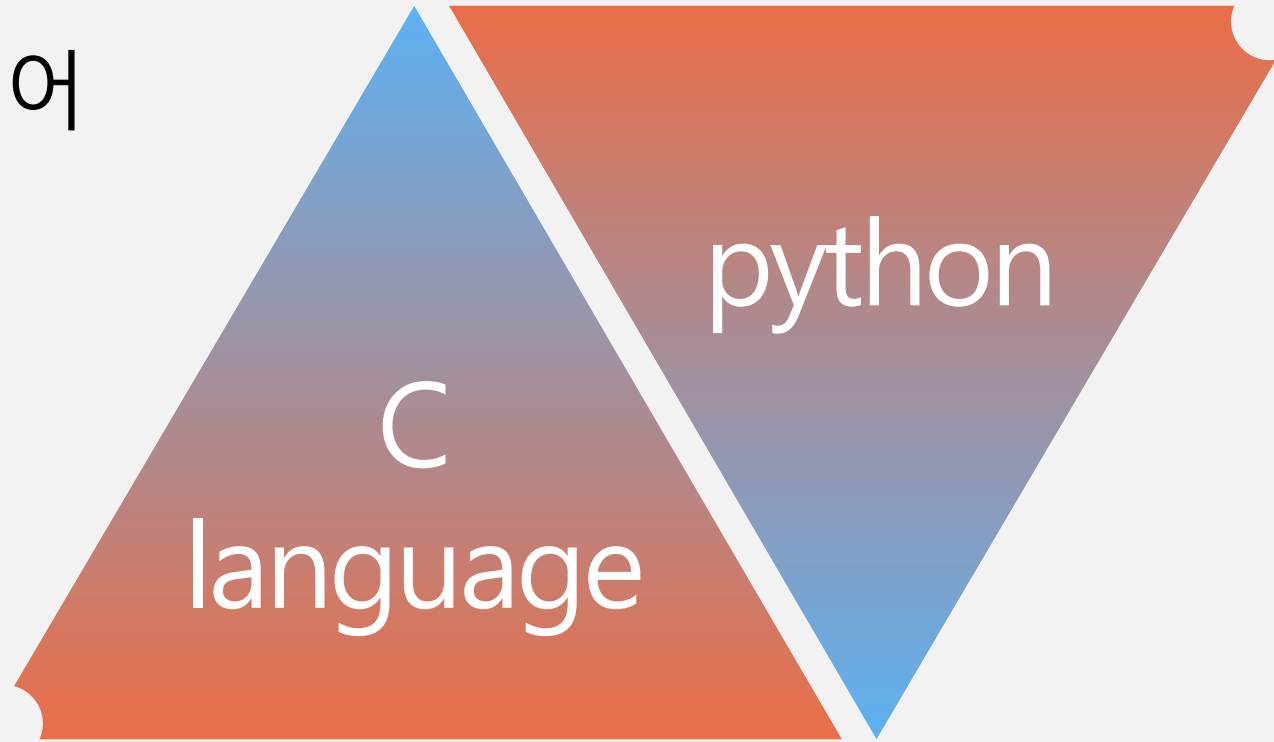




C 언어
멘토링 수업 ●
(2주차)

