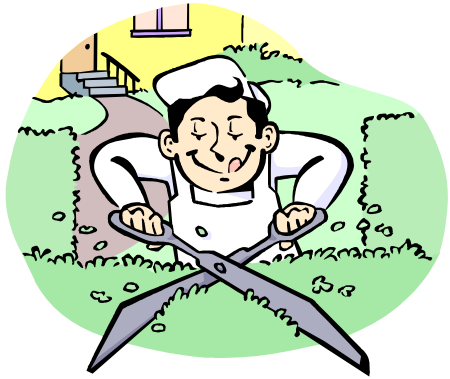
디버깅

- 소스에 존재하는 오류를 잡는 것

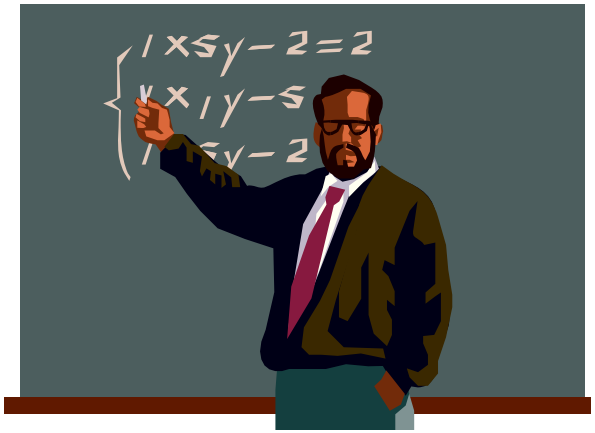


소프트웨어의 유지 보수

- 소프트웨어의 유지 보수가 필요한 이유
  - 디버깅 후에도 버그가 남아 있을 수 있기 때문
  - 소프트웨어가 개발된 다음에 사용자의 요구가 추가될 수 있기 때문
- 유지 보수 비용이 전체 비용의 50% 이상을 차지

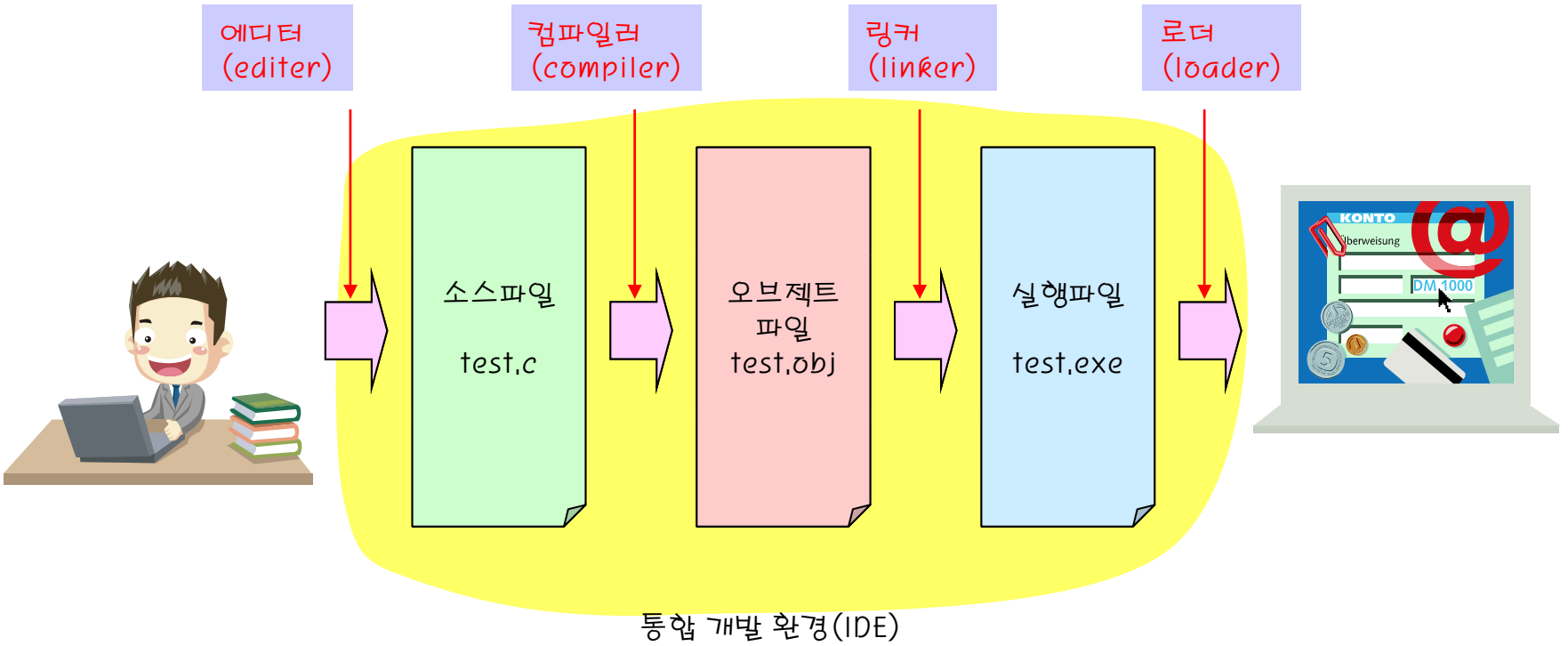


Q & A



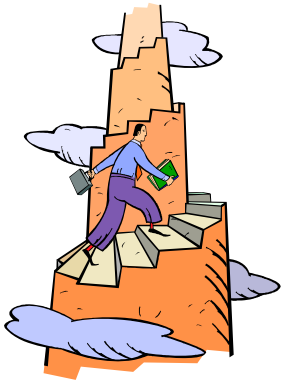
제2장 프로그램 작성 방법

프로그램 작성 과정


에디터
(editer)
컴파일러
(compiler)
링커
(linker)
로더
(loader)
소스파일
test.c
오브젝트
파일
test.obj
실행파일
test.exe
KONTO
DM 1000
통합 개발 환경(IDE)

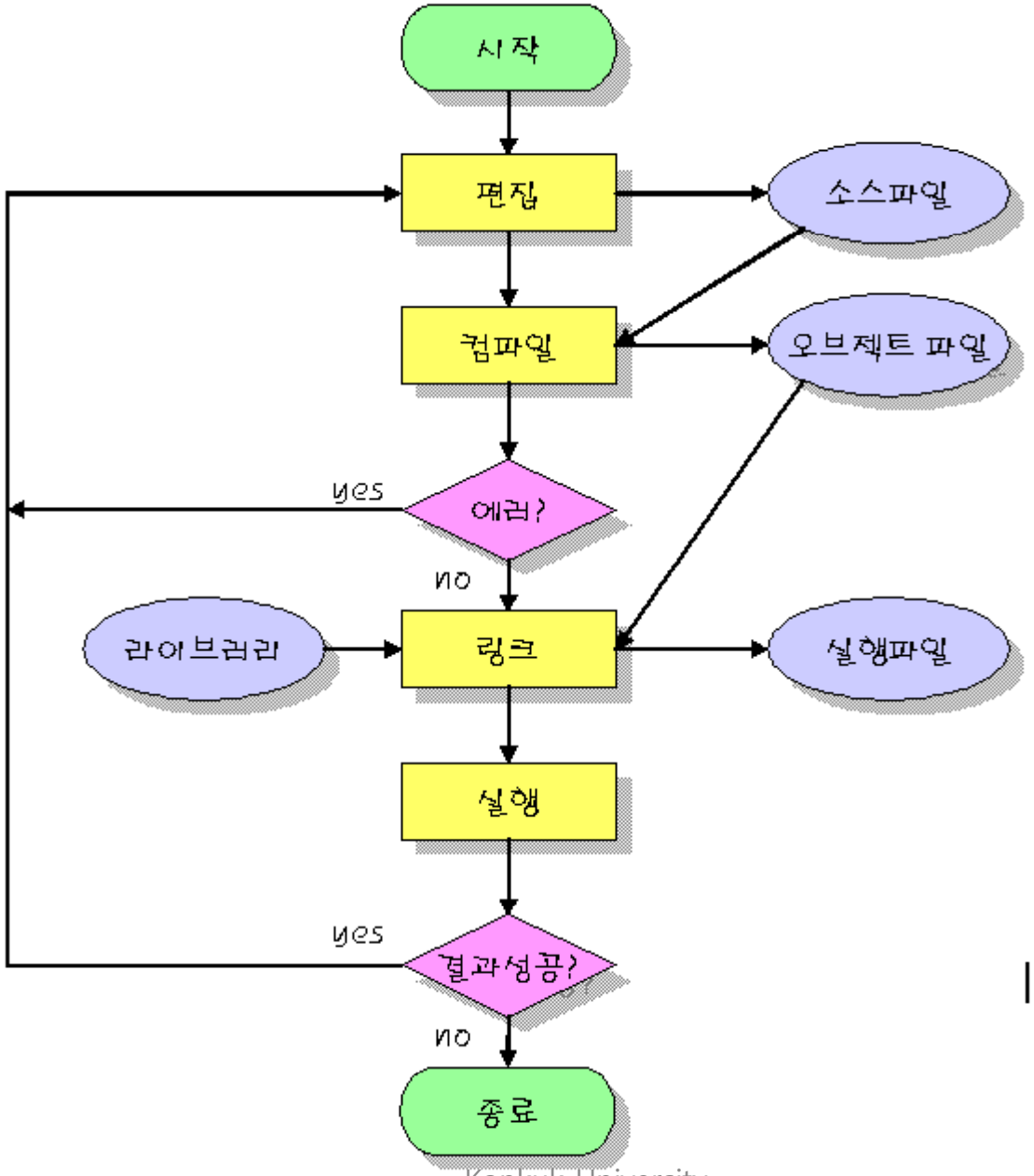
프로그램 작성 단계

- 편집 (edit)
  - 에디터를 이용하여 원하는 작업의 내용을 기술하여 소스 코드 작성
  - 소스 파일(source file): 소스 코드가 들어 있는 텍스트 파일
    - (예) test.c



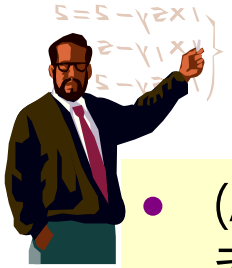
- 컴파일 (compile)
  - 소스 파일->기계어로 변환
  - 오브젝트 파일(object file) : 기계어로 변환된 파일
    - (예) test.obj

- 링크(link)
  - 오브젝트 파일들을 라이브러리 파일들과 연결하여 하나의 실행 파일 생성
  - 실행 파일 (executable file): 실행이 가능한 파일
    - (예) test.exe


시작
편집
소스파일
컴파일
오브젝트 파일
yes
에러?
no
라이브러리
링크
실행파일
실행
yes
결과성공?
no
종료

Q & A

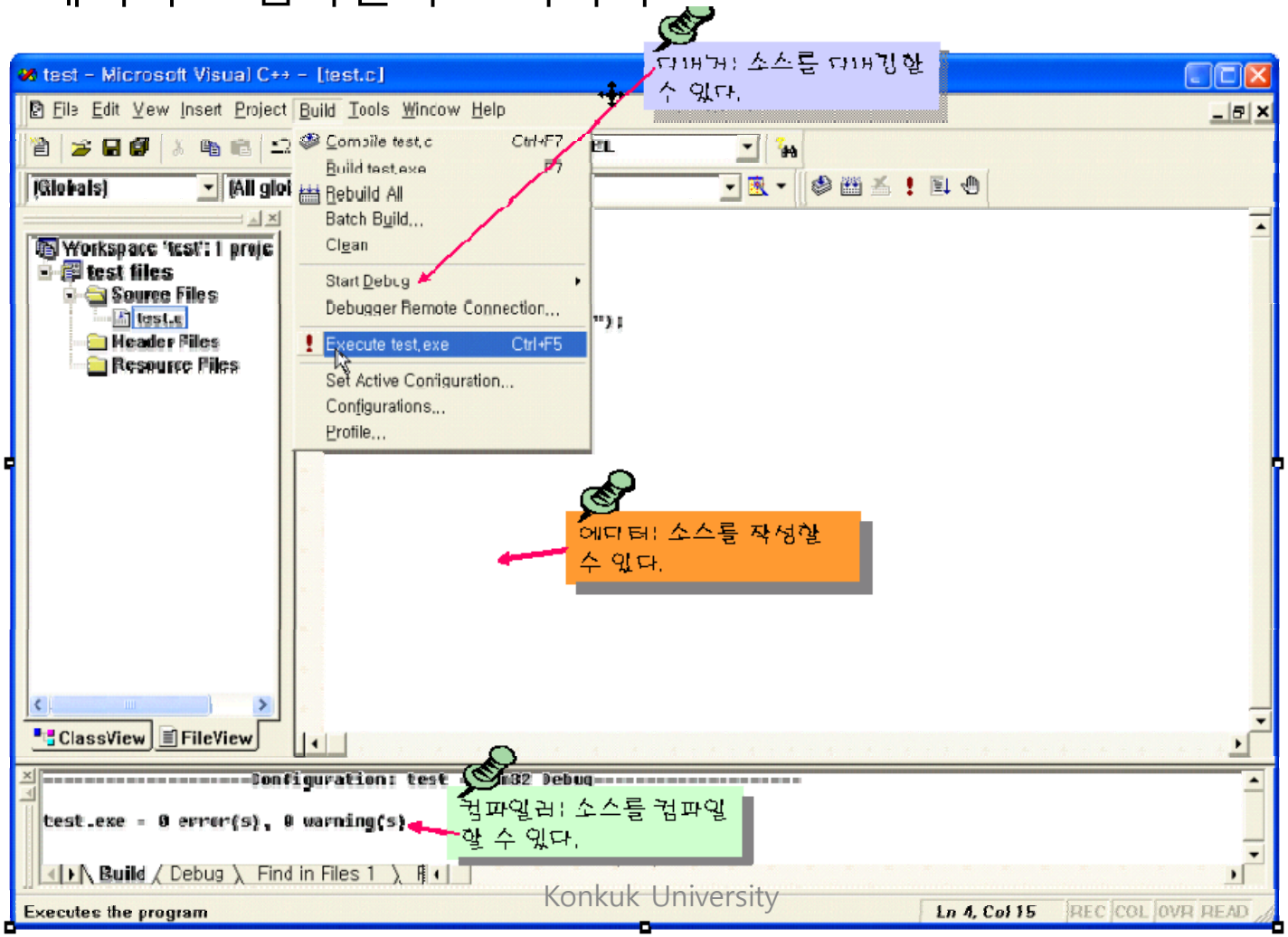
- (Q)소스 파일과 오브젝트 파일, 실행 파일 중에서 반드시 보관하여야 하는 파일은 무엇일까?



- (A) 정답은 소스 파일이다. 소스 파일만 있으면 컴파일러를 수행시켜서 오브젝트 파일, 실행 파일은 만들 수 있다. 하지만 소스 파일을 삭제하면 컴파일이 불가능하다. 따라서 반드시 소스 파일은 잘 보관하여야 한다. Visual C++에서는 프로젝트와 워크스페이스 파일도 같이 보관하는 것이 좋다. 이러한 파일들은 다시 만들 수도 있지만 번거로운 작업이 된다.

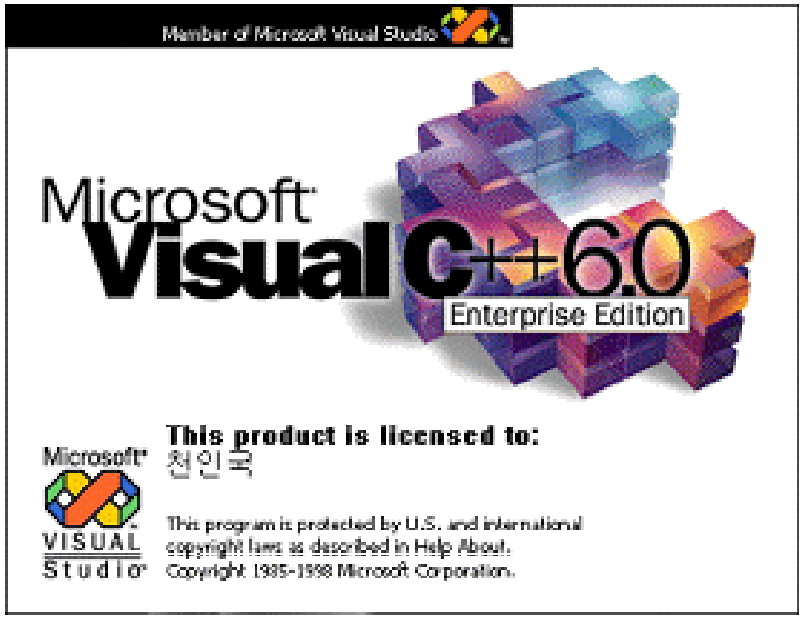
통합 개발 환경

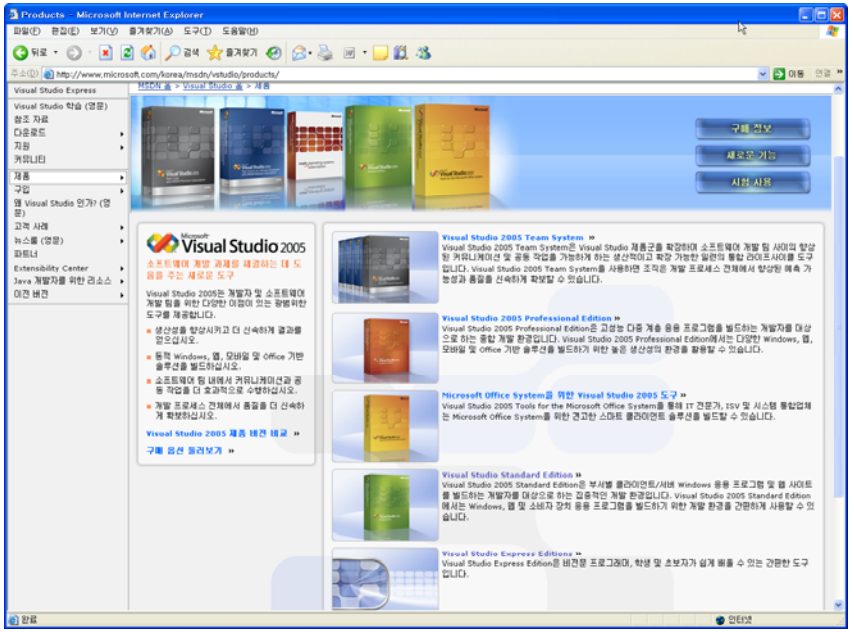
- 통합 개발 환경(IDE: Integrated Development Environment):
  - 에디터 + 컴파일러 + 디버거



