전산 SMP 1주차

2015.09.15

CSE 12 김범수
bskim45@gmail.com

시작하기 전에

- 진도 체크
- 회식 날짜

- 어싸인은 스스로, 질문은 환영
- 수업 시간에 디자인 팁을 줍니다
- 제발 손으로 디자인 먼저 하세요

- Course homepage: http://smp.bsk.im

왜 배우나요?

대세임 ㅋ

“프로그래밍”과 **“문제해결”**

강좌의 목적은 좋은 프로그래머가 되는 것이 아니고

컴퓨터의 **작동 원리** 전반에 대해 이해하고,

컴퓨터의 힘을 빌려 **문제를 해결하는 능력**을 기르는 것

컴퓨터랑 친해지기

What is a Computer?

Computer is one that computes; *specifically* : a programmable usually electronic device that can **store, retrieve, and process data**

– Merriam–Webster English Dictionary

Image from Forbes, *Dell Gives Sneak Peek Of New XPS 15 Laptop On Windows 10*

High-level vs Low-level

- 인간과 컴퓨터 중 누구에게 더 가까운가?

고급(High Level)

저급(Low Level)

Low-level Language

- 기계어, 어셈블리어 등
  - 아키텍처마다 다름
  - 성능이 강력하지만, 배우기 어려움

```
int arith(int x, int y, int z)
{
  int t1 = x+y;
  int t2 = z+t1;
  int t3 = x+4;
  int t4 = y * 48;
  int t5 = t3 + t4;
  int rval = t2 * t5;
  return rval;
}
```

```
arith:
    pushl   %ebp                    } Set
    movl    %esp, %ebp              } Up

    movl    8(%ebp), %ecx           } Body
    movl    12(%ebp), %edx          } Body
    leal    (%edx,%edx,2), %eax     } Body
    sall    $4, %eax                } Body
    leal    4(%ecx,%eax), %eax      } Body
    addl    %ecx, %edx              } Body
    addl    16(%ebp), %edx          } Body
    imull   %edx, %eax              } Body

    popl    %ebp                    } Finish
    ret                             } Finish
```

High-level Language

- C, C++, Java, Python 등
  - 사람이 알기 쉽도록 쓰여진 프로그래밍 언어
  - 컴파일러 필요
  - 가독성이 높고 다루기 쉽다

```
1  public class CreateObjectDemo
2  {
3      public static void main(String[] args)
4      {
5          // Declare and create a point object and two rectangle objects.
6          Point originOne = new Point(23, 94);
7          Rectangle rectOne = new Rectangle(originOne, 100, 200);
8          Rectangle rectTwo = new Rectangle(50, 100);
9
10         // display rectOne's width, height, and area
11         System.out.println("Width of rectOne: " + rectOne.getWidth());
12         System.out.println("Height of rectOne: " + rectOne.getHeight());
13         System.out.println("Area of rectOne: " + rectOne.getArea());
14
15 }
```