- 1 컴퓨터 소프트웨어 학부 소개
 - 2 지난 주 수업 복습
 - 3 C Hello World!
 - 4 printf 와 scanf
-

1. 컴퓨터 소프트웨어 학부 소개

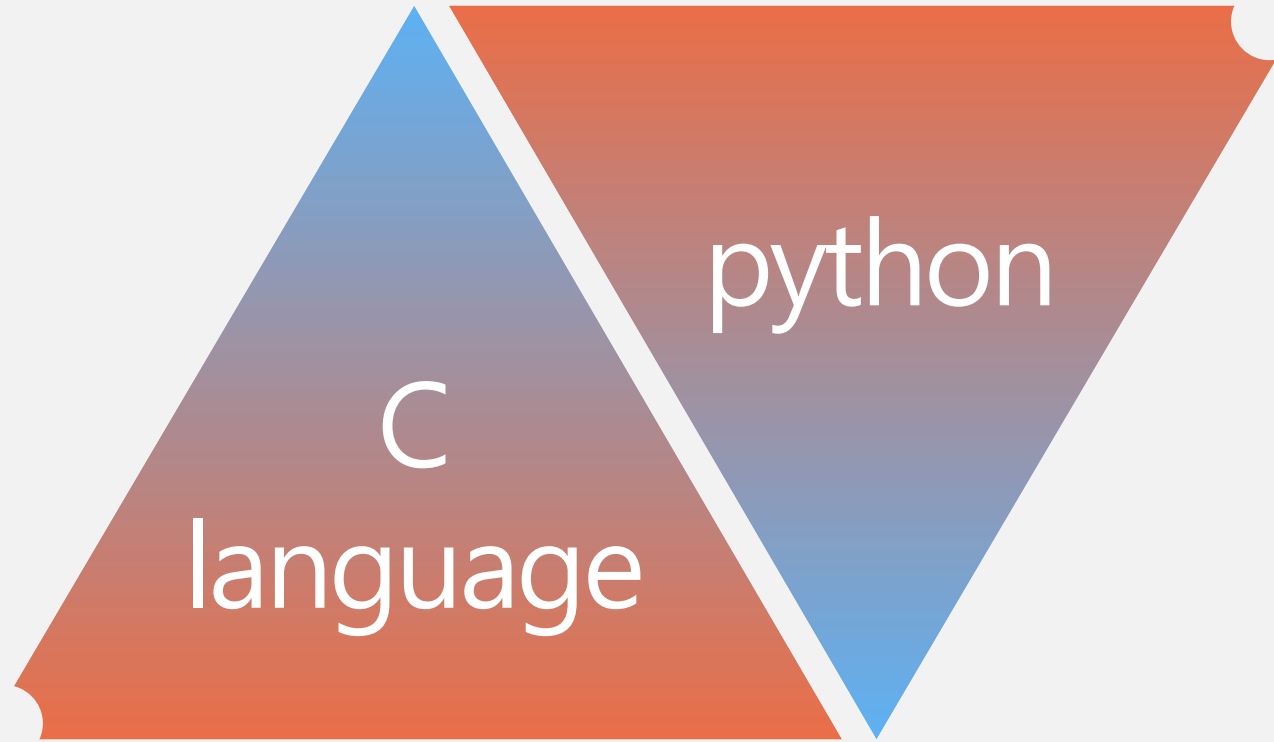




컴퓨터소프트웨어

한양대학교
컴퓨터소프트웨어 학부
이현민

2. 지난주 수업 복습



이주호



2. Data와 Code.pdf



12. 순차, 분기, 반

임을규

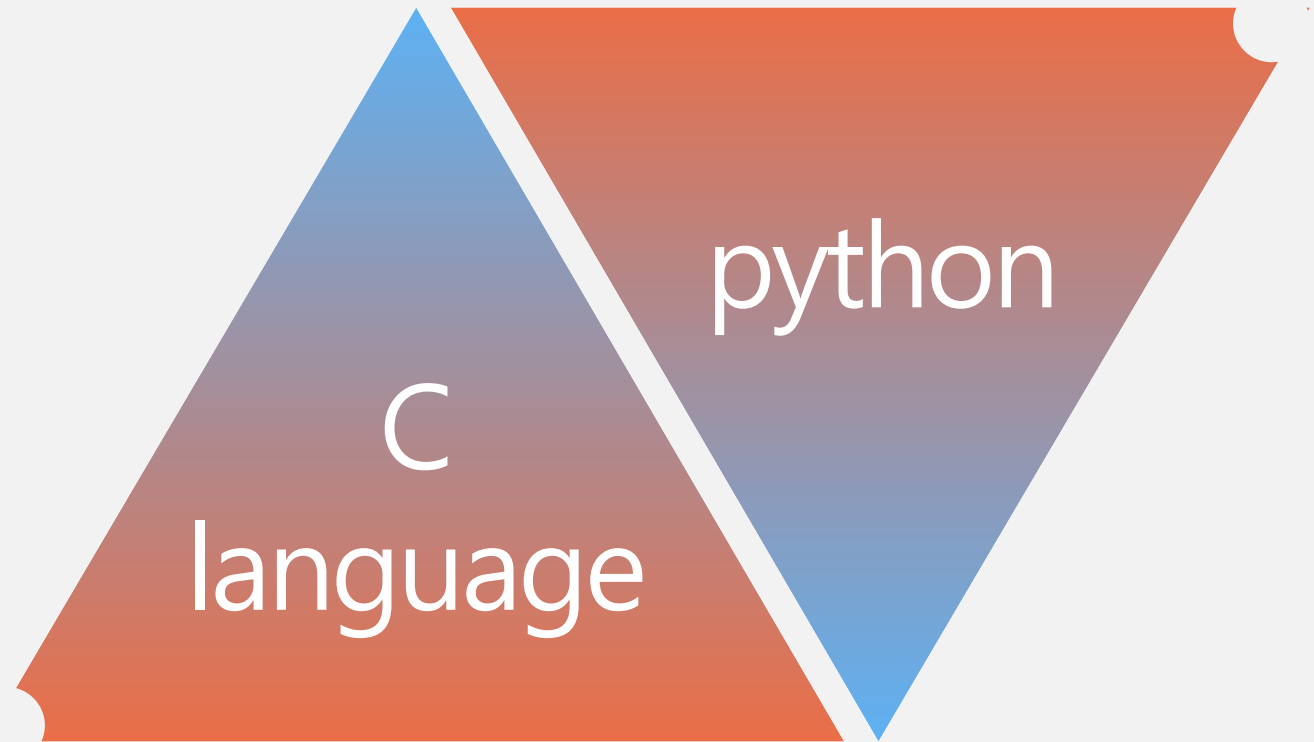


01. Introduction.pdf

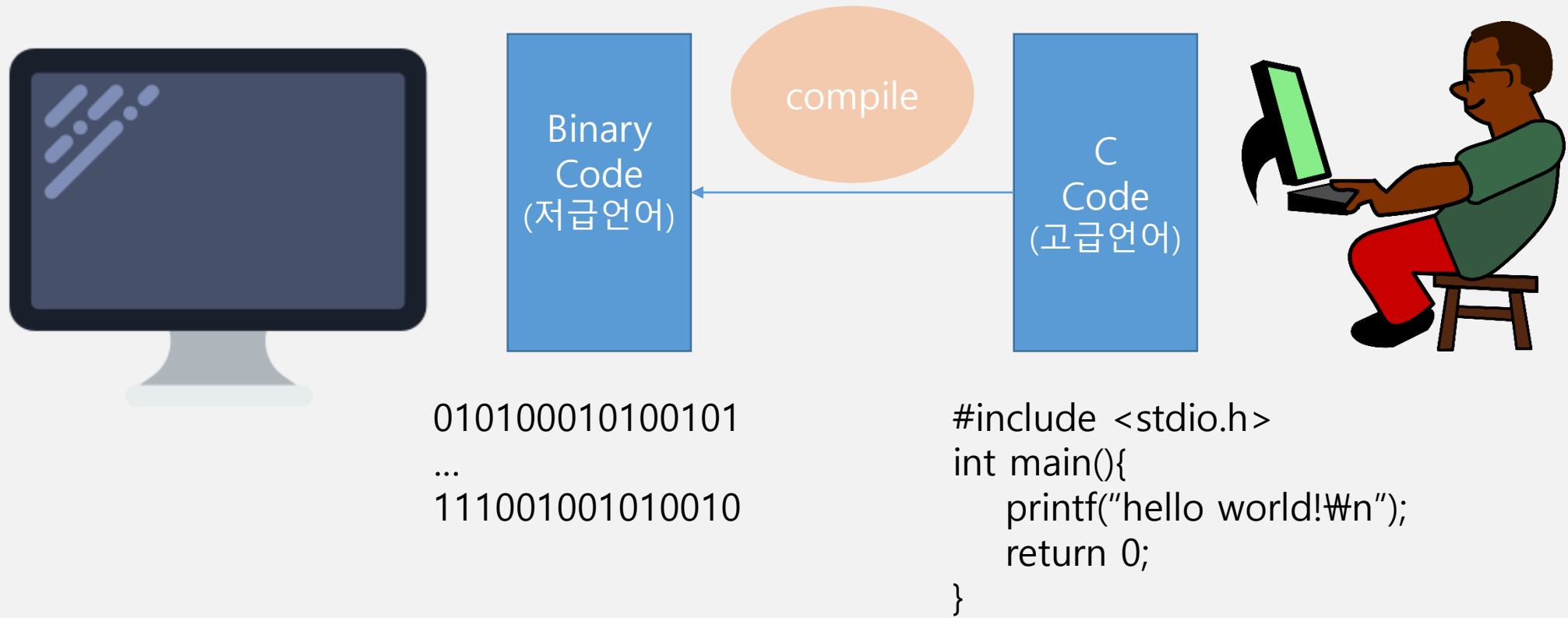


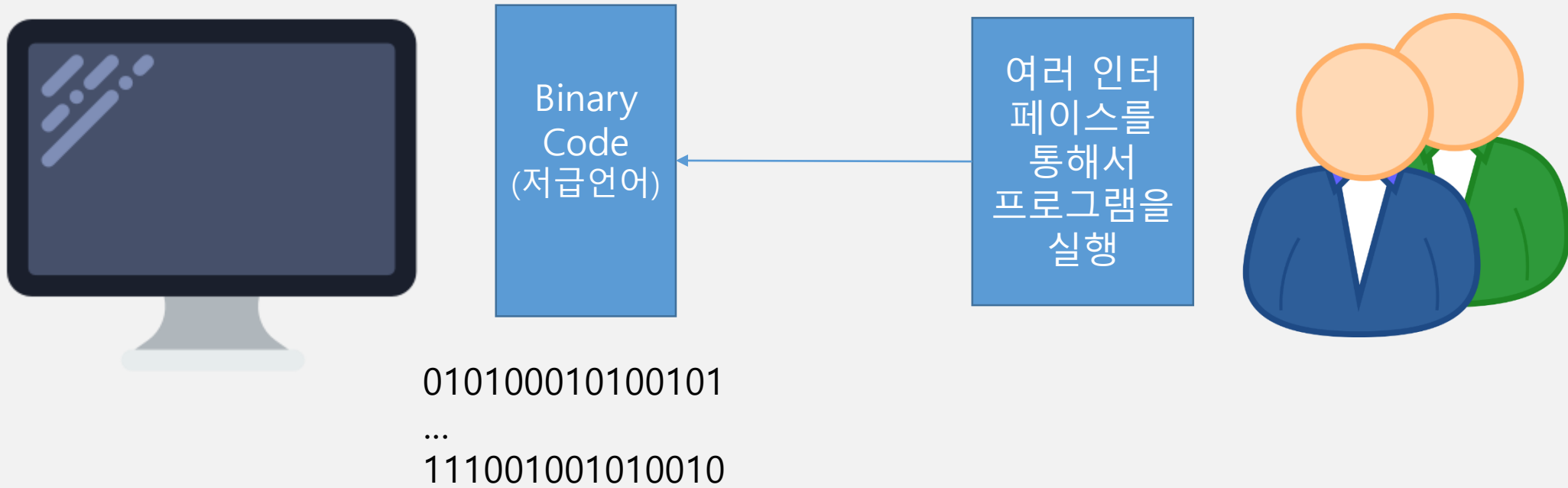