통합 개발 환경의 종류

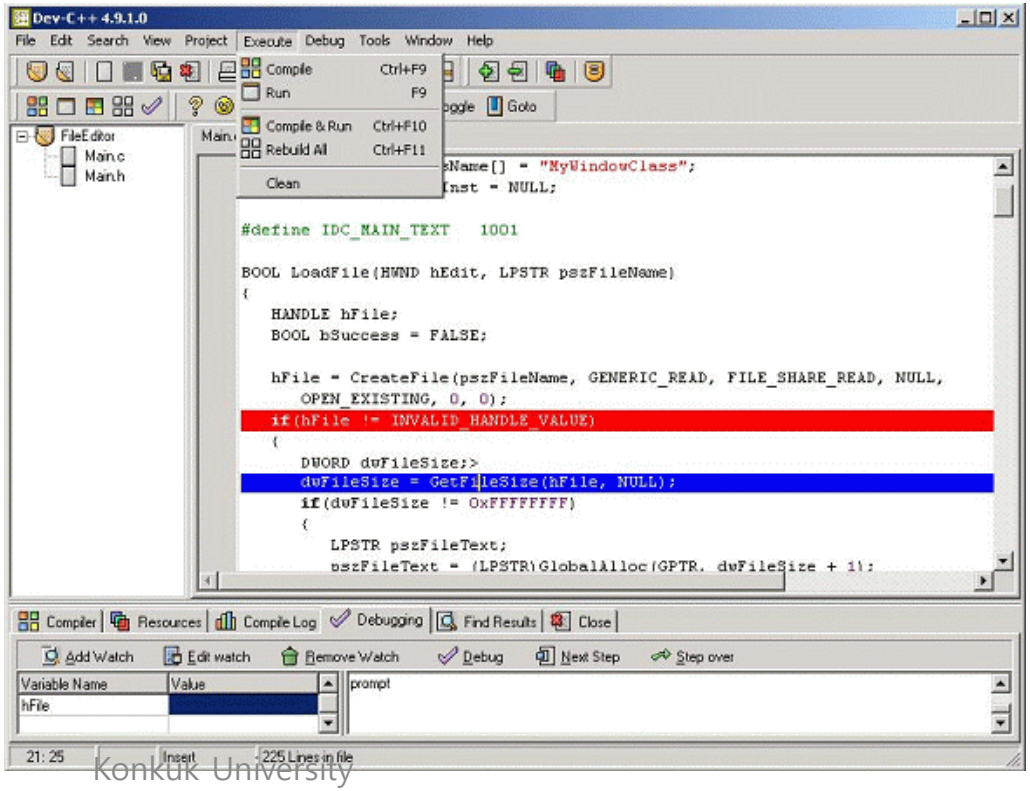
- 비주얼 C++(Visual C++)
  - 마이크로소프트사의 제품
  - 윈도우 기반의 거의 모든 형태의 응용 프로그램 제작 가능
  - 최신 버전: 비주얼 스튜디오 2005
  - 우리가 사용할 버전: 비주얼 스튜디오 6.0



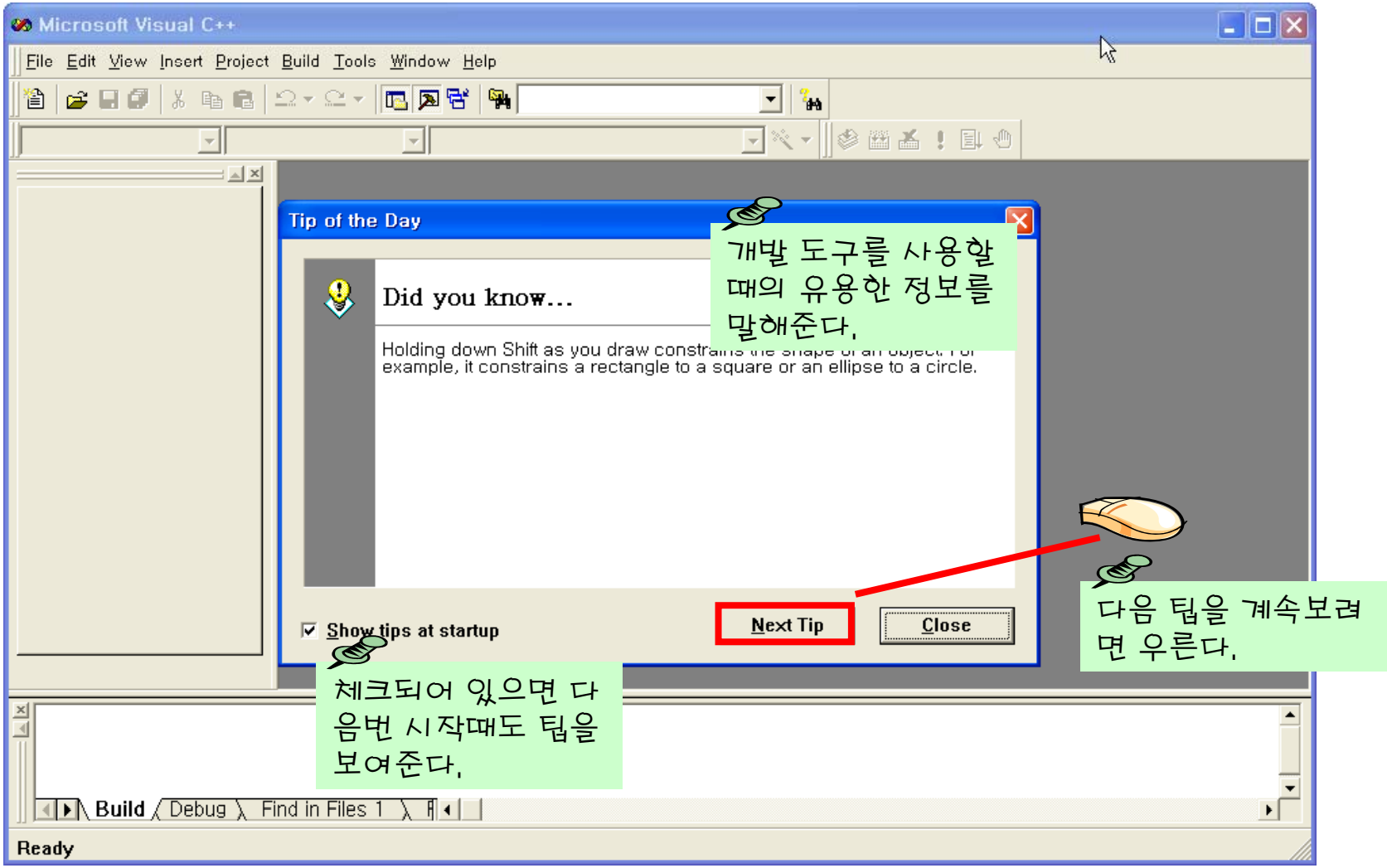


통합 개발 환경의 종류

- Dev-C++
  - 오픈 소스 프로젝트의 산물
  - C/C++ 통합 개발 환경
  - GCC 컴파일러 이용
  - 무료

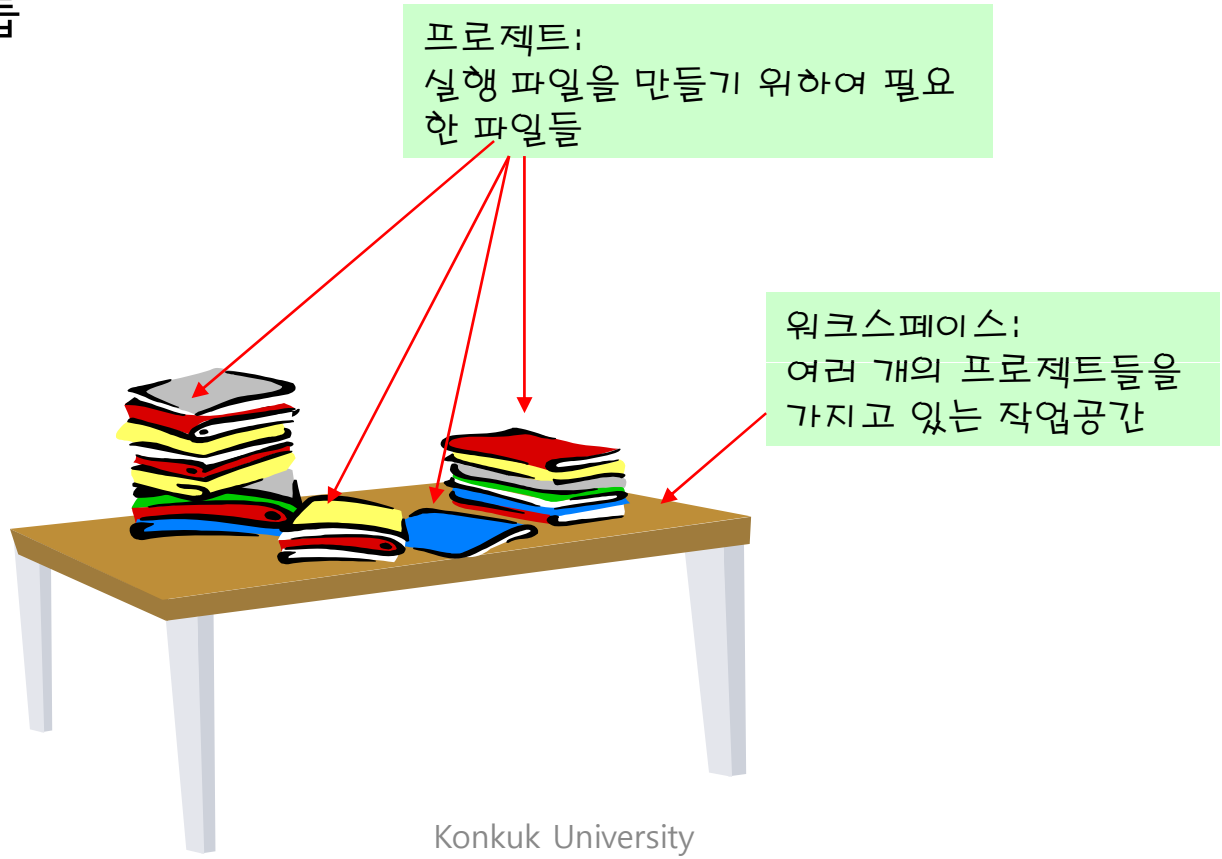


비주얼 C++ 실행

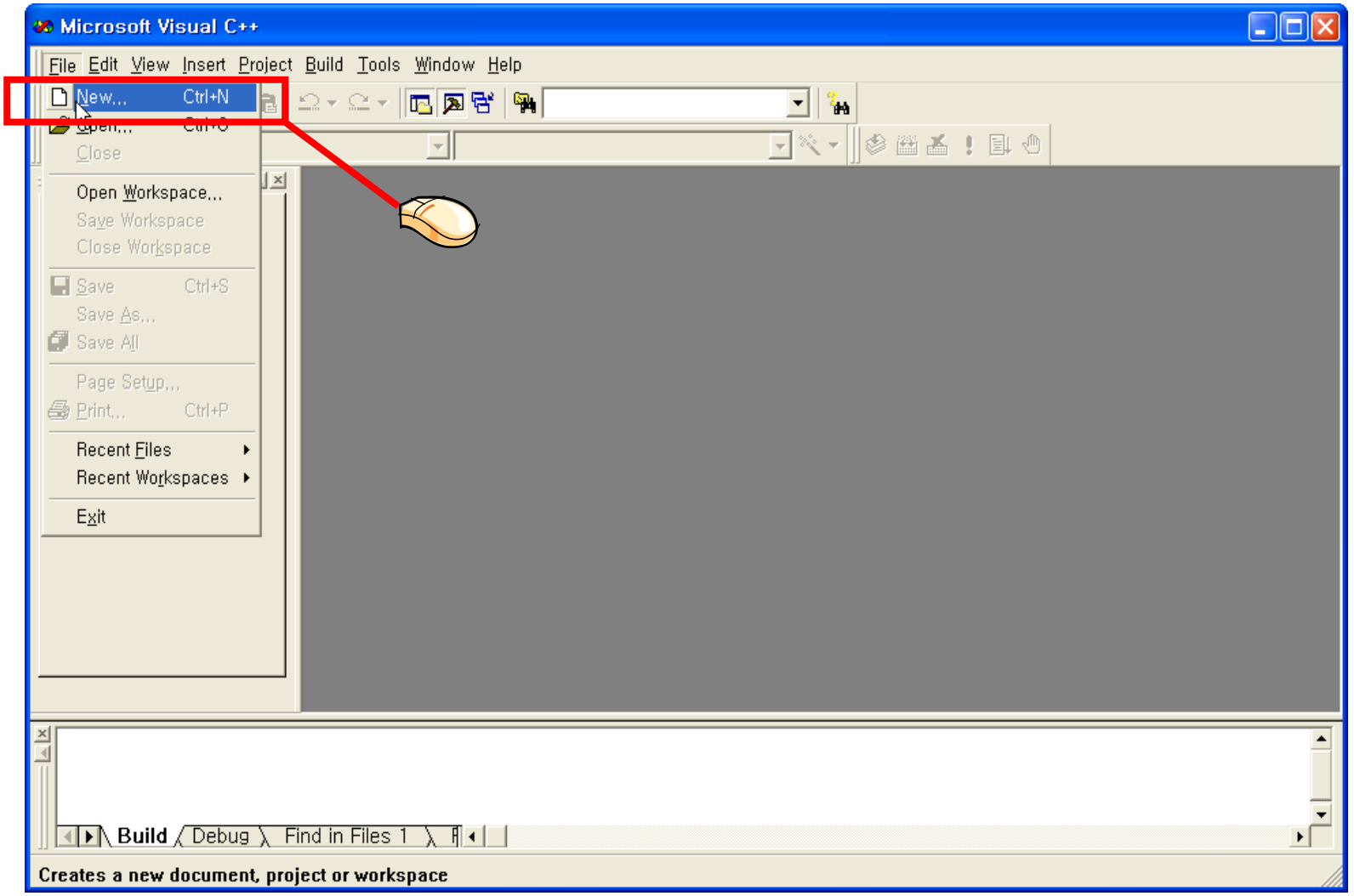

Microsoft Visual C++
File Edit View Insert Project Build Tools Window Help
Tip of the Day
Did you know...
Holding down Shift as you draw constrains the shape of an object. For example, it constrains a rectangle to a square or an ellipse to a circle.
Show tips at startup
Next Tip
Close
개발 도구를 사용할 때의 유용한 정보를 말해준다.
다음 팁을 계속보려면 우른다.
체크되어 있으면 다음번 시작때도 팁을 보여준다.
Build
Debug
Find in Files 1
Ready

워크스페이스와 프로젝트

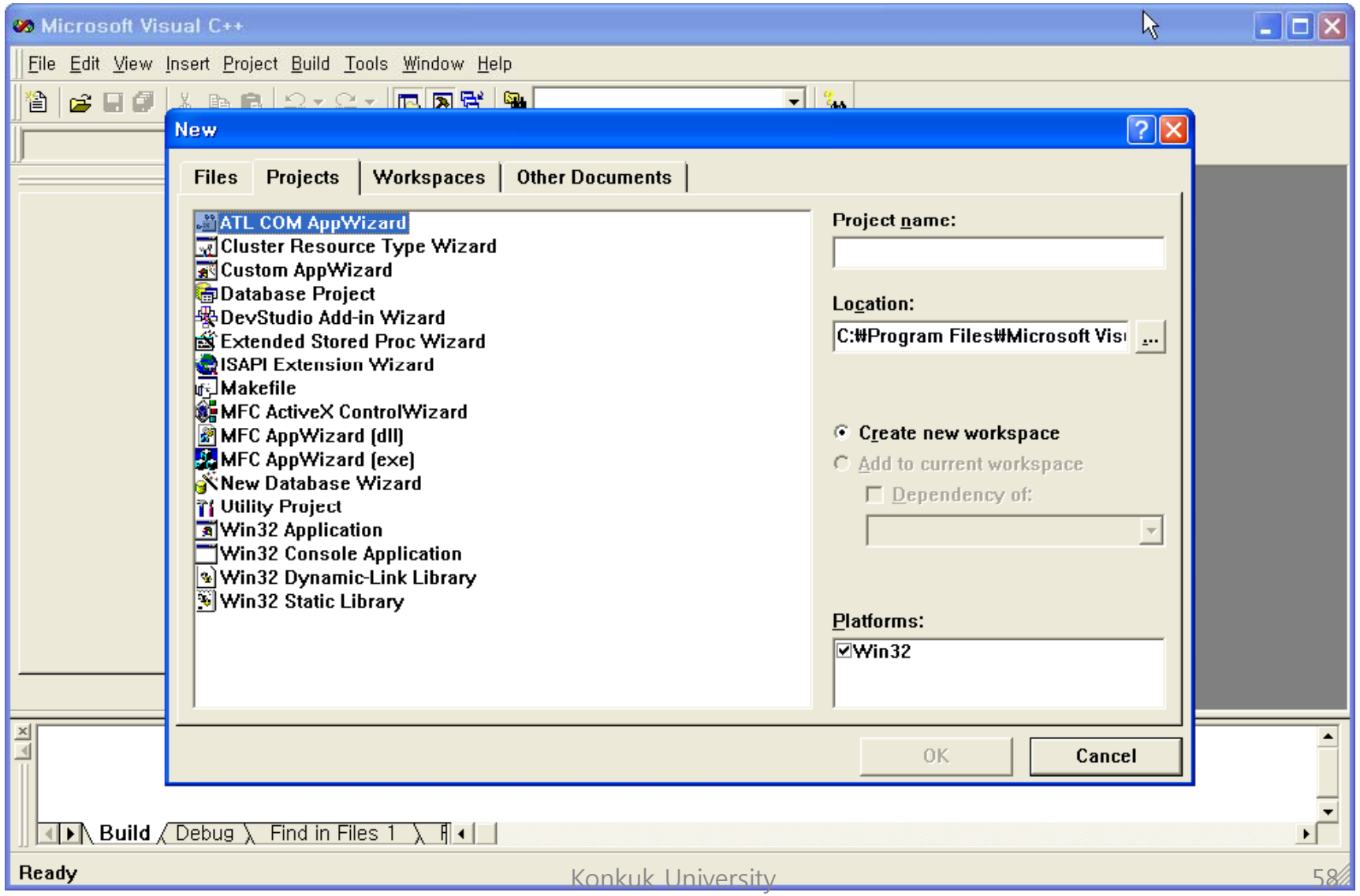
- 워크스페이스(workspace); 작업 공간, 여러 개의 프로젝트가 있을 수 있다.
- 프로젝트(project): 하나의 실행 파일을 만들기 위하여 필요한 파일들의 그룹