컴파일러

- High-level to Low-level

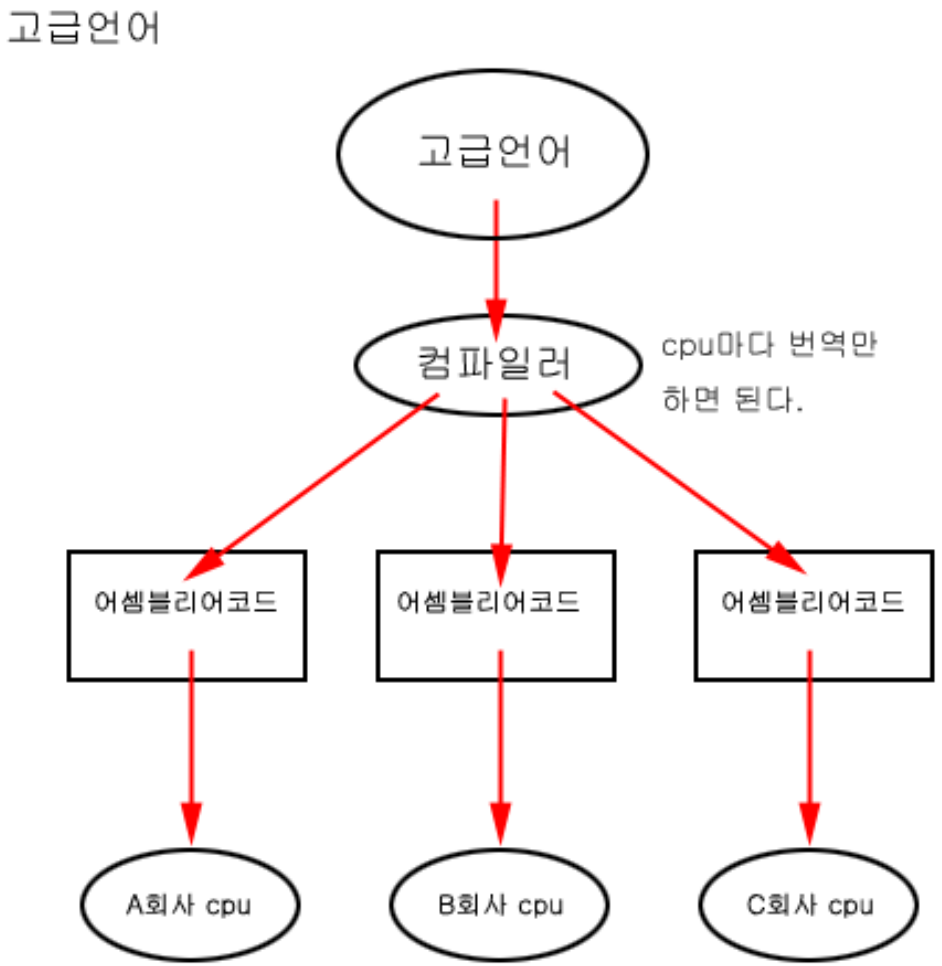