02. Guess the Number(print).pdf

3. C Hello World!

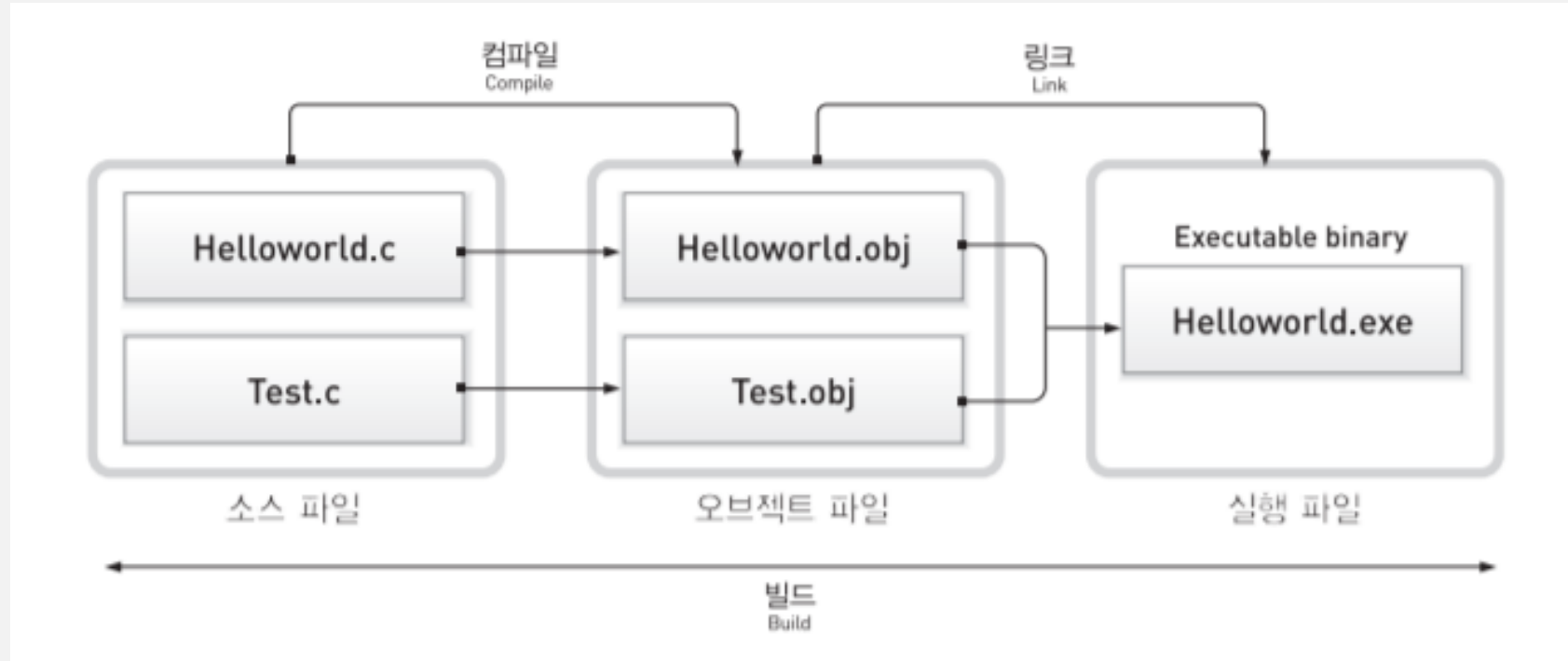


3 C Hello World!





3 C Hello World!

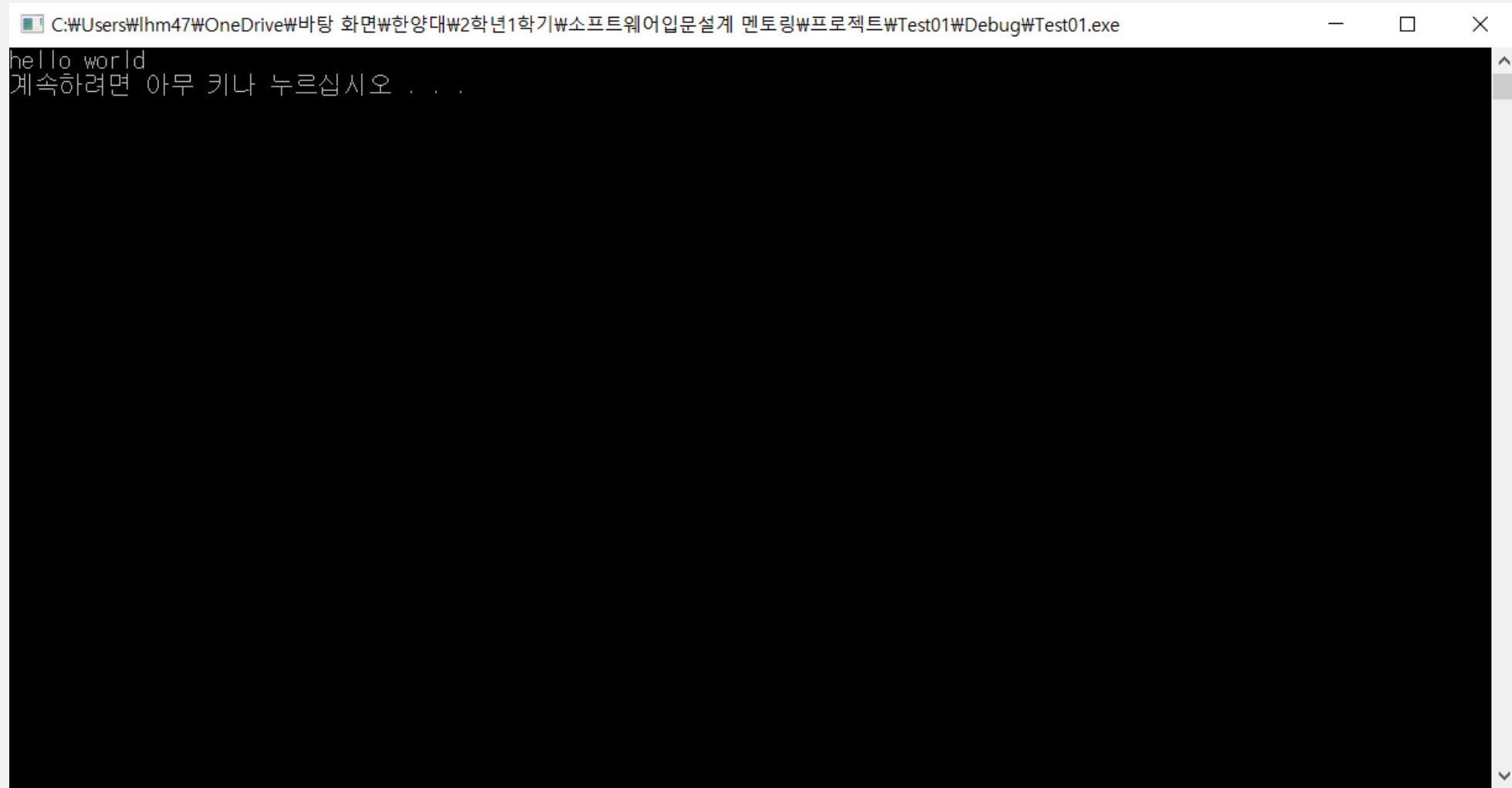


```
1  #include<stdio.h>
2  int main() {
3      printf("hello world\n");
4
5      system("pause");
6      return 0;
7  }
```



Ctrl + F5
[build]

3 C Hello World!



A screenshot of a Windows command prompt window. The title bar at the top shows the file path: C:\Users\lhm47\OneDrive\바탕 화면\한양대\2학년1학기\소프트웨어입문설계 멘토링\프로젝트\Test01\Debug\Test01.exe. The window has standard minimize, maximize, and close buttons. The command prompt area is black with white text. The first line displays 'hello world'. The second line displays Korean text: '계속하려면 아무 키나 누르십시오 . . .'. A vertical scrollbar is visible on the right side of the command prompt area.