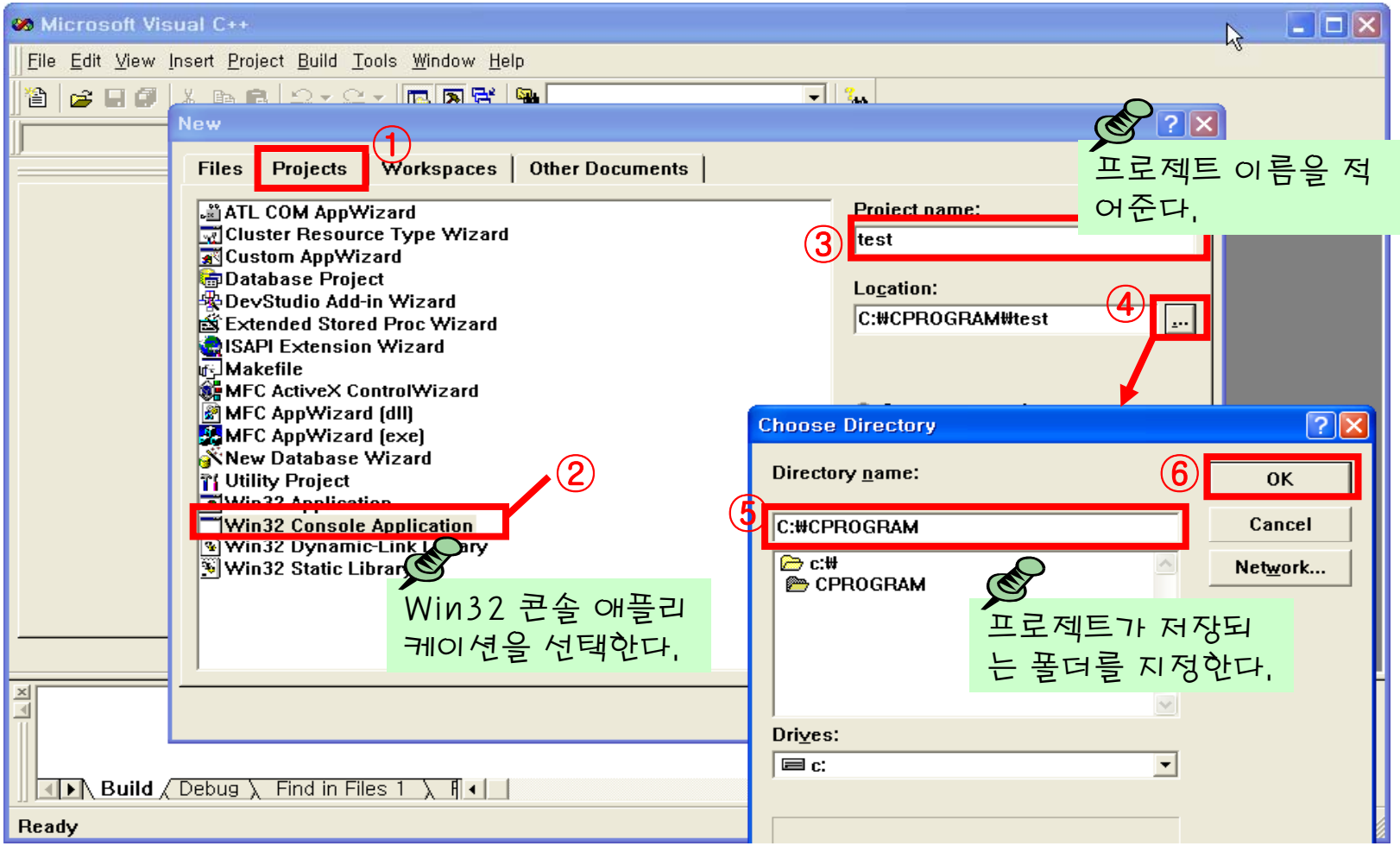
비주얼 C++ 시작하기


Microsoft Visual C++
File Edit View Insert Project Build Tools Window Help
New... Ctrl+N
Close
Open Workspace...
Save Workspace
Close Workspace
Save Ctrl+S
Save As...
Save All
Page Setup...
Print... Ctrl+P
Recent Files
Recent Workspaces
Exit
Build Debug Find in Files 1
Creates a new document, project or workspace

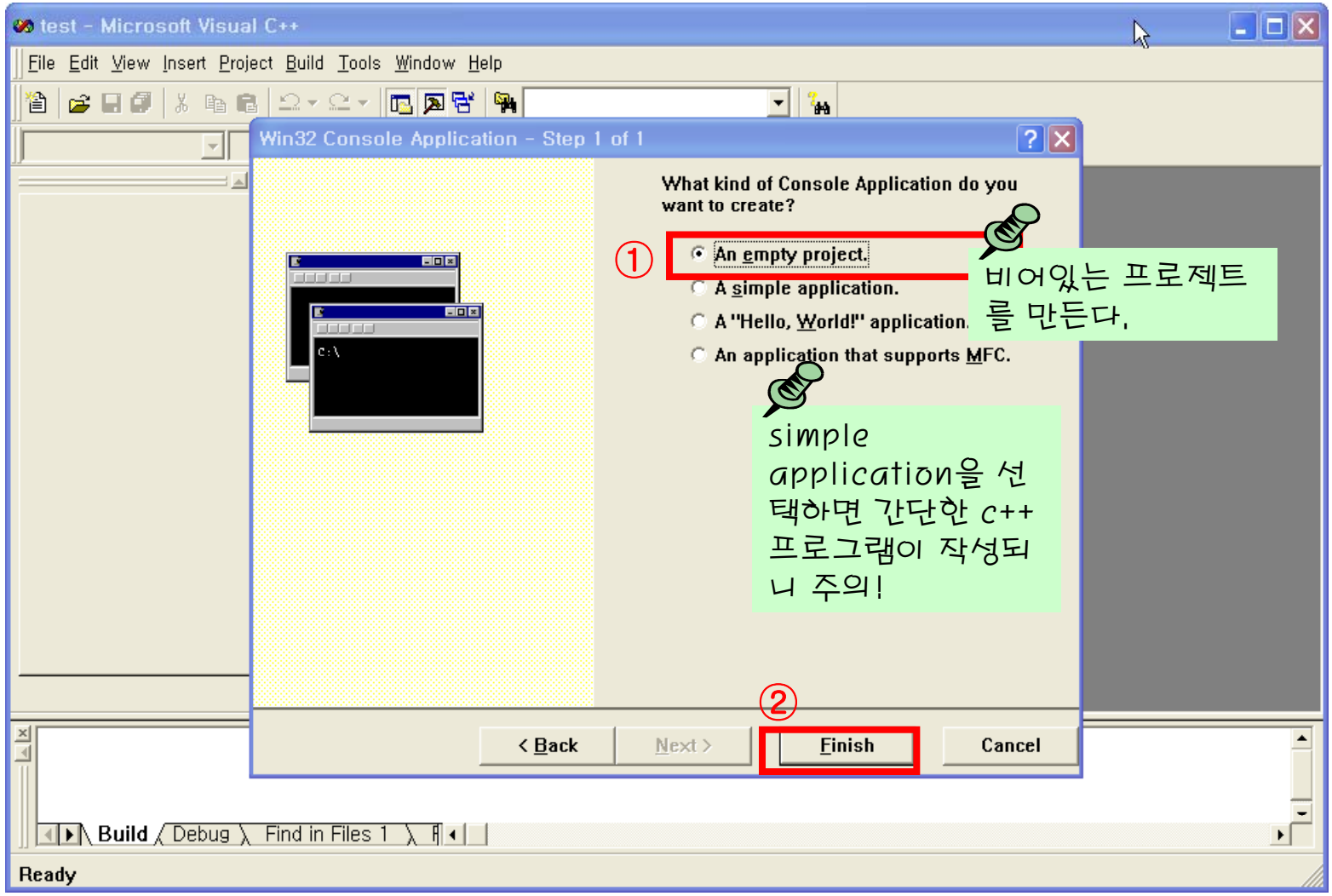
프로젝트 생성하기


Microsoft Visual C++
File Edit View Insert Project Build Tools Window Help
New
Files
Projects
Workspaces
Other Documents
ATL COM AppWizard
Cluster Resource Type Wizard
Custom AppWizard
Database Project
DevStudio Add-in Wizard
Extended Stored Proc Wizard
ISAPI Extension Wizard
Makefile
MFC ActiveX ControlWizard
MFC AppWizard (dll)
MFC AppWizard (exe)
New Database Wizard
Utility Project
Win32 Application
Win32 Console Application
Win32 Dynamic-Link Library
Win32 Static Library
Project name:
Location:
C:₩Program Files₩Microsoft Vis
...
Create new workspace
Add to current workspace
Dependency of:
Platforms:
Win32
OK
Cancel
Build
Debug
Find in Files 1
Ready

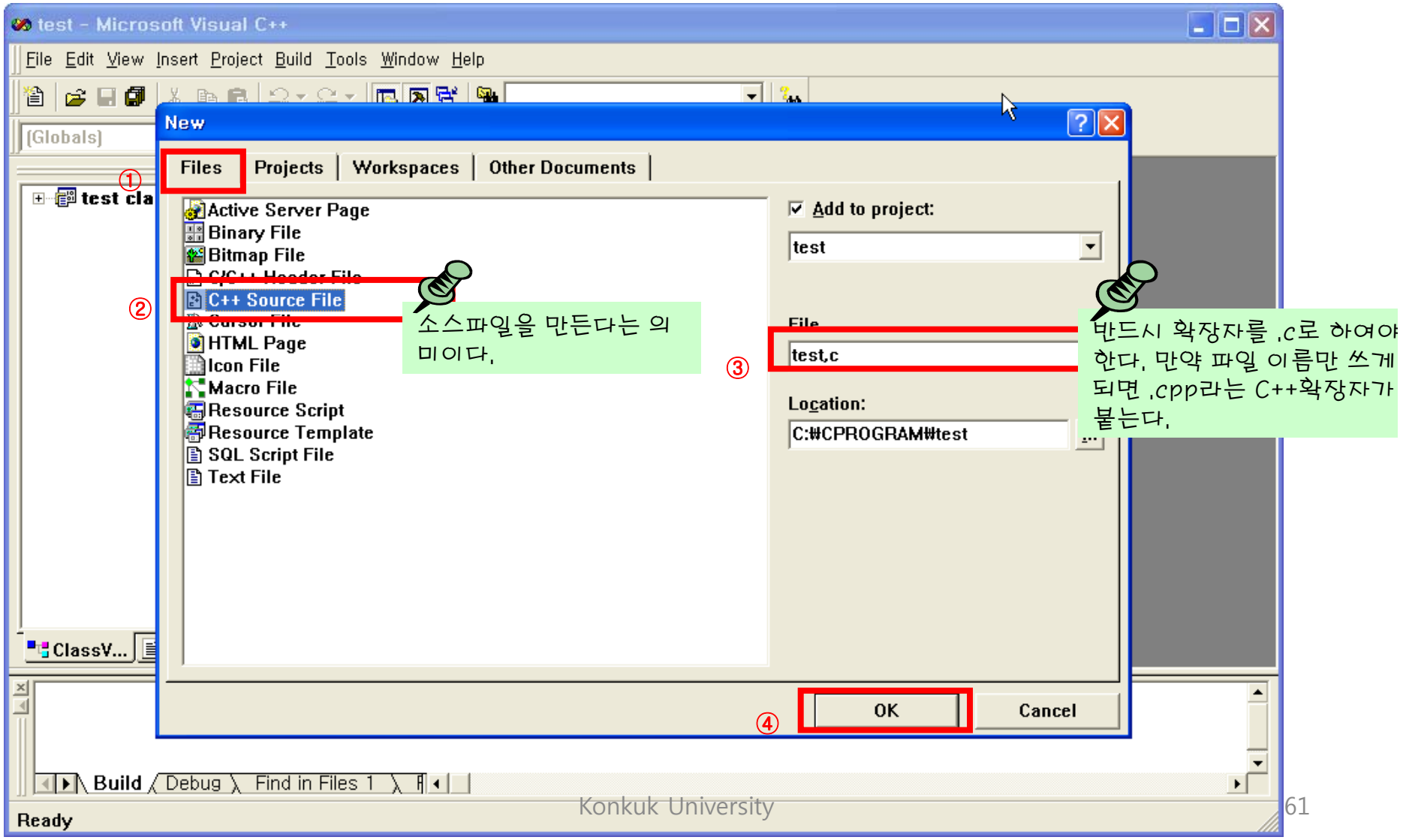
프로젝트 생성하기


Microsoft Visual C++
File Edit View Insert Project Build Tools Window Help
New
Files
Projects
Workspaces
Other Documents
ATL COM AppWizard
Cluster Resource Type Wizard
Custom AppWizard
Database Project
DevStudio Add-in Wizard
Extended Stored Proc Wizard
ISAPI Extension Wizard
Makefile
MFC ActiveX ControlWizard
MFC AppWizard (dll)
MFC AppWizard (exe)
New Database Wizard
Utility Project
Win32 Console Application
Win32 Static Library
①
②
Win32 콘솔 애플리케이션을 선택한다.
Project name:
test
③
Location:
C:₩CPROGRAM₩test
④
프로젝트 이름을 적어준다.
Choose Directory
Directory name:
C:₩CPROGRAM
⑤
OK
⑥
Cancel
Network...
c:₩
CPROGRAM
프로젝트가 저장되는 폴더를 지정한다.
Drives:
c:
Build
Debug
Find in Files 1
Ready

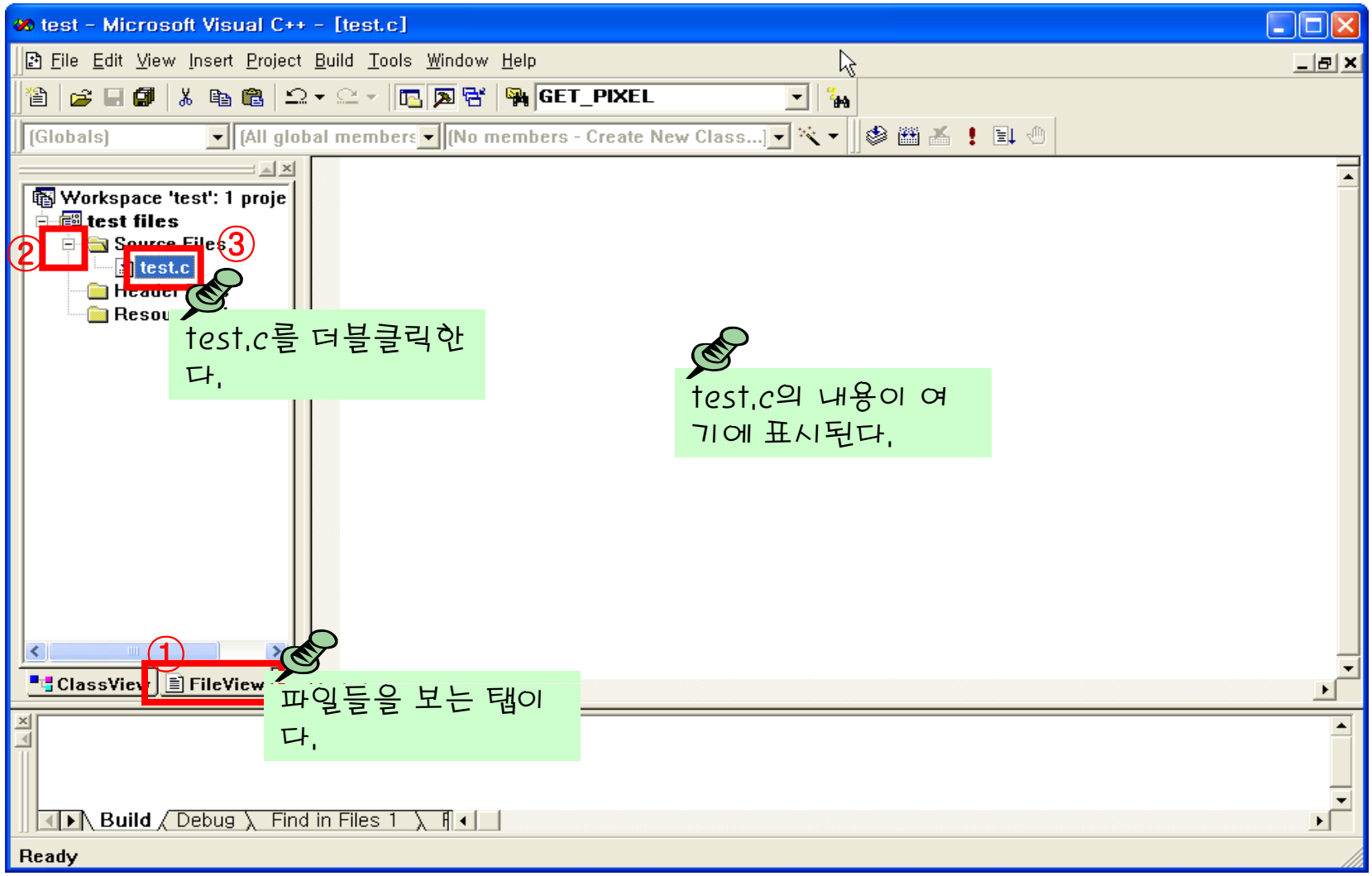
프로젝트 생성하기


test - Microsoft Visual C++
File Edit View Insert Project Build Tools Window Help
Win32 Console Application - Step 1 of 1
What kind of Console Application do you want to create?
①
An empty project.
A simple application.
A "Hello, World!" application.
An application that supports MFC.
비어있는 프로젝트를 만든다.
simple application을 선택하면 간단한 c++ 프로그램이 작성되니 주의!
②
< Back
Next >
Finish
Cancel
Build
Debug
Find in Files 1
Ready