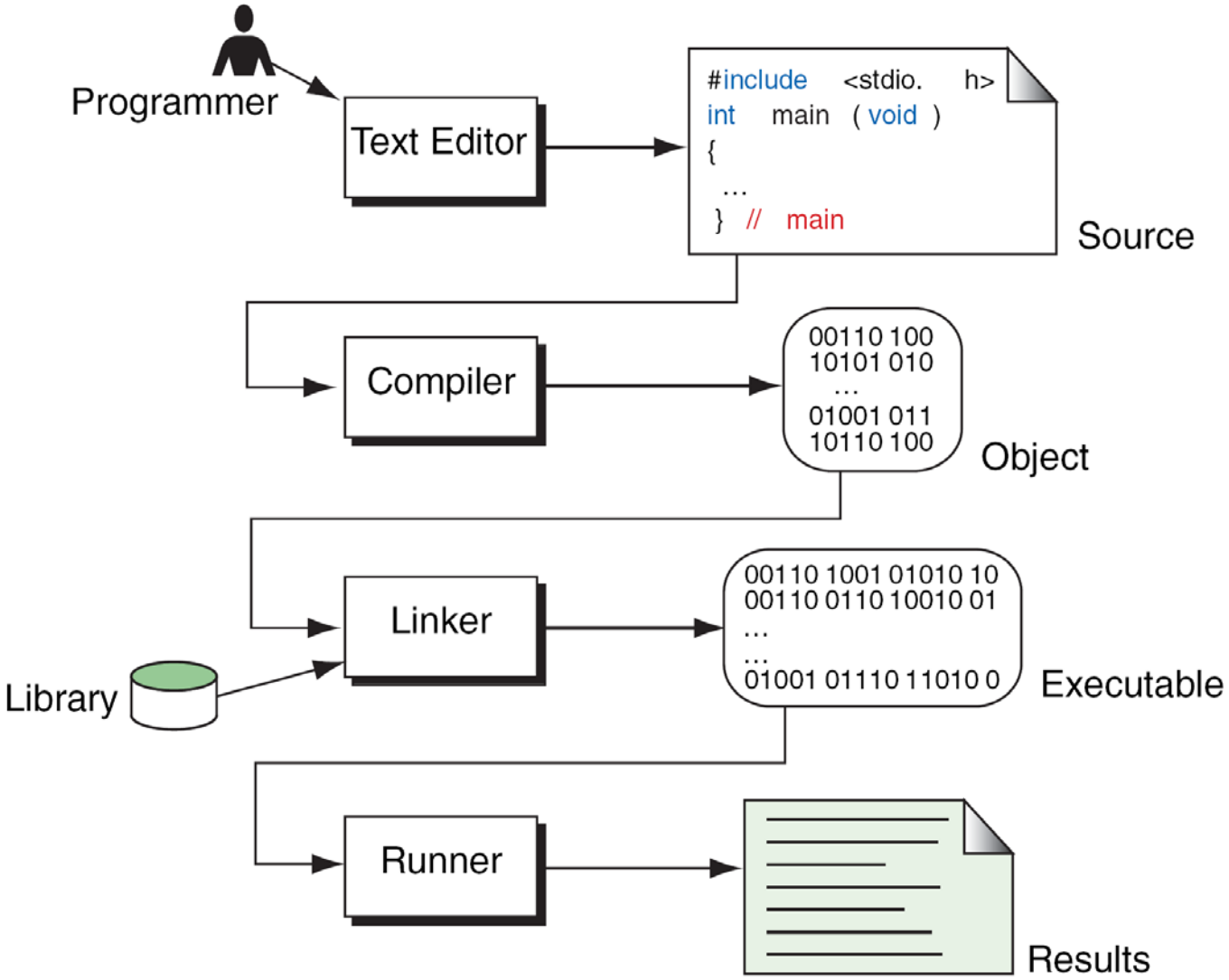
Image retrieved from http://sessionk.tistory.com/123