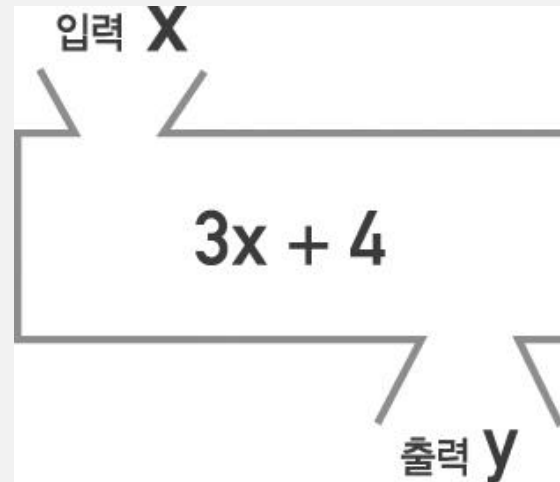
```
C:\Users\lhm47\OneDrive\바탕 화면\한양대\2학년1학기\소프트웨어입문설계 멘토링\프로젝트\Test01\Debug>Test01.exe  
hello world  
계속하려면 아무 키나 누르십시오 . . .
```

```
/* Hello.c */  
#include <stdio.h>  
  
int main(void)  
{  
    printf("Hello, World! \n");  
    return 0;  
}
```

함수(function)에 대한 이해

적절한 입력과 그에 따른 출력이 존재 하는 것을 가리켜
함수라 한다.

C 언어의 기본 단위는 함수이다.



함수 호출(func call)과 인자(arg,param) 전달

인자 전달 : 입력 x를 전달하는 행위

함수 호출 : 인자를 전달하면서 함수의 실행을 요구하는 행위

C 언어의 함수 특성

입력과 출력 존재

순차적으로 실행

함수의 기능을 정의하는 몸체 부분 존재

예제 Hello.c에서의 함수



세미콜론(;)이 필요한 문장

연산을 수행하는 문장 : 시간의 흐름에 따라서 컴퓨터에게 "이러 이러한 일을 해라"라고 명령을 하는 문장

표준 라이브러리에 대한 이해

이미 표준화 해서 만들어 놓은 함수들의 집합을 가리켜 표준 라이브러리라 한다.

헤더 파일을 포함해야 사용이 가능하다.

헤더 파일의 이해

`stdio.h` 라는 이름의 헤더 파일

헤더 파일의 포함을 알리는 선언은 제일 먼저 등장
해야 한다.

The diagram illustrates the structure of a C program. It is divided into two distinct regions, each highlighted with a speech bubble. The top region, labeled '첫번째 영역' (First Area), contains the preprocessor directive `#include <stdio.h>`. The bottom region, labeled '두번째 영역' (Second Area), contains the `main` function definition: `int main(void)`, `{`, `printf("Hello, World! \n");`, `return 0;`, and `}`. The regions are separated by a horizontal line, and the entire code block is enclosed in a light gray border.

```
#include <stdio.h>

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

return의 의미

함수를 종료(빠져 나온다).

함수를 호출한 영역으로 값을 반환

return의 특징

return은 함수 내에서 존재 하지 않을 수도 있다.

둘 이상의 return문이 존재하는 것도 가능

주석이란?

프로그래머에게 메모(memo)의 기능을 부여

컴파일러는 주석을 없는 것으로 간주

주석을 삽입 함으로 인해 프로그램의 가독성 증가

선택이 아닌 필수!

주석의 두 가지 형태

여러 줄에 걸친 주석 처리

```
/* 한 줄 짜리 주석 */
```

```
/*  
여러 줄에  
걸친 주석  
*/
```

단일 행 주석 처리

```
// 주석 하나.  
// 주석 둘.  
// 주석 셋.
```

주석의 예

```
#include <stdio.h>                // stdio.h 헤더 파일 포함

int main(void)                    // main 함수의 시작
{
    /*
        printf 함수는 모니터로 출력을 하는 경우에 쓴다.
        인자로 문자열을 전달하면 문자열을 출력한다.
    */
    printf("Hello World! \n");      //모니터로 문자열 출력
    return 0;                      // 0을 반환한다.
}                                  // main 함수의 끝
```

주석 처리에 있어서의 주의점

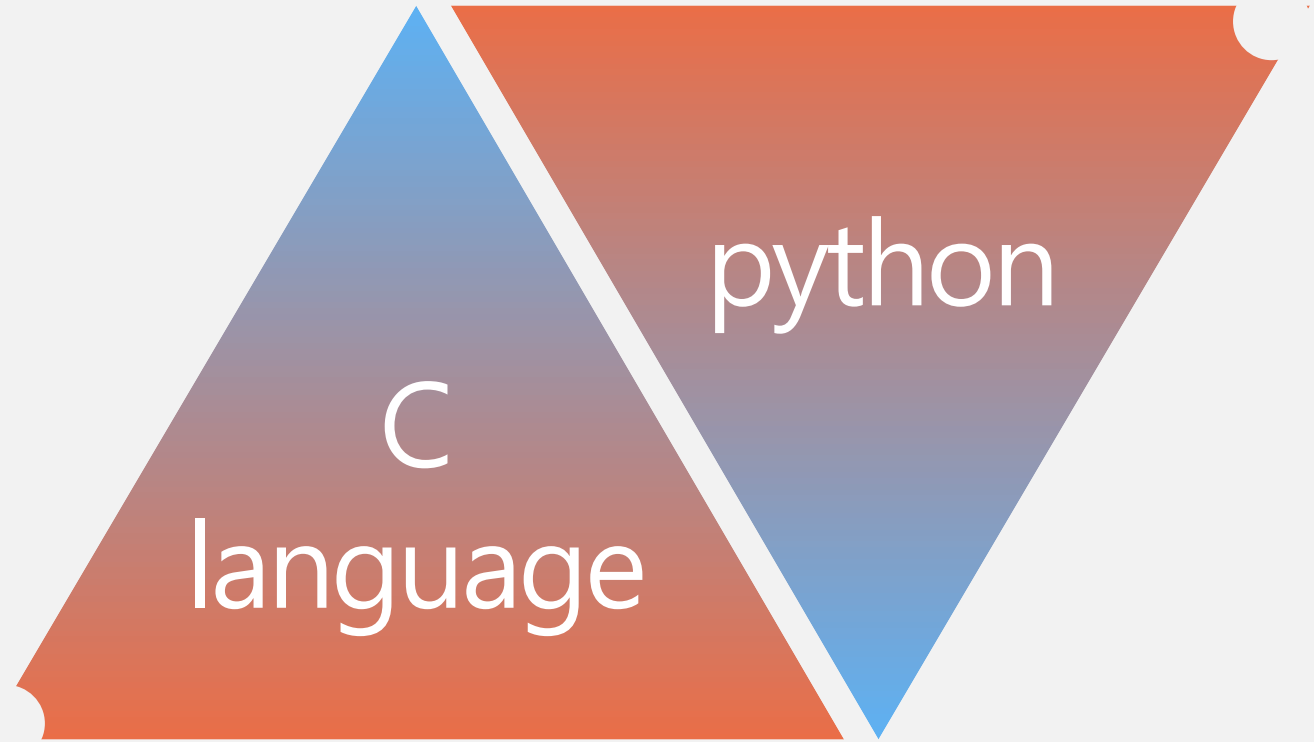
주석을 나타내는 기호는 중복될 수 없다.