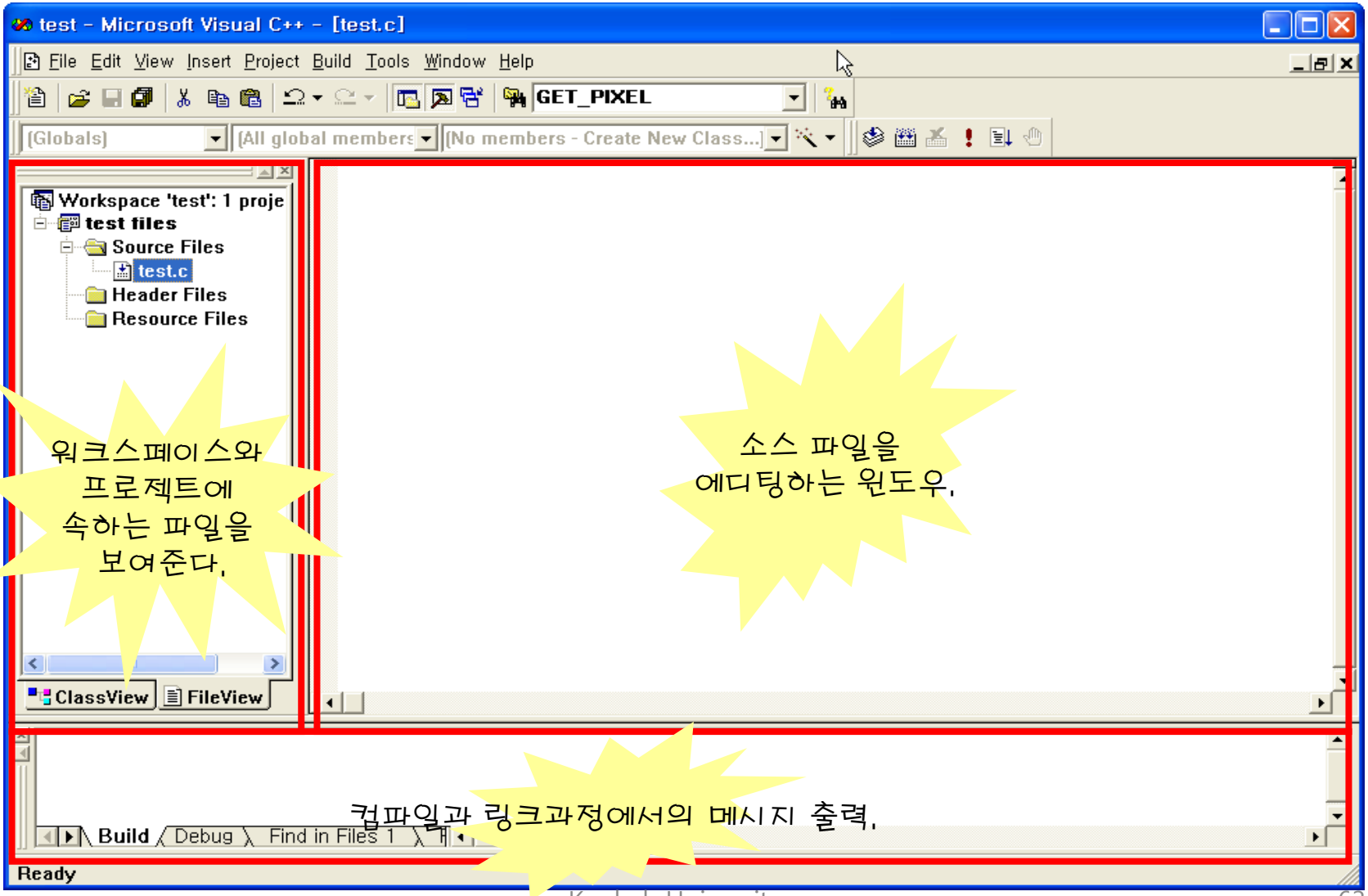
소스 파일 생성하기


test - Microsoft Visual C++
File Edit View Insert Project Build Tools Window Help
(Globals)
test cla
ClassV...
New
Files
Projects
Workspaces
Other Documents
Active Server Page
Binary File
Bitmap File
C++ Source File
HTML Page
Icon File
Macro File
Resource Script
Resource Template
SQL Script File
Text File
Add to project:
test
File
test.c
Location:
C:₩CPROGRAM₩test
OK
Cancel
①
②
③
④
소스파일을 만든다는 의미이다.
반드시 확장자를 .c로 하여야 한다. 만약 파일 이름만 쓰게 되면 .cpp라는 C++확장자가 붙는다.
Build
Debug
Find in Files 1
Ready

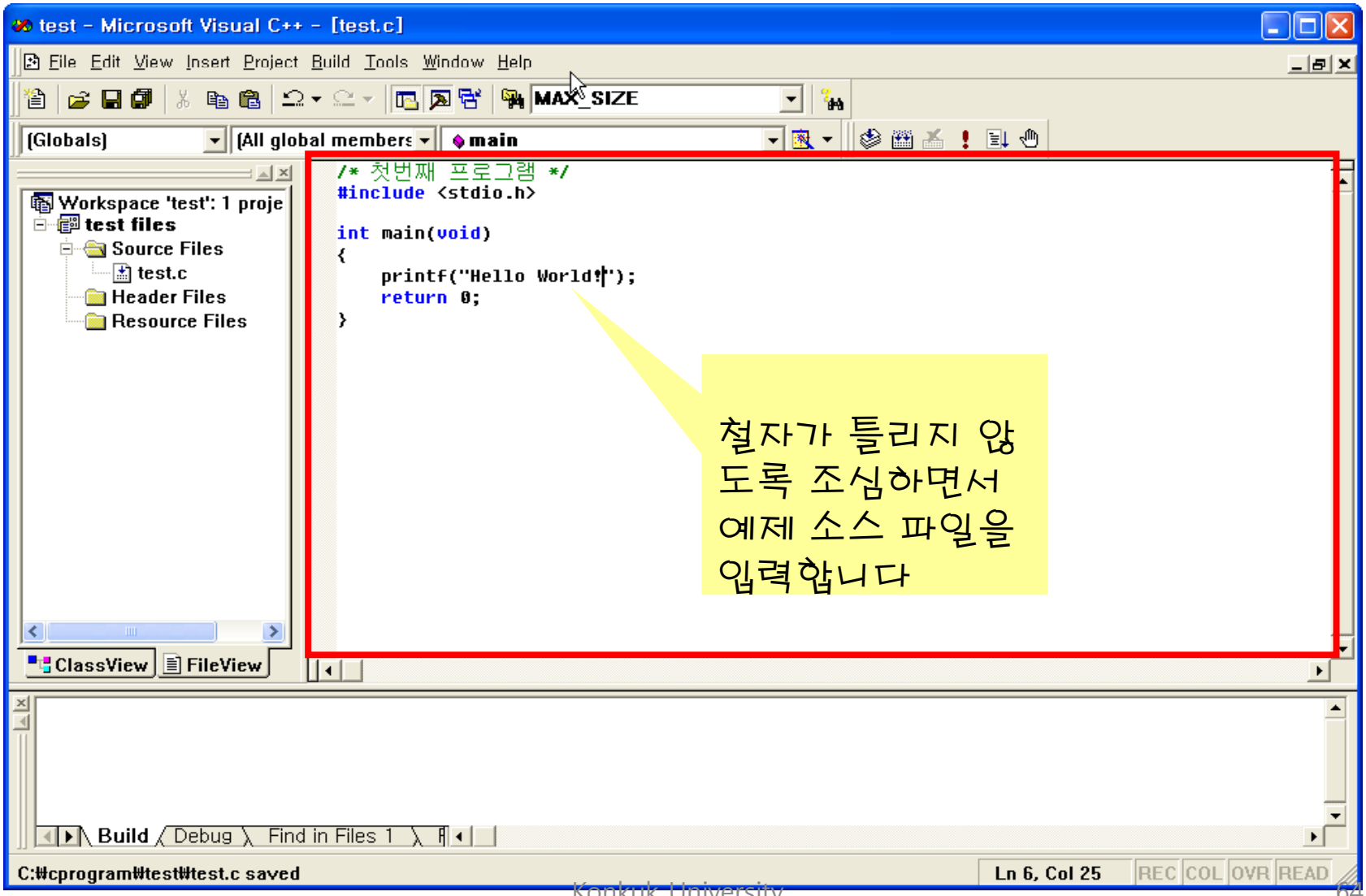
소스 파일에 프로그램 작성



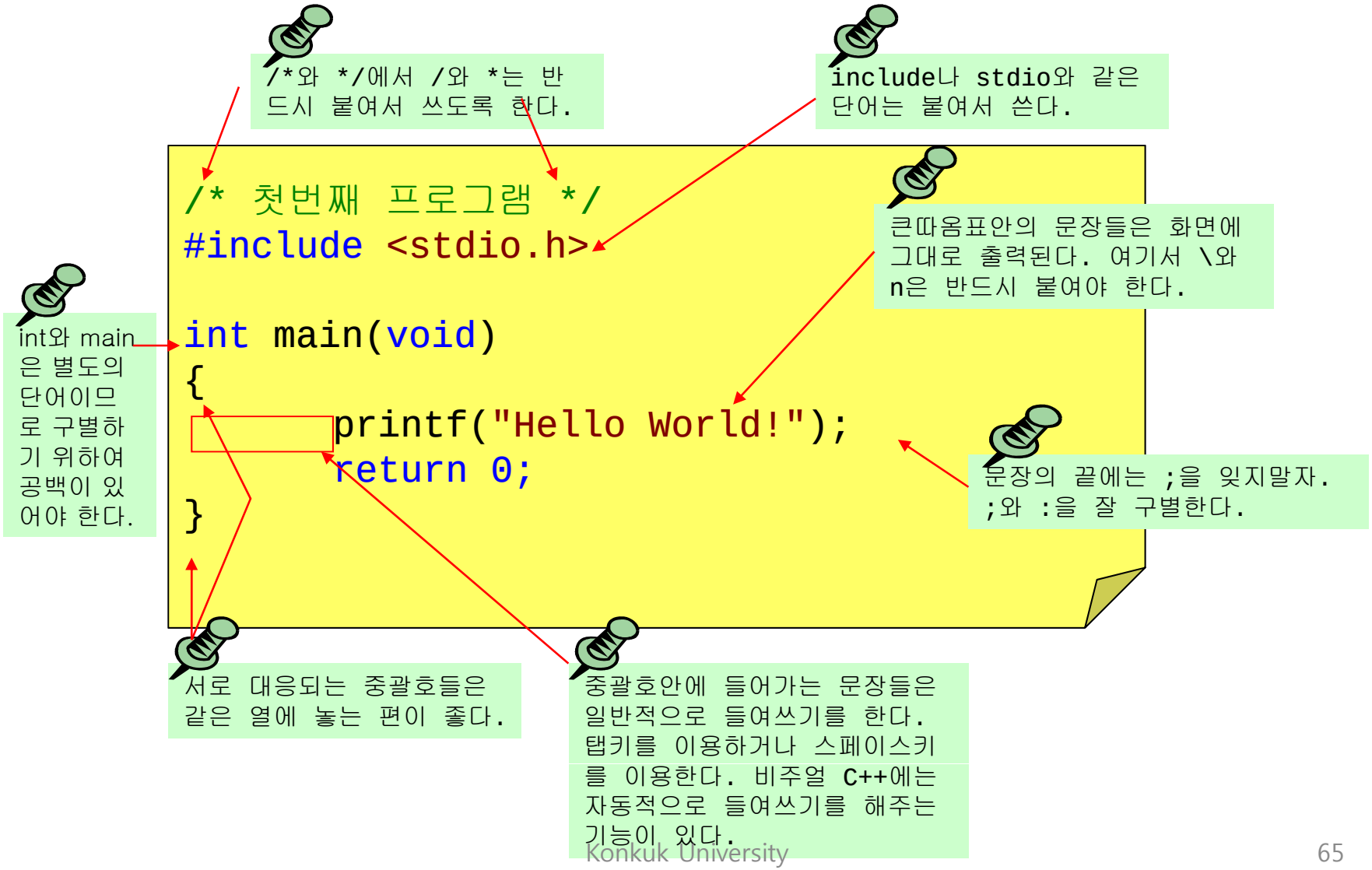
비주얼 C++의 전체 구조


test - Microsoft Visual C++ - [test.c]
File Edit View Insert Project Build Tools Window Help
GET_PIXEL
(Globals)
(All global members
(No members - Create New Class...)
Workspace 'test': 1 proje
test files
Source Files
test.c
Header Files
Resource Files
ClassView
FileView
Build
Debug
Find in Files 1
Ready
워크스페이스와 프로젝트에 속하는 파일을 보여준다.
소스 파일을 에디팅하는 윈도우.
컴파일과 링크과정에서의 메시지 출력.

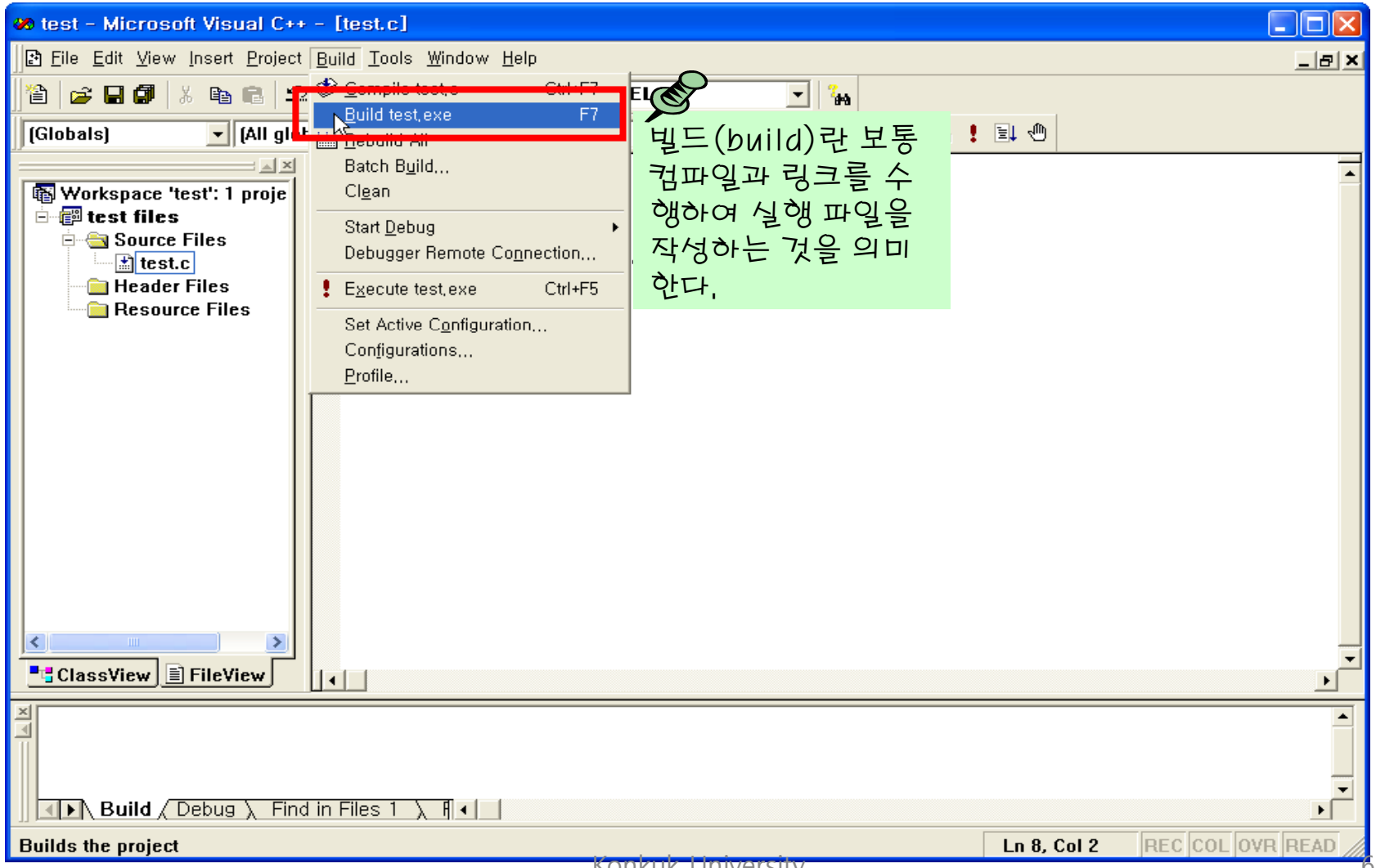
프로그램 입력


test - Microsoft Visual C++ - [test.c]
File Edit View Insert Project Build Tools Window Help
MAX_SIZE
(Globals)
(All global members
main
Workspace 'test': 1 proje
test files
Source Files
test.c
Header Files
Resource Files
ClassView
FileView
/* 첫번째 프로그램 */
#include <stdio.h>
int main(void)
{
printf("Hello World");
return 0;
}
철자가 틀리지 않도록 조심하면서 예제 소스 파일을 입력합니다
Build
Debug
Find in Files 1
C:₩cprogram₩test₩test.c saved
Ln 6, Col 25
REC COL OVR READ