C 언어

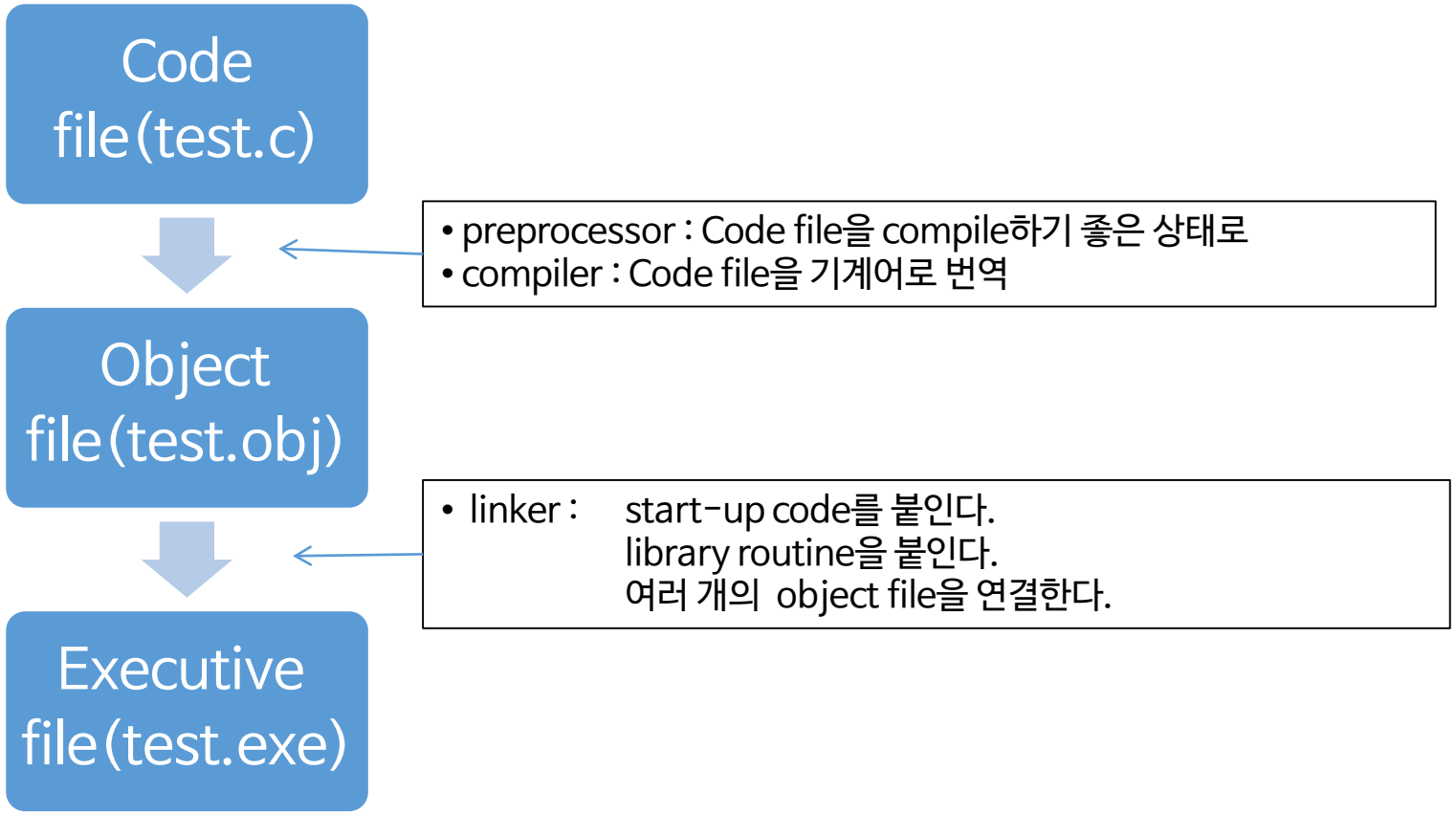
- 가장 대표적인 프로그래밍 언어
- 1972년 벨 연구소 Kenneth Thompson, **Dennis Ritchie**
- UNIX 운영 체제에서 사용하기 위해 개발한 프로그래밍 언어
- 거의 모든 컴퓨터 시스템에서 사용 가능(표준 C 라이브러리)
  - '어셈블리 언어'는 시스템에 따라 다르다 → 이식성이 낮음
- 절차지향적(imperative) 언어
  - Imperative programming is about modifying mutable variables, using assignments and control structures such as if-then-else, loops, break,continue, return.
  - The most common informal way to understand imperative programs is as instruction sequences for a Von Neumann computer.

- Martin Odersky, "Functional Programming Principles in Scala"