```
/* 주석의 시작, 여러 행에 걸쳐서  
    /* 단일 행 주석 처리 */  
*/
```

단, 단일 행 주석은 중복 가능하다.

```
/* 주석의 시작, 여러 행에 걸쳐서  
    // 단일 행 주석 처리  
*/
```

4. printf와 scanf



printf scanf

C언어에서의 자료형, 변수 그리고 연산자에 대해 배워보고,
printf 함수와 scanf 함수를 사용해 출력과 입력을 해보자.

자료형, 변수,
연산자

printf
함수

scanf
함수

자료형(data type)

여러 종류의 데이터를 식별하는 분류

특정 변수의 키워드

```
int val;
```

기본 자료형(primitive data type)

기본적으로 제공이 되는 자료형

그 외 자료형

포인터, 배열

사용자 정의 자료형

표준화된 사용자 정의 자료형

기본 자료형 종류와 데이터의 표현 범위

자료형(data type)		할당되는 메모리 크기	표현 가능한 데이터의 범위
정수형	char	1 바이트	-128 ~ +127
	short	2 바이트	-32768 ~ +32767
	int	4 바이트	-2147483648 ~ +2147483647
	long	4 바이트	-2147483648 ~ +2147483647
실수형	float	4 바이트	$3.4 \times 10^{-37} \sim 3.4 \times 10^{+38}$
	double	8 바이트	$1.7 \times 10^{-307} \sim 1.7 \times 10^{+308}$
	long double	8 바이트 혹은 그 이상	차이를 많이 보임

다양한 자료형이 제공되는 이유

데이터의 표현 방식이 다르기 때문

- 정수형 데이터를 표현하는 방식
- 실수형 데이터를 표현하는 방식

메모리 공간을 적절히 사용하기 위해서

- 데이터의 표현 범위를 고려해서 자료형 선택
 - 작은 메모리 공간에 큰 데이터를 저장하는 경우
데이터 손실이 발생할 수 있음
-

연산자란 무엇인가?

연산을 요구할 때 사용되는 기호

ex : +, -, *, /

```
int main(void)
{
    3+4;    // 덧셈 결과를 저장할 필요가 있다.
    return 0;
}
```

변수란 무엇인가?

데이터를 저장할 수 있는 메모리 공간에 붙여진 이름

다양한 형태(자료형)의 변수

정수형 : char, int, long

실수형 : float, double

변수의 선언 및 대입

대입 연산자(=): 값을 대입하기 위한 용도의 연산자

```
int main(void)
{
    int val;    // int형 변수 val의 선언
    val = 20;   // 변수 val에 20을 저장
    . . . . .
```

변수를 이용한 예제

```
#include <stdio.h>

int main(void)
{
    int a, b;                // 쓰레기 값으로 초기화
    int c=30, d=40;

    a=10;
    b=20;

    printf("%d %d\n", a, b);
    printf("%d %d\n", c, d);
    return 0;
}
```

변수 선언 시 주의 사항 1

변수를 함수 내에 선언할 경우, 등장 위치!

```
#include <stdio.h>

int main(void)
{
    int a;
    int b;

    a=10;
    b=20;

    printf("%d %d \n", a, b);
    return 0;
}
```

변수 선언 시 주의 사항 2

첫째 : 변수의 이름은 알파벳, 숫자 언더바(_)로 구성

둘째 : 대 소문자 구분

셋째 : 변수의 이름은 숫자로 시작 불가, 키워드 사용 불가

넷째 : 공백이 포함될 수 없음

적절치 않은 변수의 이름	적절치 않은 이유
int 7th_val	변수의 이름이 숫자로 시작
int live_inthe#	#과 같은 특수 문자는 올 수 없다.
int kor year	변수 이름에 공백이 삽입될 수 없다.

완성된 덧셈 프로그램