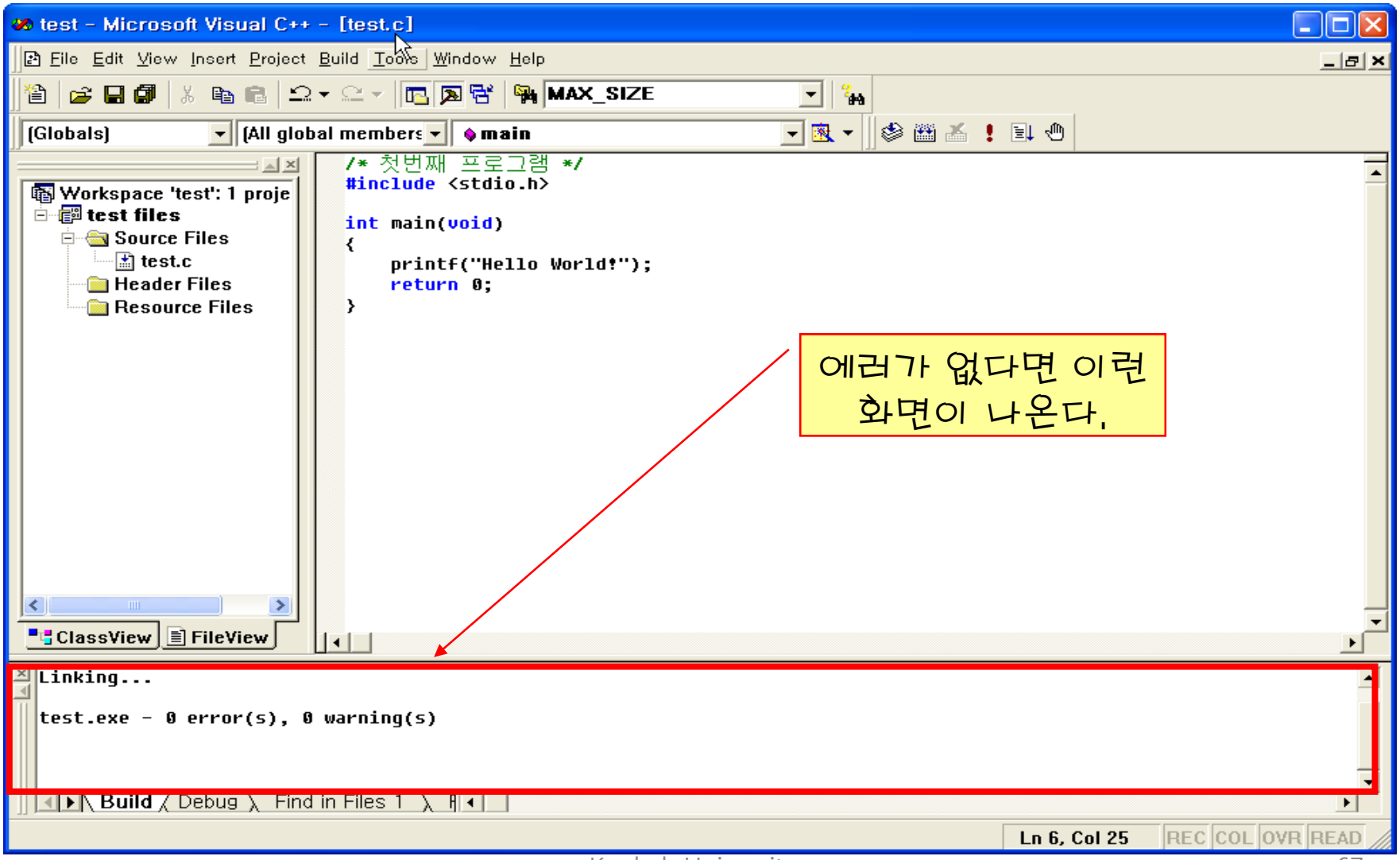
프로그램 입력시 주의 사항



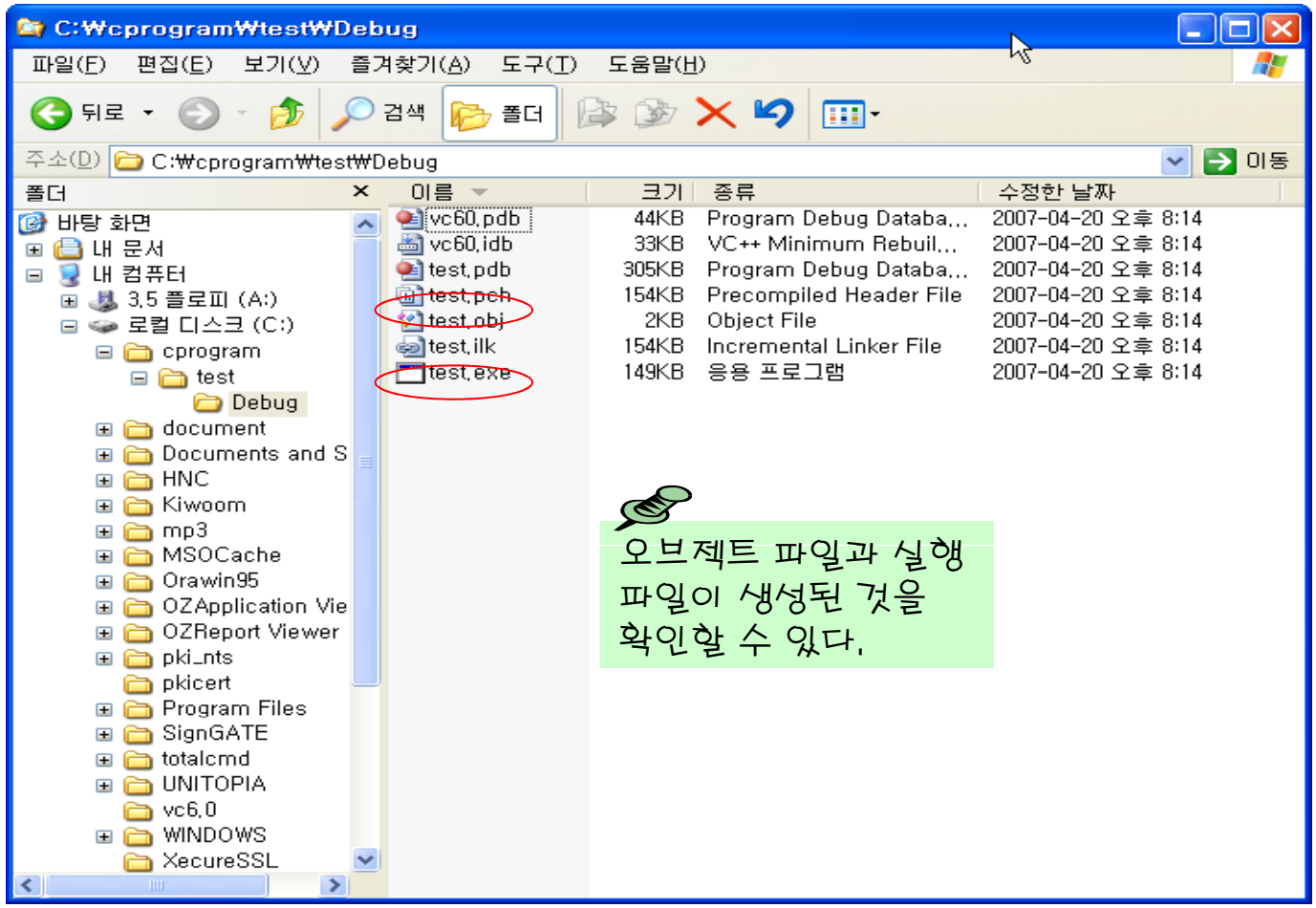
컴파일하기


test - Microsoft Visual C++ - [test.c]
File Edit View Insert Project Build Tools Window Help
Build test.exe F7
Batch Build...
Clean
Start Debug
Debugger Remote Connection...
Execute test.exe Ctrl+F5
Set Active Configuration...
Configurations...
Profile...
Workspace 'test': 1 proje
test files
Source Files
test.c
Header Files
Resource Files
ClassView
FileView
Build
Debug
Find in Files 1
Builds the project
Ln 8, Col 2
REC COL OVR READ
빌드(build)란 보통 컴파일과 링크를 수행하여 실행 파일을 작성하는 것을 의미한다.

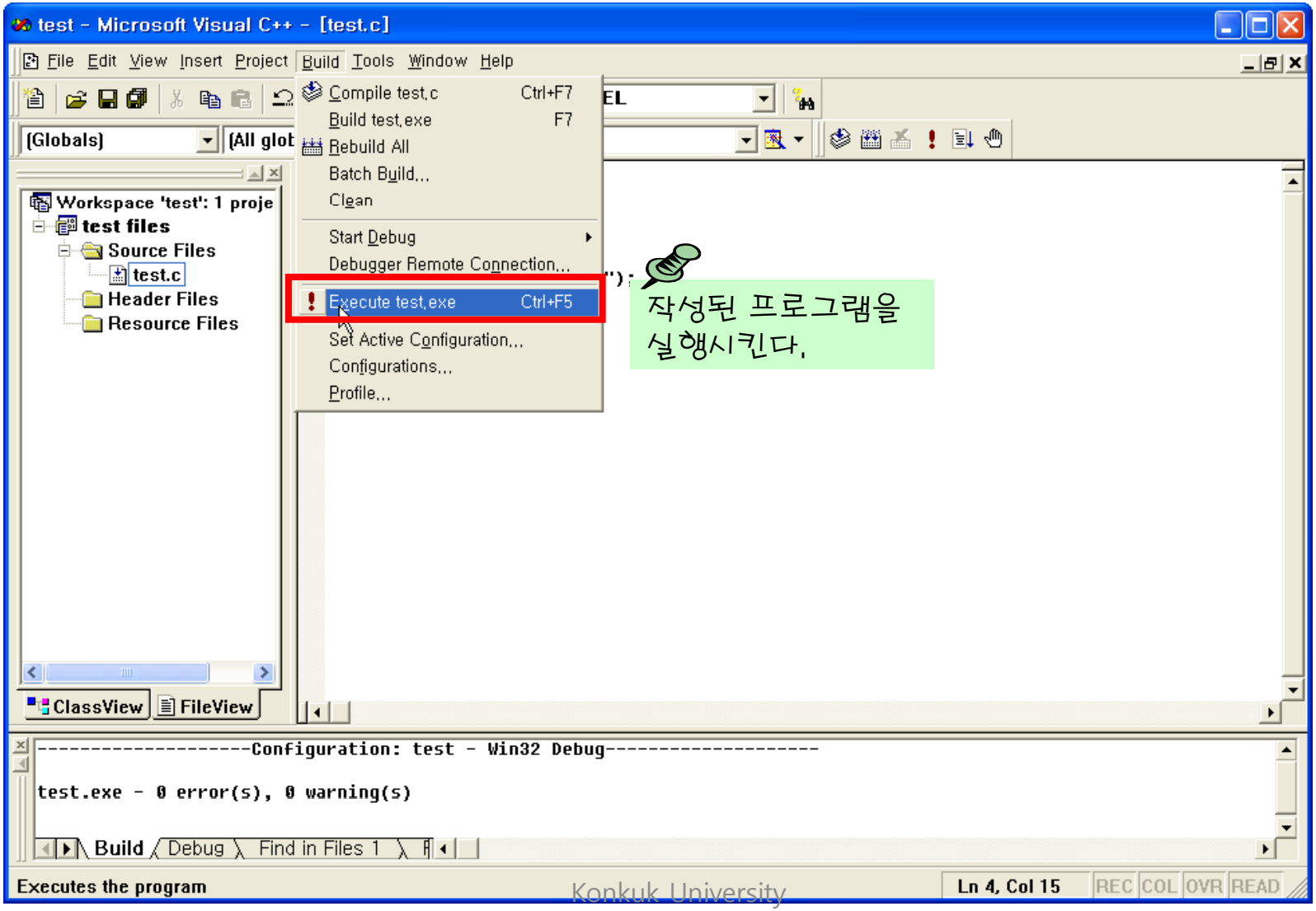
컴파일 결과


test - Microsoft Visual C++ - [test.c]
File Edit View Insert Project Build Tools Window Help
MAX_SIZE
(Globals)
(All global members
main
Workspace 'test': 1 proje
test files
Source Files
test.c
Header Files
Resource Files
ClassView
FileView
/* 첫번째 프로그램 */
#include <stdio.h>
int main(void)
{
printf("Hello World!");
return 0;
}
에러가 없다면 이런 화면이 나온다.
Linking...
test.exe - 0 error(s), 0 warning(s)
Build
Debug
Find in Files 1
Ln 6, Col 25
REC COL OVR READ

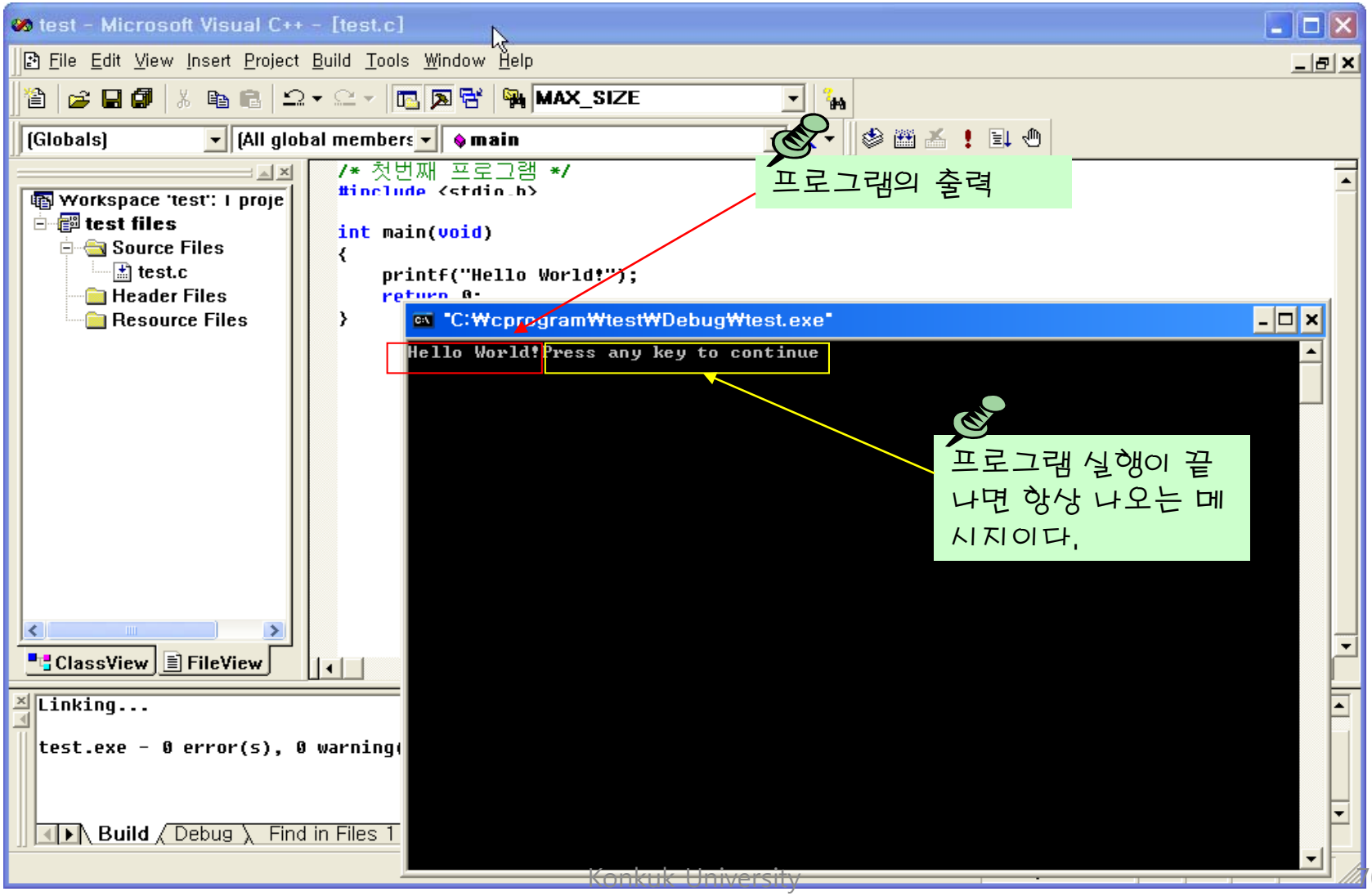
오브젝트 파일과 실행 파일의 생성



프로그램 실행 하기


test - Microsoft Visual C++ - [test.c]
File Edit View Insert Project Build Tools Window Help
Compile test.c Ctrl+F7
Build test.exe F7
Rebuild All
Batch Build...
Clean
Start Debug
Debugger Remote Connection...
Execute test.exe Ctrl+F5
Set Active Configuration...
Configurations...
Profile...
Workspace 'test': 1 proje
test files
Source Files
test.c
Header Files
Resource Files
ClassView
FileView
작성된 프로그램을
실행시킨다.
--------------------Configuration: test - Win32 Debug--------------------
test.exe - 0 error(s), 0 warning(s)
Build
Debug
Find in Files 1
Executes the program
Ln 4, Col 15
REC COL OVR READ

실행 결과 화면


test - Microsoft Visual C++ - [test.c]
File Edit View Insert Project Build Tools Window Help
MAX_SIZE
(Globals)
(All global members
main
Workspace 'test': 1 proje
test files
Source Files
test.c
Header Files
Resource Files
ClassView
FileView
/* 첫번째 프로그램 */
#include <stdio.h>
int main(void)
{
printf("Hello World!");
}
"C:₩cprogram₩test₩Debug₩test.exe"
Hello World!Press any key to continue
프로그램의 출력
프로그램 실행이 끝나면 항상 나오는 메시지이다.
Linking...
test.exe - 0 error(s), 0 warning
Build
Debug
Find in Files 1

첫번째 프로그램의 설명

```c
/* 첫번째 프로그램*/
#include <stdio.h>

int main(void)
{
    printf("Hello World!");
    return 0;
}
```

주석

- 주석(comment): 프로그램에 대한 설명

```
/* 한줄로된주석 */

int main(void)   /* 줄의일부분인주석 */

/* 여러
줄로
된주석 */
```

헤더 파일 포함

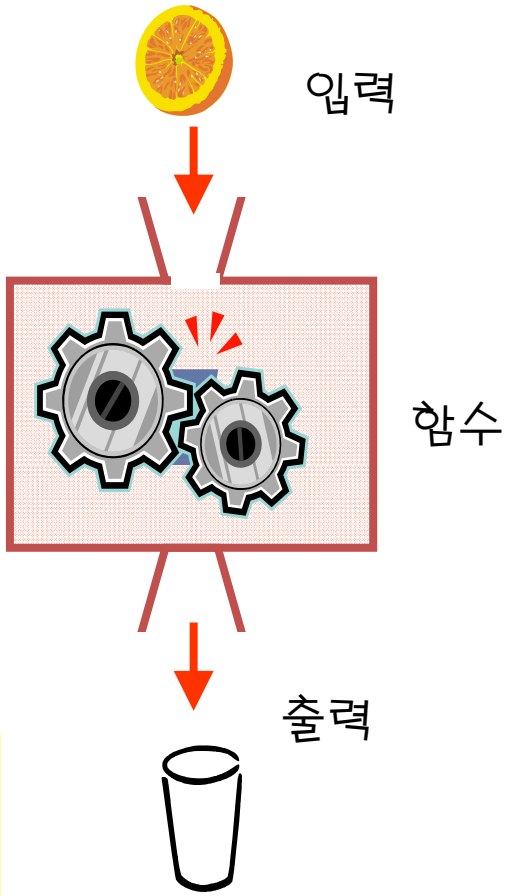
```
#include <stdio.h>
```

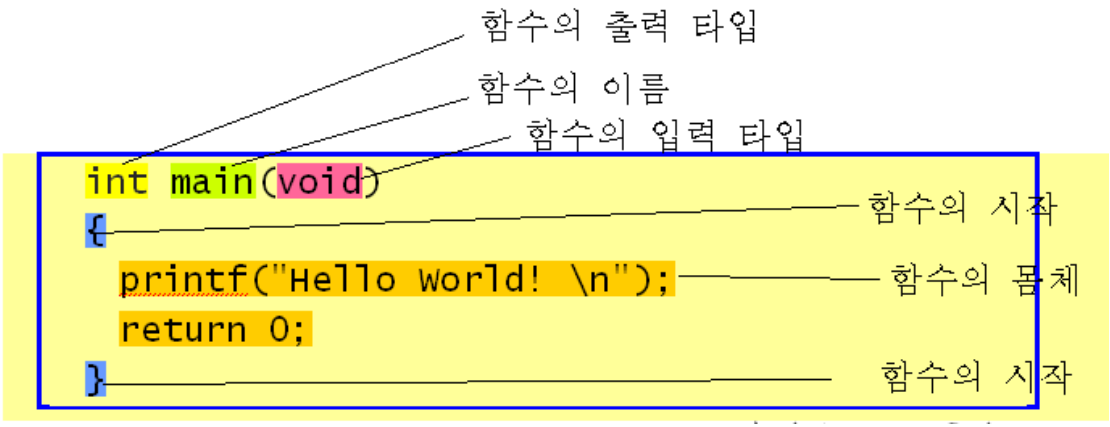
- #include는 소스 코드 안에 특정 파일을 현재의 위치에 포함
- 헤더 파일(header file): 컴파일러가 필요로 하는 정보를 가지고 있는 파일
- stdio.h: standard input output header file
- 주의!: 전처리기 지시자 문장 끝에는 세미콜론을 붙이면 안 된다.

함수

```
int main(void)
```

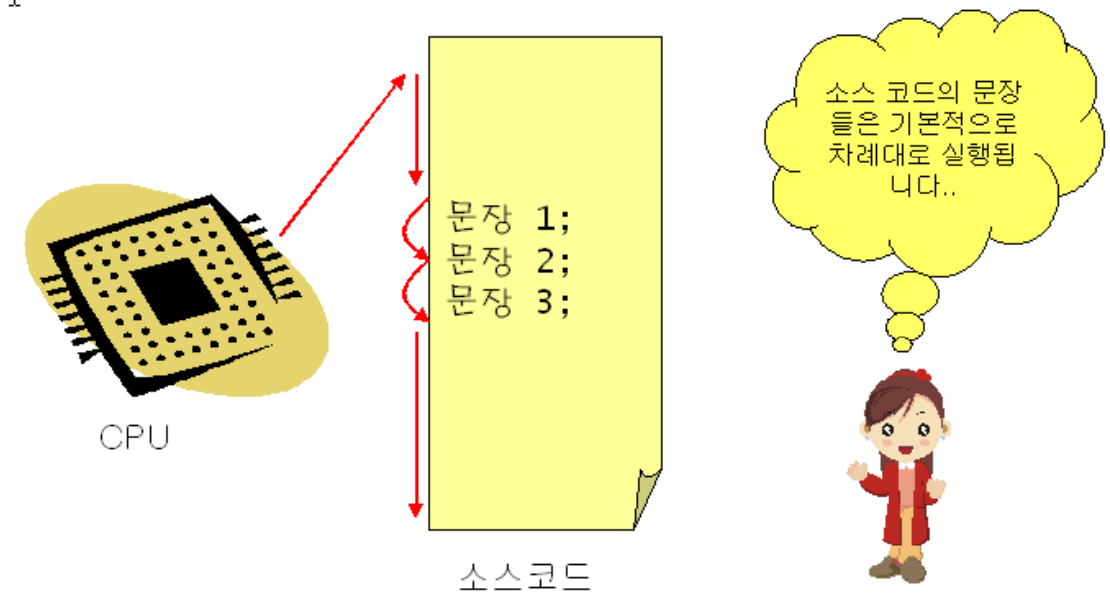
- 함수(function): 특정한 작업을 수행하기 위하여 작성된 독립적인 코드
- (참고) 수학적인 함수 $y = x^2 + 1$
- 프로그램 = 함수의 집합
- main()은 가장 먼저 수행되는 함수





문장

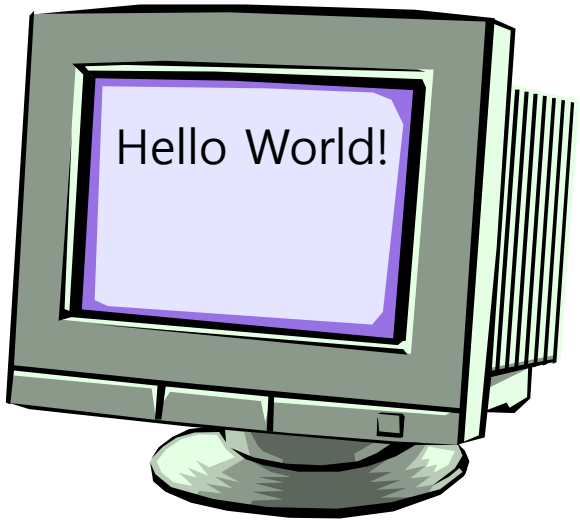
- 함수는 여러 개의 문장으로 이루어진다.
- 문장들은 순차적으로 실행된다.



출력 함수 printf()

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

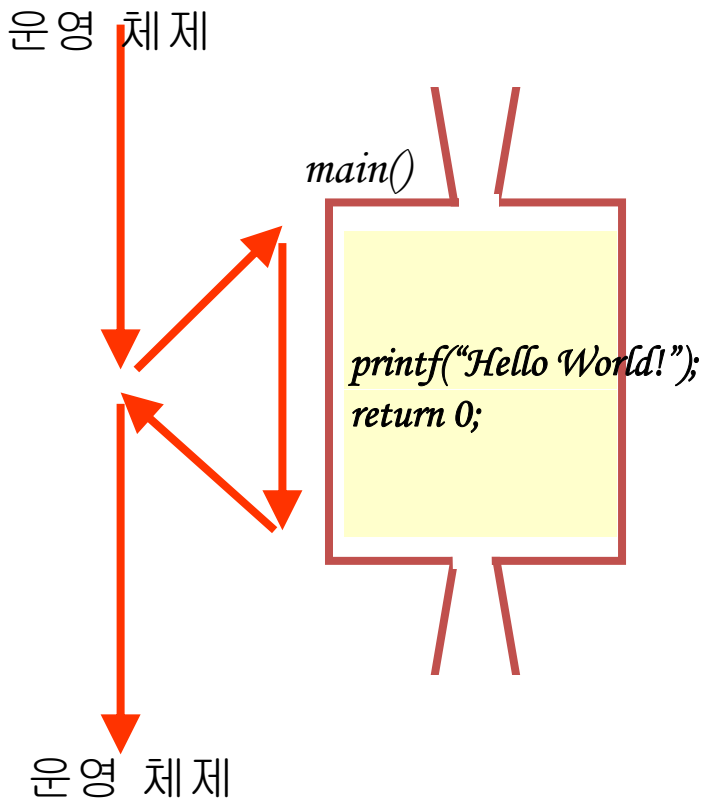
- printf()는 컴파일러가 제공하는 함수로서 출력을 담당합니다.
- 큰따옴표 안의 문자열을 화면에 출력합니다.



함수 반환문

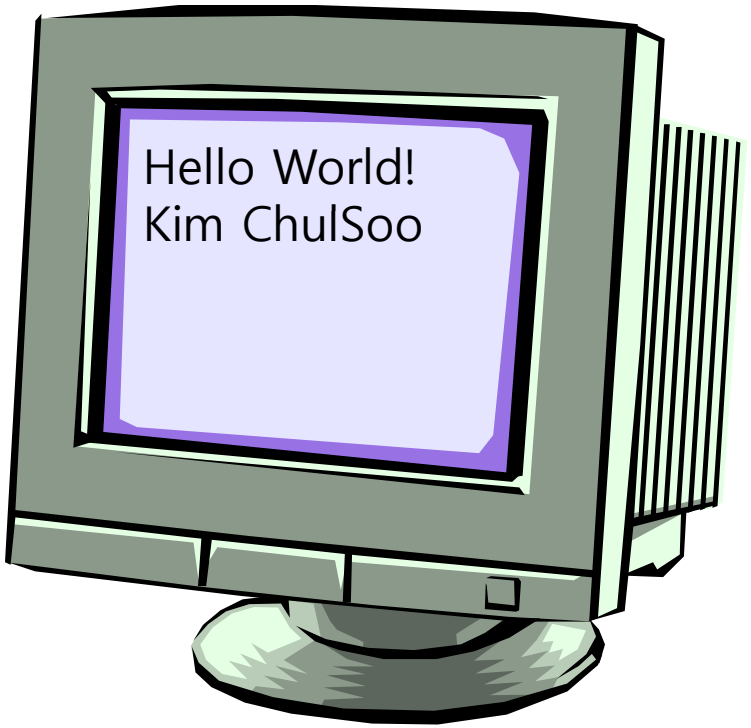
```
return 0;
```

- return은 함수의 결과값을 외부로 반환합니다.



응용 프로그램 #1

- 다음과 같은 출력을 가지는 프로그램을 제작하여 보자.



첫번째 버전

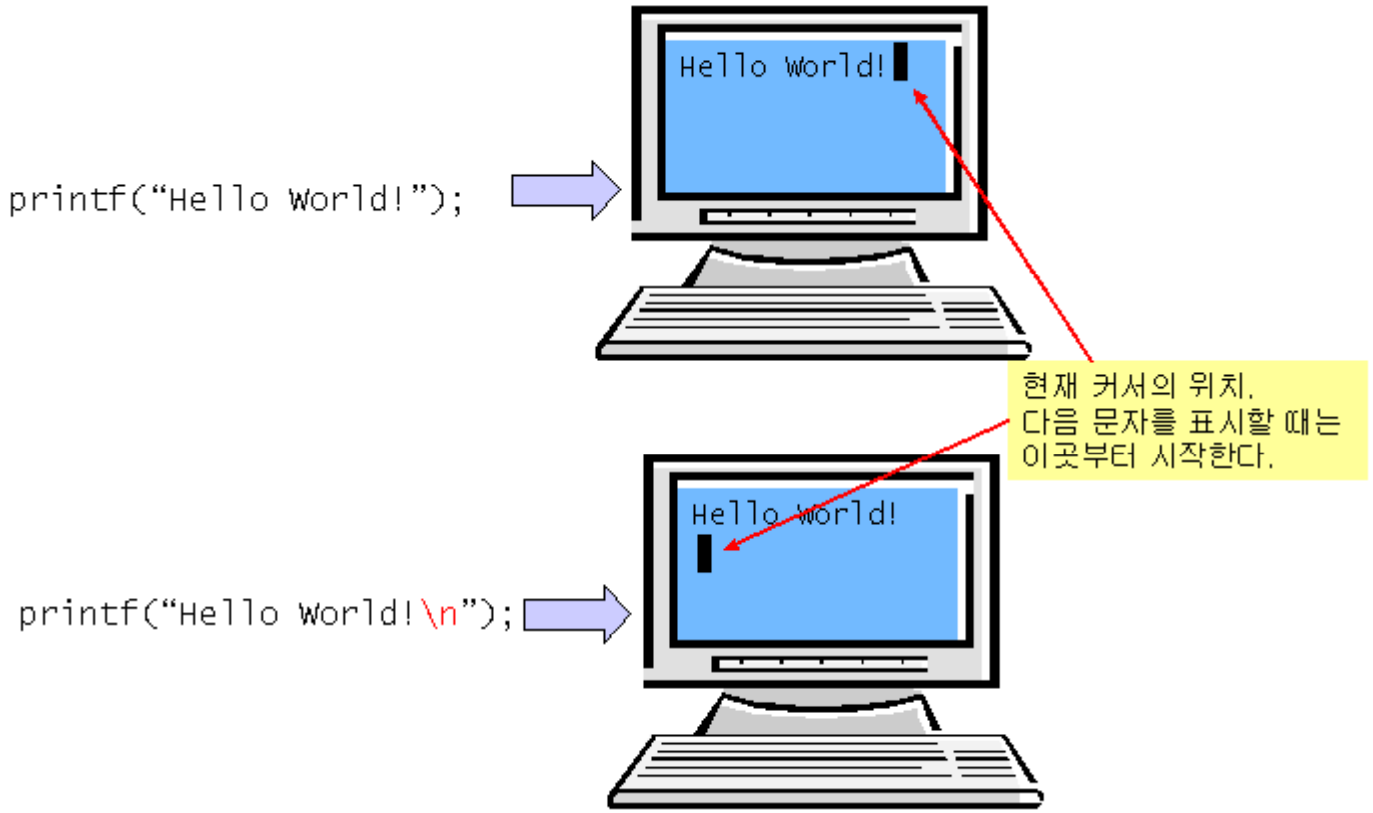
- 문장들은 순차적으로 실행된다는 사실 이용

```
/* 첫번째 프로그램*/
#include <stdio.h>

int main(void)
{
    printf("Hello World!");
    printf("Kim ChulSoo");
    return 0;
}
```

```
Hello World!Kim ChulSoo
```

줄바꿈 문자 ₩n

- 줄바꿈 문자인 ₩n은 화면에서 커서는 다음줄로 이동하게 한다.



변경된 프로그램

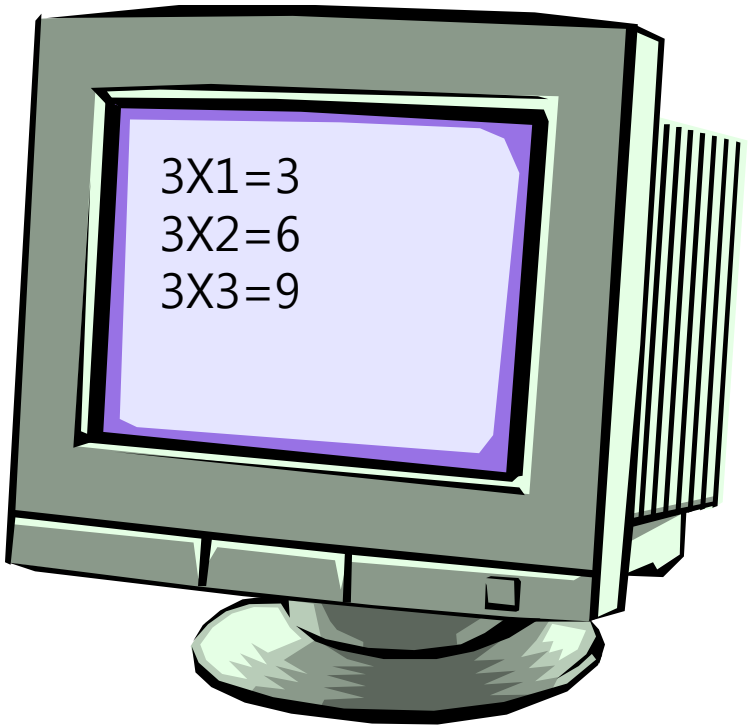
- 줄바꿈 문자를 포함하면 우리가 원하던 결과가 된다.

```
/* 첫번째 프로그램*/
#include <stdio.h>

int main(void)
{
    printf("Hello World!\n");
    printf("Kim ChulSoo");
    return 0;
}
```

```
Hello World!
Kim ChulSoo
```

응용 프로그램 #2

- 다음과 같은 출력을 가지는 프로그램을 제작하여 보자.