```
/* simpleadd2.c */
#include <stdio.h>

int main(void)
{
    int result;                //변수 선언
    result=3+4;                //덧셈 결과 저장

    printf("덧셈 결과 : %d \n", result);
    printf("%d 더하기 %d는 %d 입니다. \n", 3, 4, result);
    printf("변수 result에 저장된 값 : %d \n", result);

    return 0;
}
```

변수와는 다른 상수!

상수도 메모리 공간을 할당 받는다.
하지만 데이터의 변경이 불가능하다.

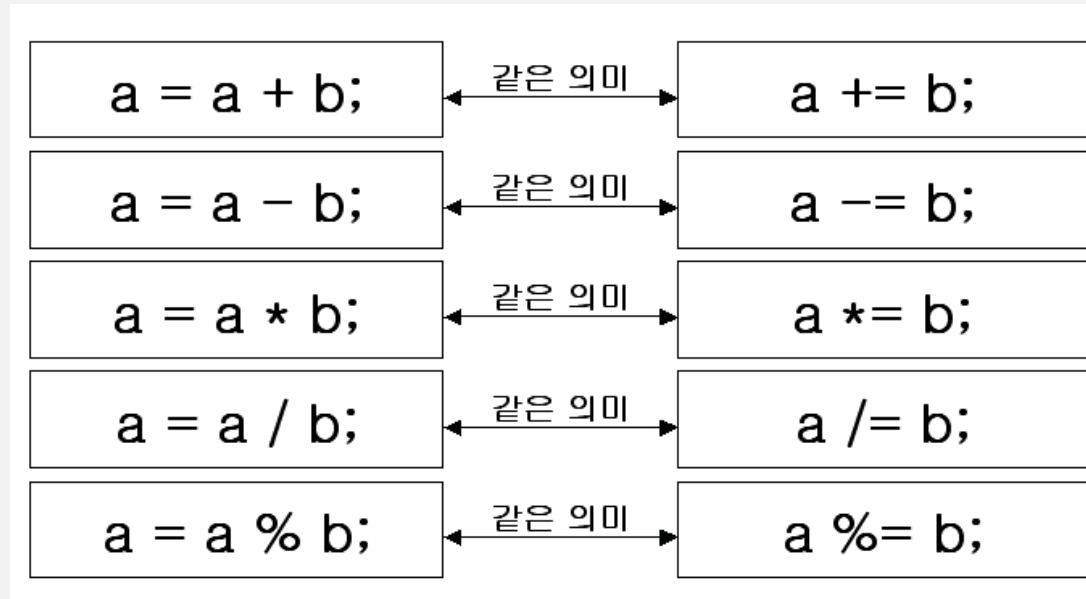


대입 연산자와 산술 연산자

연산자	연산의 예	의미	결합성
=	a=20	대입	←
+	a=4+3	덧셈	→
-	a=4-3	뺄셈	→
*	a=4*3	곱셈	→
/	a=4/3	나눗셈	→
%	a=4%3	나머지	→

기타 대입 연산자

대입 연산자와 산술 연산자가 합해져서
다양한 형태의 대입 연산자 정의



부호 연산으로서 +, - 연산자

단항 연산자로서 +, -

Ex) `int a=+3; int b=-7;`

증가 감소 연산자

연산자	연산의 예	의미	결합성
<code>++a</code>	<code>printf("%d", ++a)</code>	선 증가, 후 연산	←
<code>a++</code>	<code>printf("%d", a++)</code>	선 연산, 후 증가	←
<code>--b</code>	<code>printf("%d", --a)</code>	선 감소, 후 연산	←
<code>b--</code>	<code>printf("%d", a--)</code>	선 연산, 후 감소	←

관계 연산자(비교 연산자)

두 피연산자의 관계(크다, 작다 혹은 같다)를 따지는 연산자
true(논리적 참, 1), false(논리적 거짓, 0) 반환

연산자	연산의 예	의미	결합성
<	a<b	a가 b보다 작은가	→
>	a>b	a가 b보다 큰가	→
==	a==b	a와 b가 같은가	→
!=	a!=b	a와 b가 같지 않은가	→
<=	a<=b	a가 b보다 작거나 같은가	→
>=	a>=b	a가 b보다 크거나 같은가	→

논리 연산자

and, or, not을 표현하는 연산자

true(1), false(0) 반환

연산자	연산의 예	의미	결합성
&&	a&&b	true면 true 리턴	→
	a b	하나라도 true면 true 리턴	→
!	!a	true면 false를, false면 true 리턴	→

연산자의 우선 순위

연산 순서를 결정짓는 순위

연산자의 결합성

우선 순위가 같은 연산자들의 연산 방향

$$3+4*5/2-10$$

printf scanf

C언어에서의 자료형, 변수 그리고 연산자에 대해 배워보고,
printf 함수와 scanf 함수를 사용해 출력과 입력을 해보자.

자료형, 변수,
연산자

printf
함수

scanf
함수

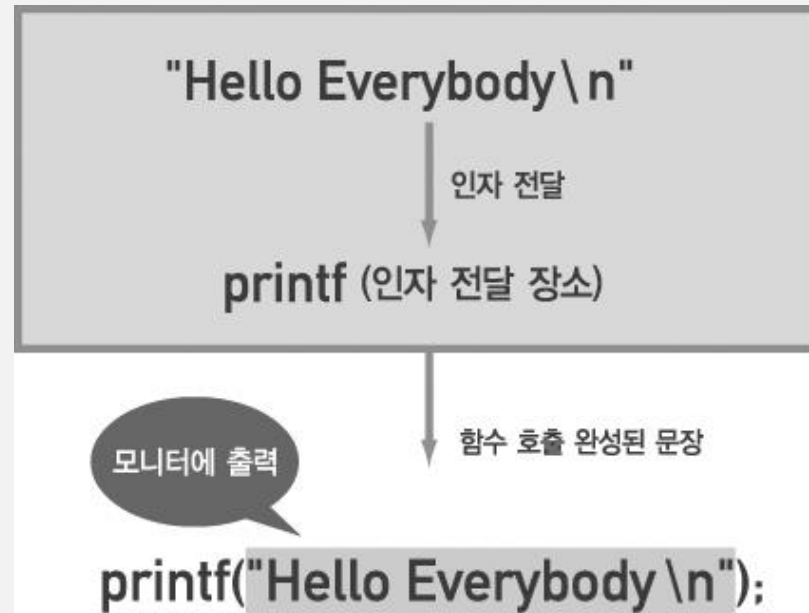
printf 함수 사용의 예 1