응용 프로그램

- 역시 문장들은 순차적으로 수행된다는 점을 이용한다.

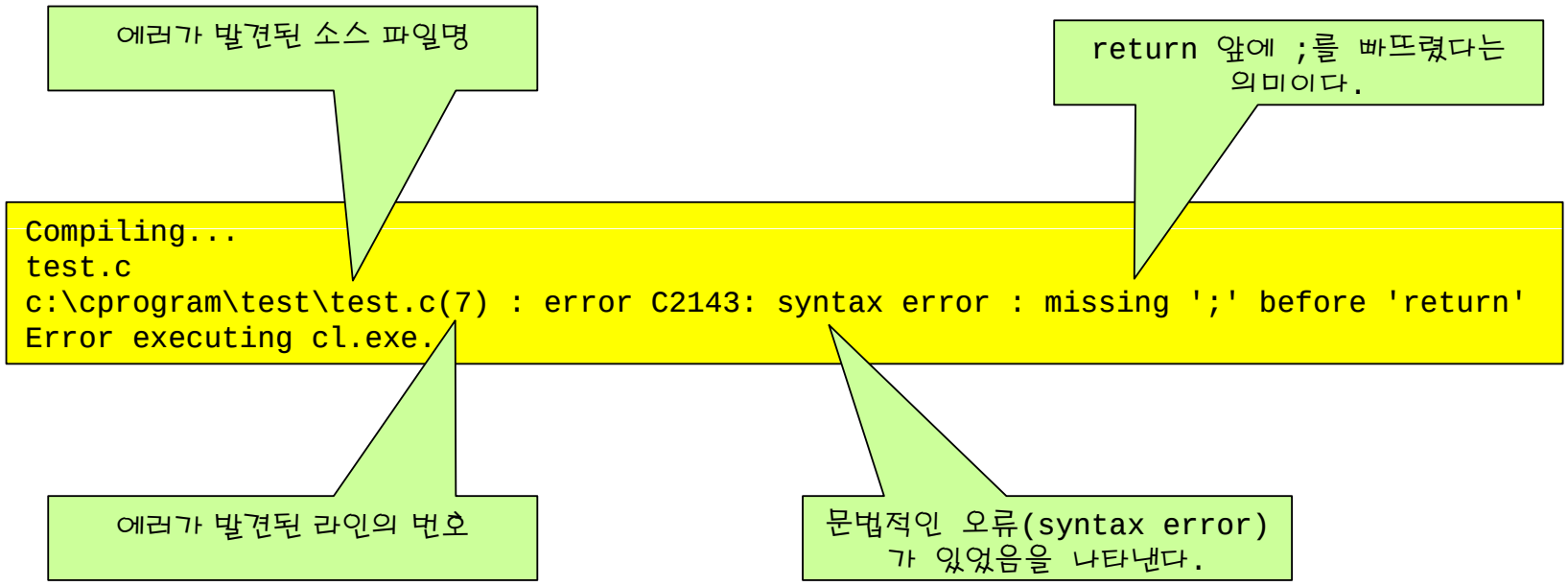
```
/* 첫번째 프로그램의 응용*/
#include <stdio.h>

int main(void)
{
    printf("3 X 1 = 3\n");
    printf("3 X 2 = 6\n");
    printf("3 X 3 = 9\n");

    return 0;
}
```

오류 수정 및 디버깅

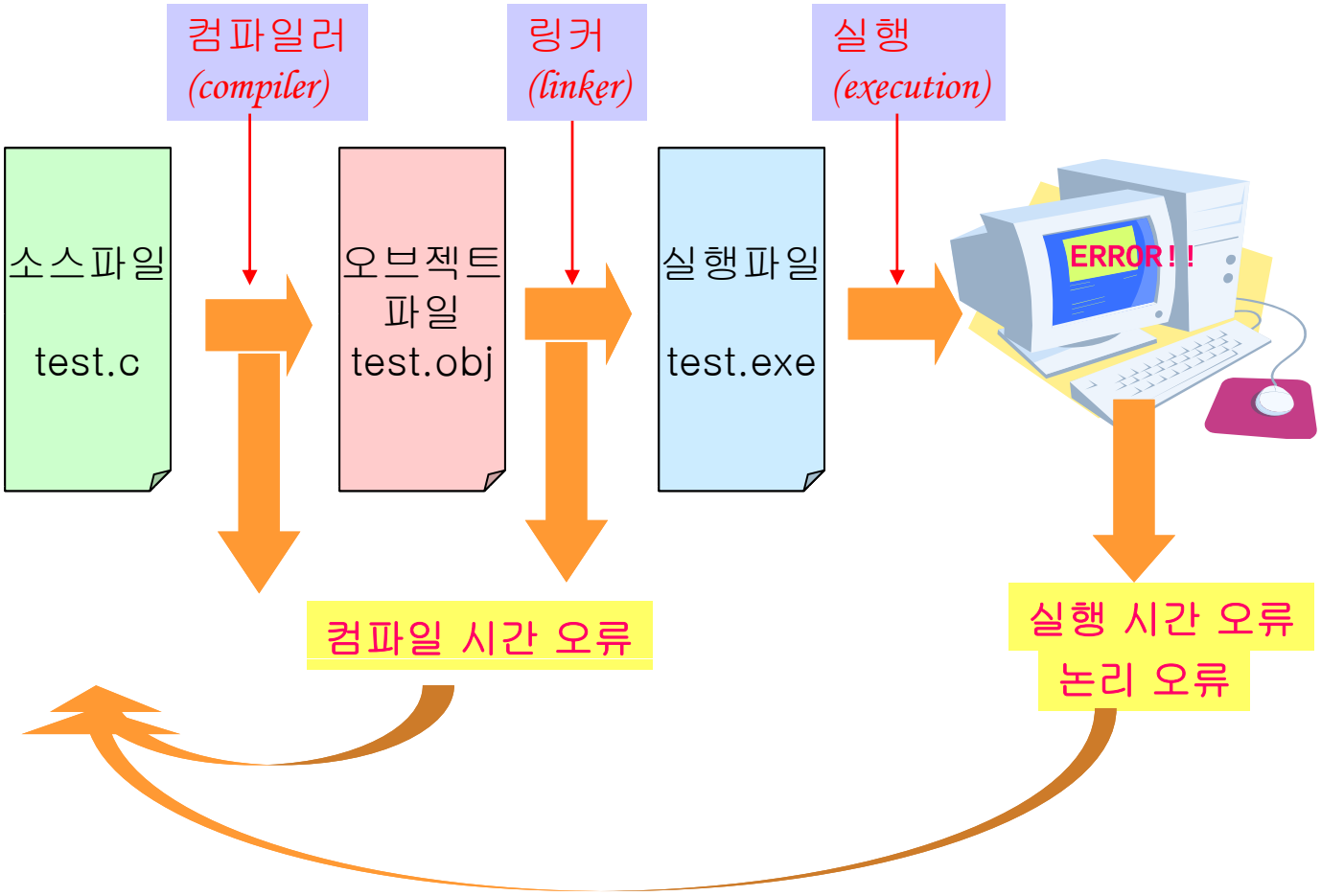
- 컴파일이나 실행 시에 오류가 발생할 수 있다.
- 에러와 경고
  - 에러(error): 심각한 오류
  - 경고(warning): 경미한 오류

- 오류의 종류
  - 컴파일 시간 오류: 대부분 문법적인 오류
  - 실행 시간 오류: 실행 중에 0으로 나누는 연산 같은 오류
  - 논리 오류: 논리적으로 잘못되어서 결과가 의도했던 대로 나오지 않는 오류

오류 메시지의 분석


에러가 발견된 소스 파일명
return 앞에 ;를 빠뜨렸다는 의미이다.
Compiling...
test.c
c:\cprogram\test\test.c(7) : error C2143: syntax error : missing ';' before 'return'
Error executing cl.exe.
에러가 발견된 라인의 번호
문법적인 오류(syntax error)가 있었음을 나타낸다.

오류 수정 과정


컴파일러
(compiler)
링커
(linker)
실행
(execution)
소스파일
test.c
오브젝트
파일
test.obj
실행파일
test.exe
ERROR!!
컴파일 시간 오류
실행 시간 오류
논리 오류

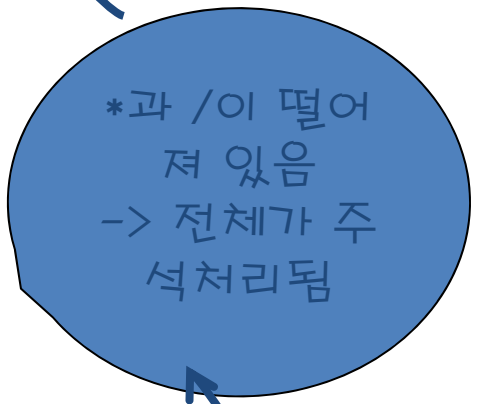
오류 #1

```c
/* 에러가 발생하는 프로그램 */
#include <stdio.h>

int main(void)
{
    printf("Hello World!\n")
    return 0;
}
```


```
--------------------Configuration: test - Win32 Debug--------------------
Compiling...
test.c
C:\PROJECT\test\test.c(7) : error C2143: syntax error : missing ';' before 'return'
Error executing cl.exe.

test.exe - 1 error(s), 0 warning(s)
```

오류 #2

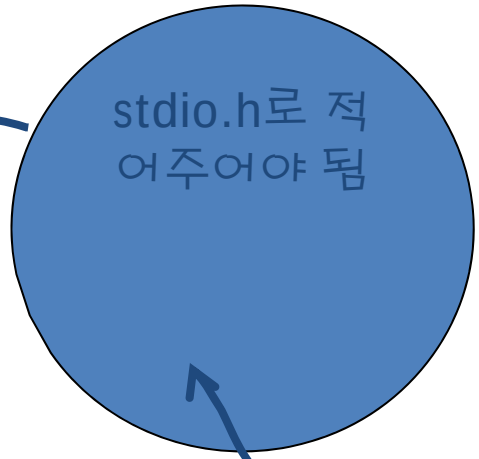
```
/* 에러가발생하는프로그램* /
#include <stdio.h>

int main(void)
{
    printf("Hello World!\n")
    return 0;
}
```




```
--------------------Configuration: test - Win32 Debug--------------------
Compiling...
test.c
c:\project\test\test.c(9) : fatal error C1071: unexpected end of file found in
comment
Error executing cl.exe.

test.exe - 1 error(s), 0 warning(s)
```

오류 #3

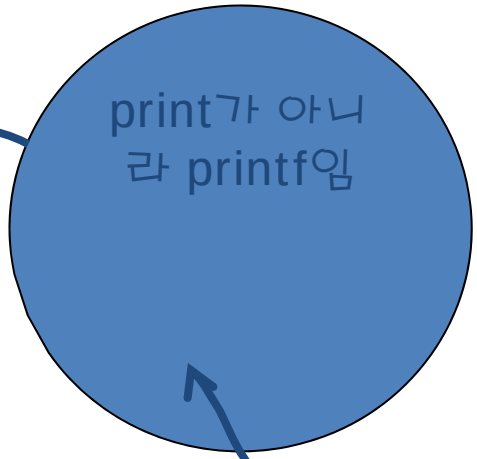
```
/* 첫번째 프로그램*/
#include <stdio,h>

int main(void)
{
    print("Hello World!");
    return 0;
}
```




```
--------------------Configuration: test - Win32 Debug--------------------
Compiling...
test.c
c:\project\test\test.c(2) : fatal error C1083: Cannot open include file: 'stdio,h':
No such file or directory
```

오류 #4

```
/* 첫번째프로그램*/
#include <stdio.h>

int main(void)
{
    print("Hello World!");
    return 0;
}
```




```
------------------Configuration: test - Win32 Debug-------------------
Compiling...
test.c
C:\CPROGRAM\test\test.c(6) : warning C4013: 'print' undefined; assuming extern
returning int
Linking...
test.obj : error LNK2001: unresolved external symbol _print
Debug/test.exe : fatal error LNK1120: 1 unresolved externals
Error executing link.exe.

test.exe - 2 error(s), 1 warning(s)
```

논리 오류

- 다음과 같은 출력을 가지는 프로그램을 작성하여 보자.

논리 오류가 존재하는 프로그램

```
/* 첫번째 프로그램*/
#include <stdio.h>

int main(void)
{
    printf("Hello World!\n");        // ①
    printf("Good Morning\n");
    return 0;
}
```


```
Hello World!
Good Morning
```

논리 오류가 수정된 프로그램

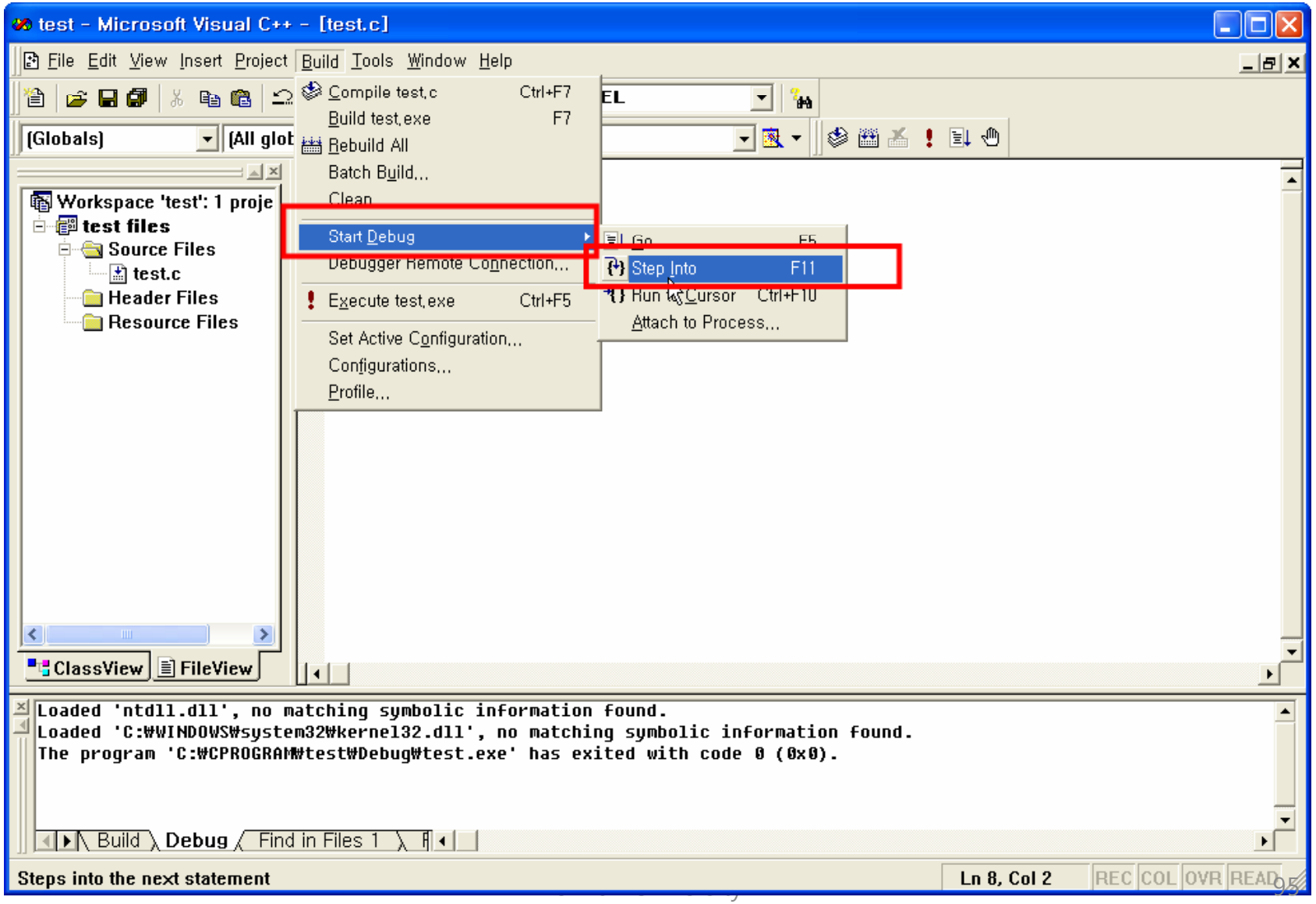
```c
/* 첫번째 프로그램*/
#include <stdio.h>

int main(void)
{
    printf("Hello World! ");          // ①
    printf("Good Morning\n");
    return 0;
}
```


```
Hello World!  Good Morning
```

디버깅

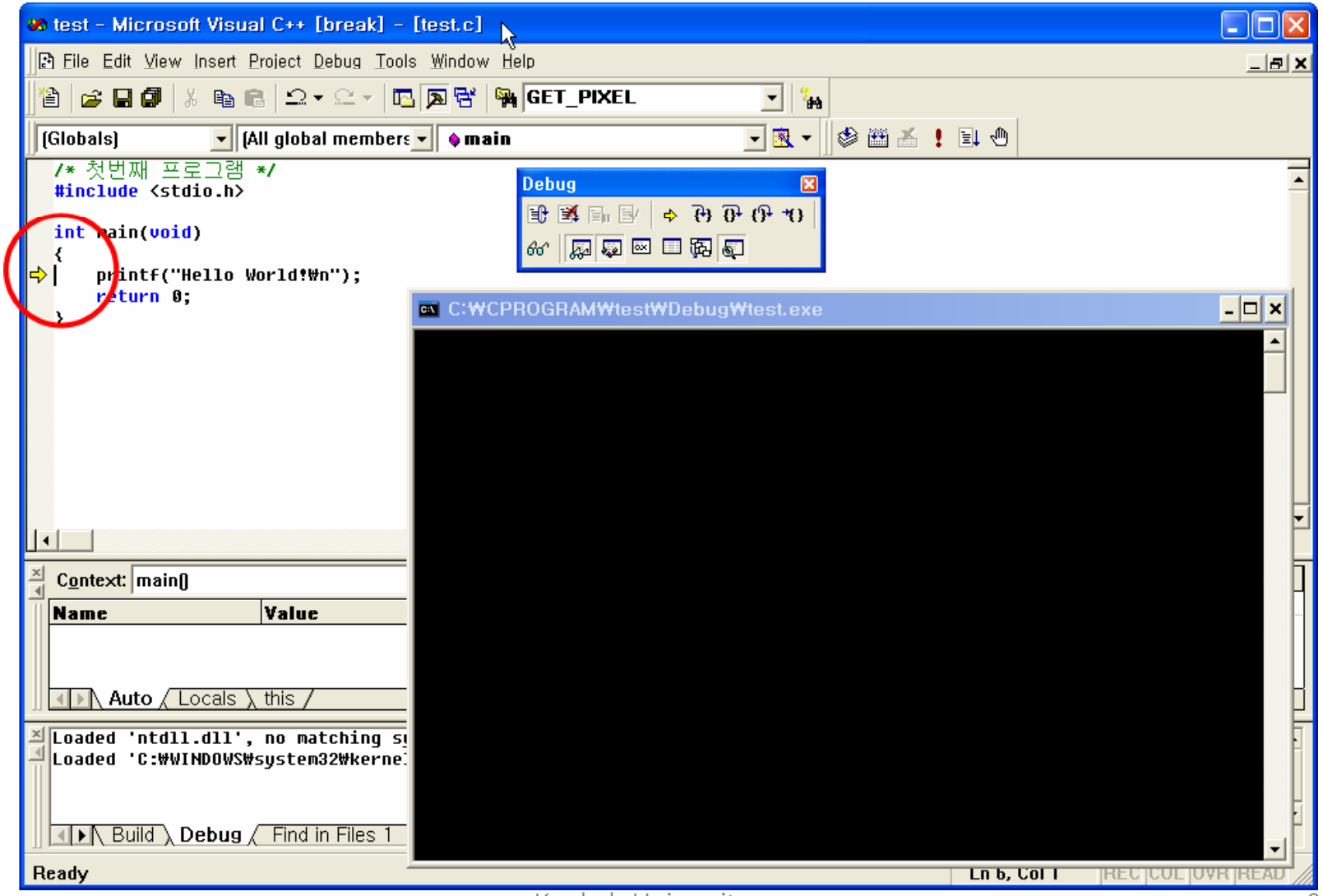
- 디버깅: 논리 오류를 찾는 과정

디버거(debugger)

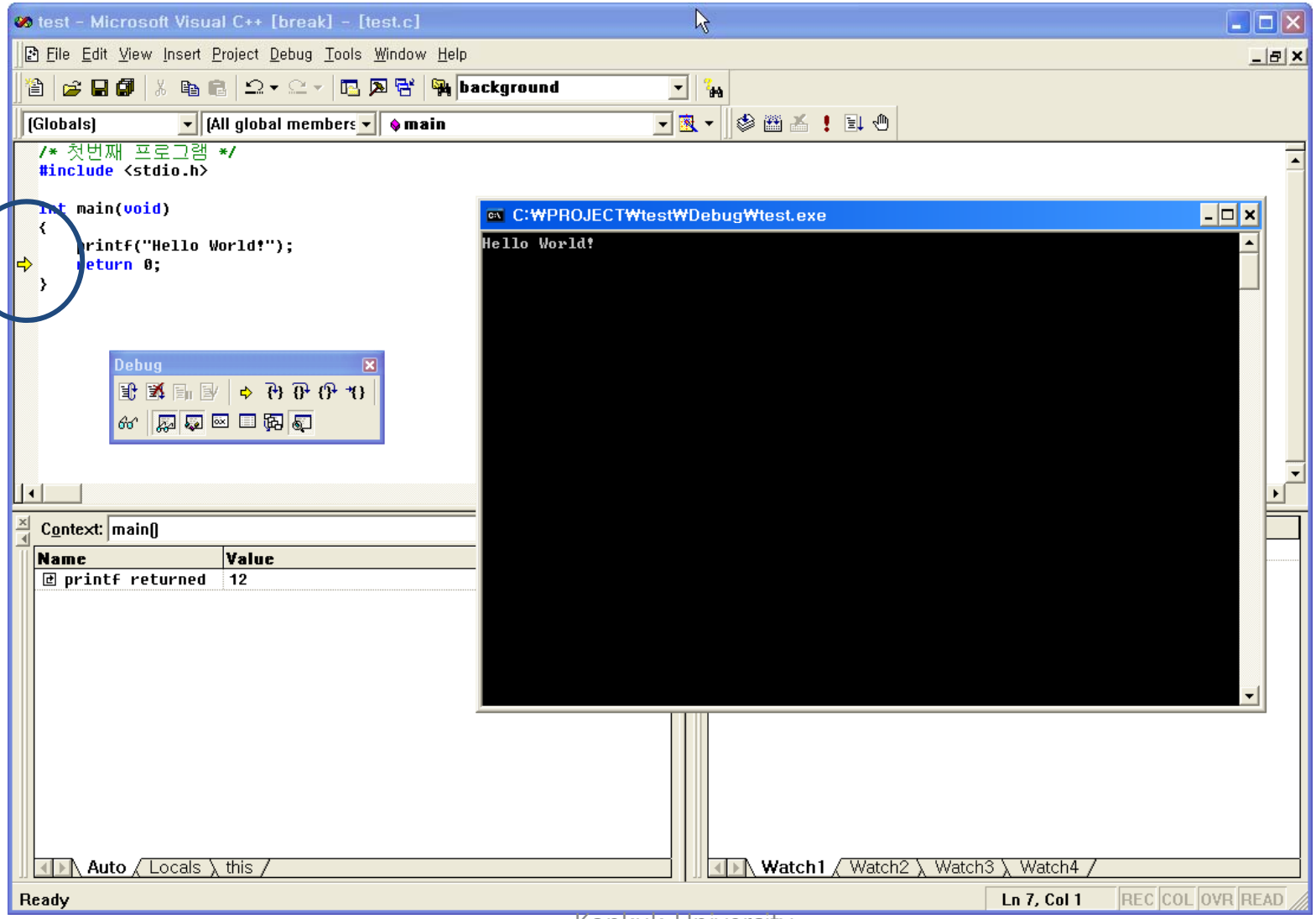


디버거의 실행 과정

F10



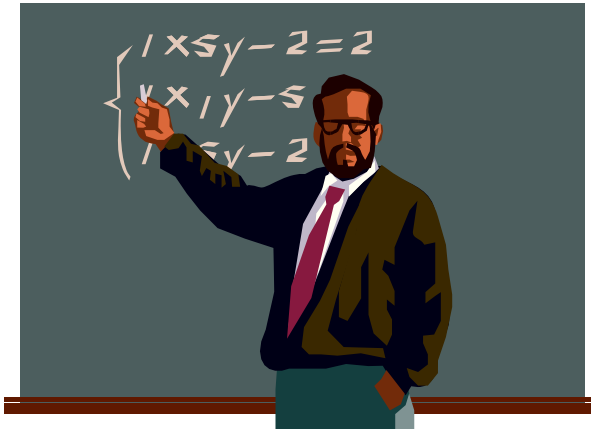
디버거의 실행 과정


F10
test - Microsoft Visual C++ [break] - [test.c]
File Edit View Insert Project Debug Tools Window Help
background
(Globals)
(All global members
main
/* 첫번째 프로그램 */
#include <stdio.h>
int main(void)
{
printf("Hello World!");
return 0;
}
Debug
C:₩PROJECT₩test₩Debug₩test.exe
Hello World!
Context: main()
Name
Value
printf returned
12
Auto
Locals
this
Watch1
Watch2
Watch3
Watch4
Ready
Ln 7, Col 1
REC COL OVR READ

디버거의 명령어 정의

- F5 (Go): 실행
- F10 (Step Over): 한 문장씩 실행(함수도 하나의 문장 취급)
- F11 (Step Into): 한 문장씩 실행(함수 안으로 진입)
- F9 (Breakpoint): 현재 문장에 중단점을 설정

Q & A



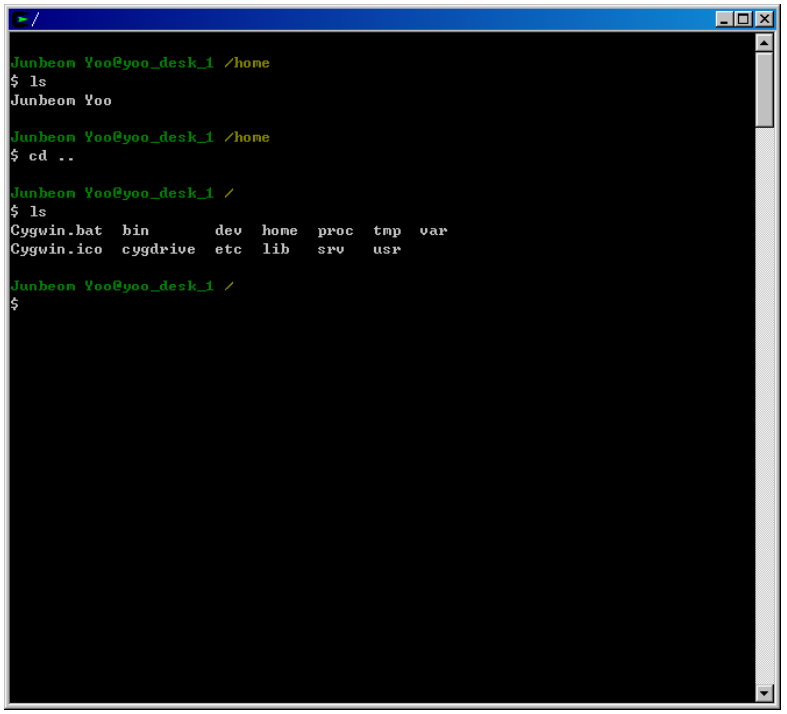
LINUX 환경에서의 프로그래밍

Linux 환경

- Cygwin
  - Open SW
  - PC MS Window 환경
  - MS Window와 File System을 공유
  - 다수의 terminal 실행 가능

  - 설치 시 주의 사항
    - Dev, Shell 패키지 설치.
    - 설치 후, 기본설정 필요.

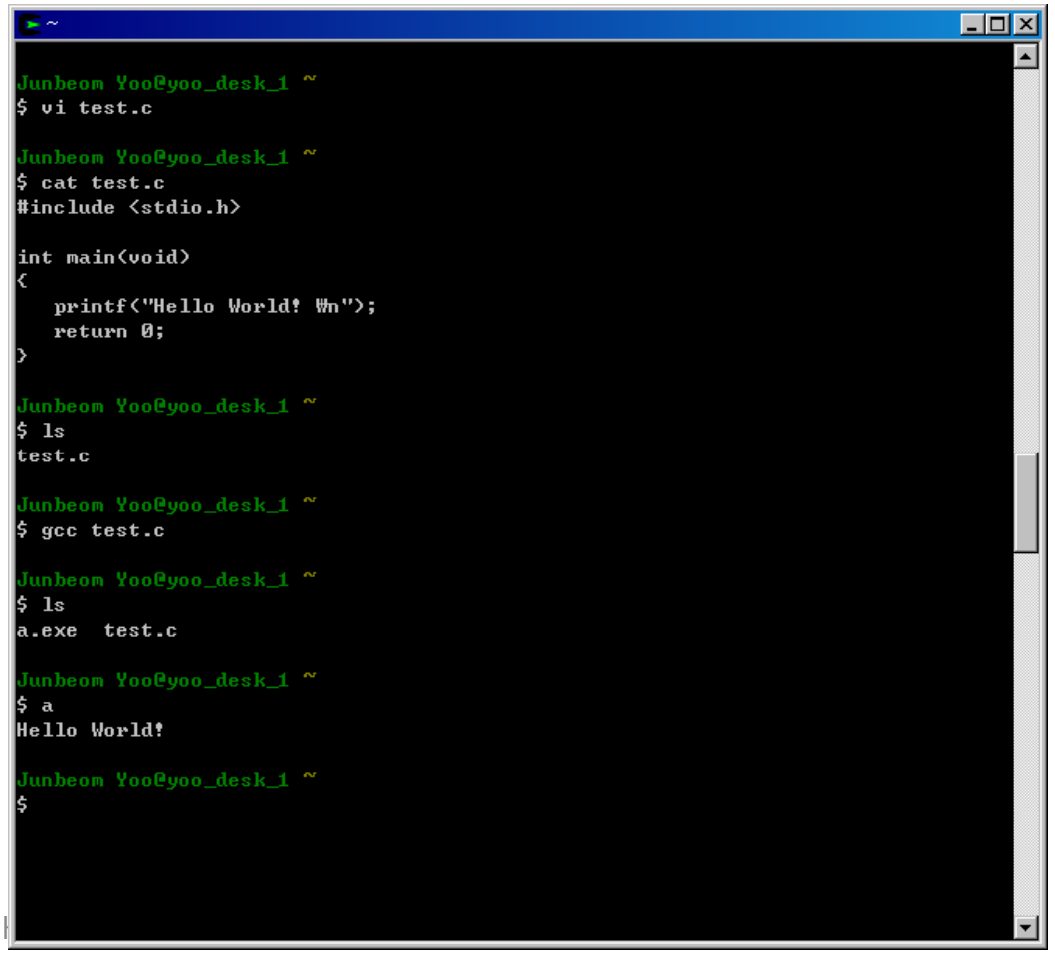
    - .bash_profile 을 Notepad로 열고 마지막에 PATH=${HOME}/bin:${PATH}:/home/"자신의 Directory" 을 추가합니다.
    - 다음으로 .source .bash_profile 을 실행합니다.



C 프로그램 컴파일 및 실행

- 프로그램 작성
  - vi test.c
- 프로그램 컴파일
  - gcc test.c
  - gcc test.c –o aaa
- 프로그램 실행
  - a
  - a.exe
  - aaa

```c
/* 첫번째 프로그램*/
#include <stdio.h>

int main(void)
{
	printf("Hello World!");
	return 0;
}
```



vi 에디터 사용법

vi 시작	
vi filename	파일열기, 작성
vi +18 filename	18행으로 파일 열기
vi +/"string" fn	"string"의 처음 발생 단어부터
vi -r filename	손상된 파일 회복
view filename	읽기 전용으로 파일 열기
커서명령(이동)	
h(←)	왼쪽으로 커서 이동
j(↓)	아래로 커서 이동
k(↑)	위로 커서 이동
l(→)	오른쪽으로 커서 이동
w	한 단어 오른 쪽으로 커서 이동
b	한 단어 왼쪽으로 커서 이동
Return	한 행 아래로 커서 이동
Back Space	한 문자 왼쪽으로 커서 이동
Space Bar	한 문자 오른 쪽으로 커서 이동
H	화면의 맨위로 이동
M	화면의 중간으로 이동
L	화면의 맨 아래로 이동
Ctrl + F	한 화면 앞으로 이동
Ctrl + D	반 화면 앞으로 이동
Ctrl + B	한 화면 뒤로 이동
Ctrl + U	반 화면 뒤로 이동
문자와 행 삽입	
a	커서 오른쪽에 문자 삽입
A	커서 오른쪽, 행의 끝에 문자 삽입
i	커서 왼쪽에 문자 삽입
I	커서 왼쪽, 행의 처음에 문자 삽입
o	커서 아래에 행 삽입
O	커서 위에 행 삽입
텍스트 변경	
cw (종료:ESC)	단어변경
cc (종료:ESC)	행 변경
C (종료:ESC)	커서 오른쪽의 행 변경
s (종료:ESC)	커서가 위치한 문자열 대체
r	커서 위치의 문자를 다른 문자로 대체
r - Return	행 분리
J	현재 행과 아래 행 결합
xp	커서 위치 문자와 오른쪽 문자 교환
~	문자형(대.소문자)변경
u	이전 명령 취소
U	행 변경 사항 취소
:u	이전의 최종 행 취소
.	이전 최종 명령 반복

텍스트 삭제	
x	문자 삭제
dw	단어 삭제
dd	행 삭제
D	커서 오른쪽 행 삭제
:5,10 d	5-10 번째 행 삭제
텍스트 복사 및 이동	
yy	행 yank 또는 복사
Y	행 yank 또는 복사
dd	행 삭제
P	yank되거나 삭제된 행 현재 행 위에 삽입
p	yank되거나 삭제된 행 현재 행 아래에 삽입
:1,2 co 3	1-2행을 3행 다음으로 복사
:4,5 m 6	4-5행을 6행 위로 이동
행 번호 설정	
:set nu	행 번호 표시
:set nonu	행 번호 숨기기
행 찾기	
G	파일의 마지막 행으로 가기
21G	파일의 21번째 행을 가기
탐사 및 대체	
/string/	string 탐색
?string?	string 역방향 탐색
n(N)	string의 다음(이전) 계속 탐색
:g/search-string/s//replace-string/gc	
	각 발생 탐색 후 확인하고 대체
:s/srt/rep	현재 행의 str을 rep로 대체
:1,.s/str/rep/	1부터 현재 행의 str을 rep로 대체
:%s/str/rep/g	파일 전체 str을 rep로 전부 대체
화면정리	
:Ctrl-l	불필요한 화면정리 후 다시 표시
파일을 파일로 삽입	
:r filename	커서 다음에 파일 삽입
:34 r filename	파일을 34번째 행 다음에 삽입
보관 및 종료	
:w	변경사항 보관
:w filename	버퍼를 파일로 보관
:wq	변경사항 보관 후 vi 종료
ZZ	변경사항 보관 후 vi 종료
:q!	변경사항 보관하지 않고 종료