```
/* printf1.c */
#include <stdio.h>

int main(void)
{
    printf("Hello Everybody \n");
    printf("%d \n", 1234);
    printf("%d %d \n", 10, 20);
    return 0;
}
```

printf 함수 호출의 이해 1

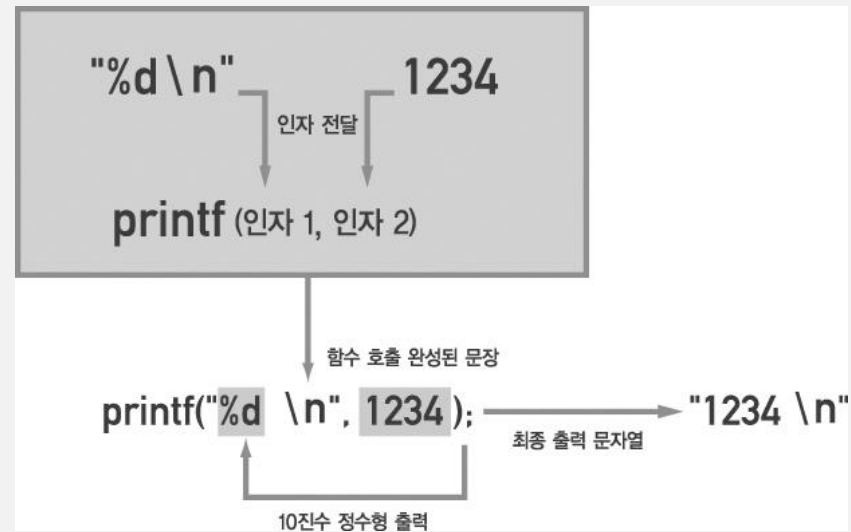
```
printf("Hello Everybody \n");
```



printf 함수 호출의 이해 2 : 서식 문자

서식 문자(Conversion specifier)란 출력 대상의 출력 형태를 지정하기 위한 문자

```
printf("%d \n", 1234);
```



printf 함수 호출의 이해 3

```
printf("%d %d \n", 10, 20);
```



printf 함수 사용의 예 2

```
/* printf2.c */
#include <stdio.h>

int main(void)
{
    printf("My age : %d  \n", 20);
    printf("%d is my point \n", 100);
    printf("Good \nmorning \neverybody\n");

    return 0;
}
```


printf 함수 호출의 이해 4

printf("My age : %d \n", 20) → "My age : 20 \n"

↑ 최종 출력 문자열

printf("%d is my point \n", 100) → "100 is my point \n"

↑ 최종 출력 문자열

printf("Good \n morning \n everybody \n"); → Good
morning
everybody

출력 ↑ 출력 ↑ 출력 ↑ 최종 출력 형태

줄 바꾸고 줄 바꾸고 줄 바꾸고

printf scanf

C언어에서의 자료형, 변수 그리고 연산자에 대해 배워보고,
printf 함수와 scanf함수를 사용해 출력과 입력을 해보자.



scanf 함수를 이용한 정수의 입력

```
int main(void)
{
    int val;
    scanf("%d", &val);
    . . . . .
```

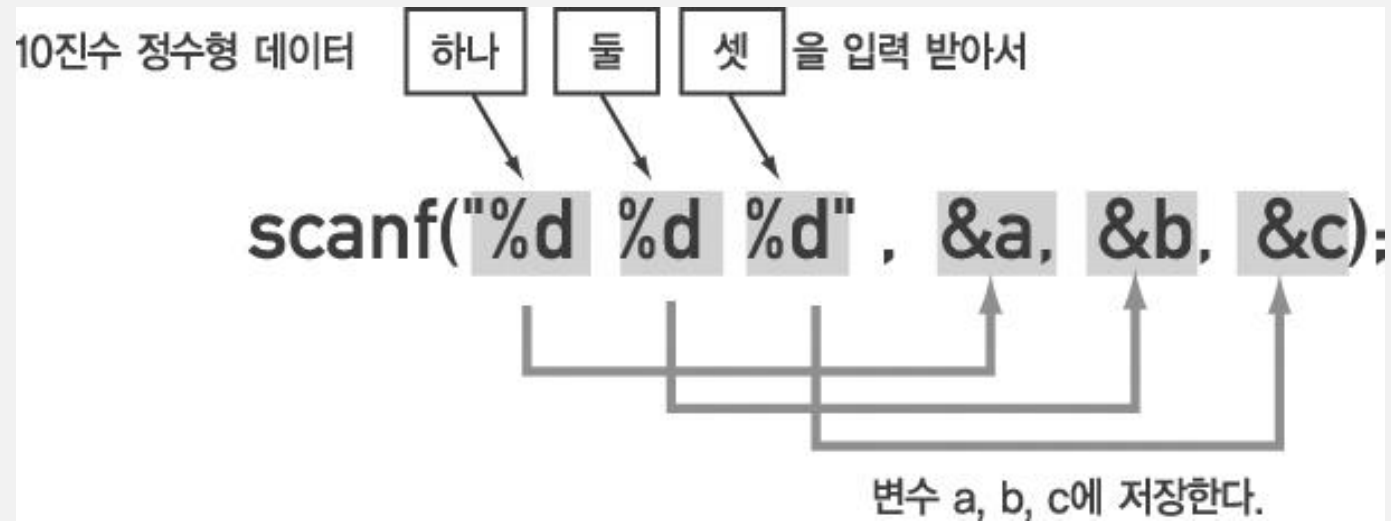
scanf("%d" , &val);

↓
10진수 정수형으로 입력 받아서

↑
변수 val에 저장하라.

scanf 함수를 이용한 입력 형태의 지정

입력 형태의 지정이 가능



복습