C 언어에서 프로그램(Executable)으로



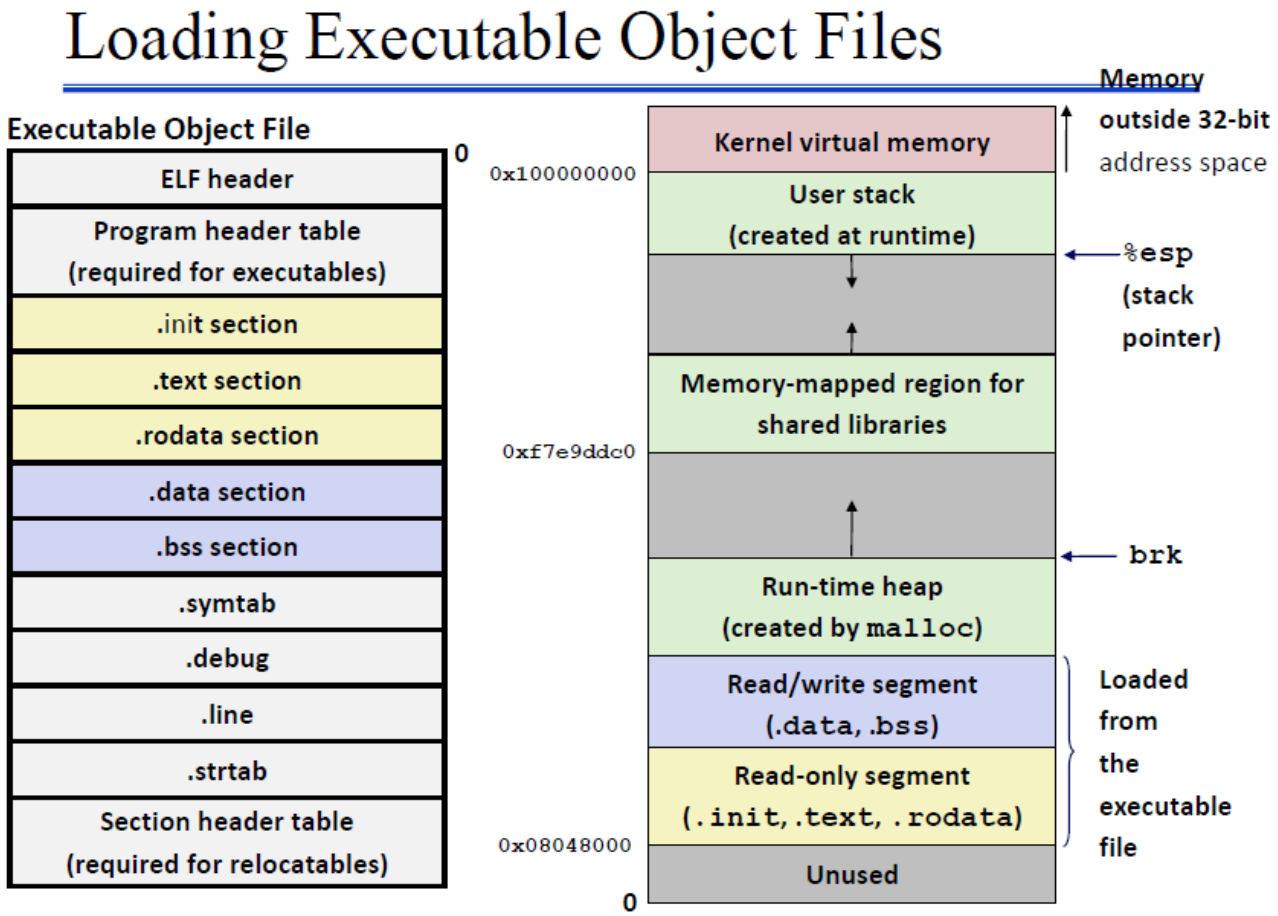
C 언어에서 프로그램으로

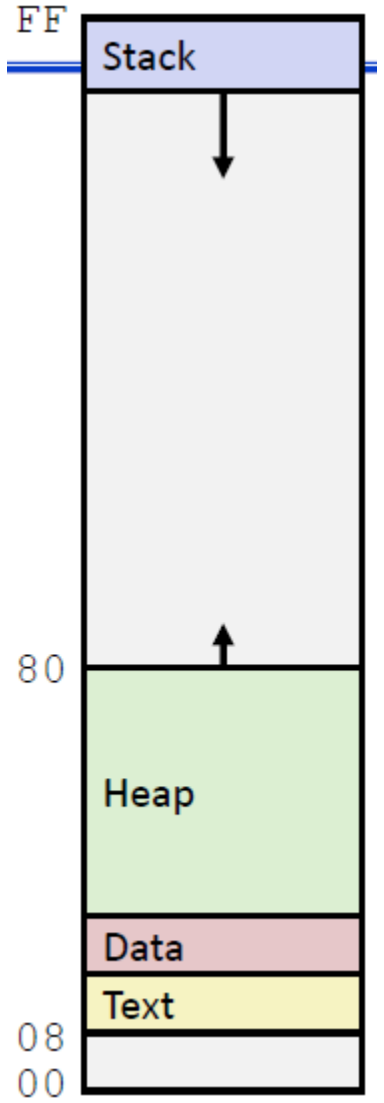


• start-up code : os와 원시 file의 다리 역할: main을 부르고 전달인자를 main에게 넘겨준다.
• library routine : 함수 library(printf, scanf)의 기계어 code 부분들이 들어오는 자리

프로그램을 실행하면…

- 코드가 메모리에 로드된다





Stored Program (von Neumann) Architecture

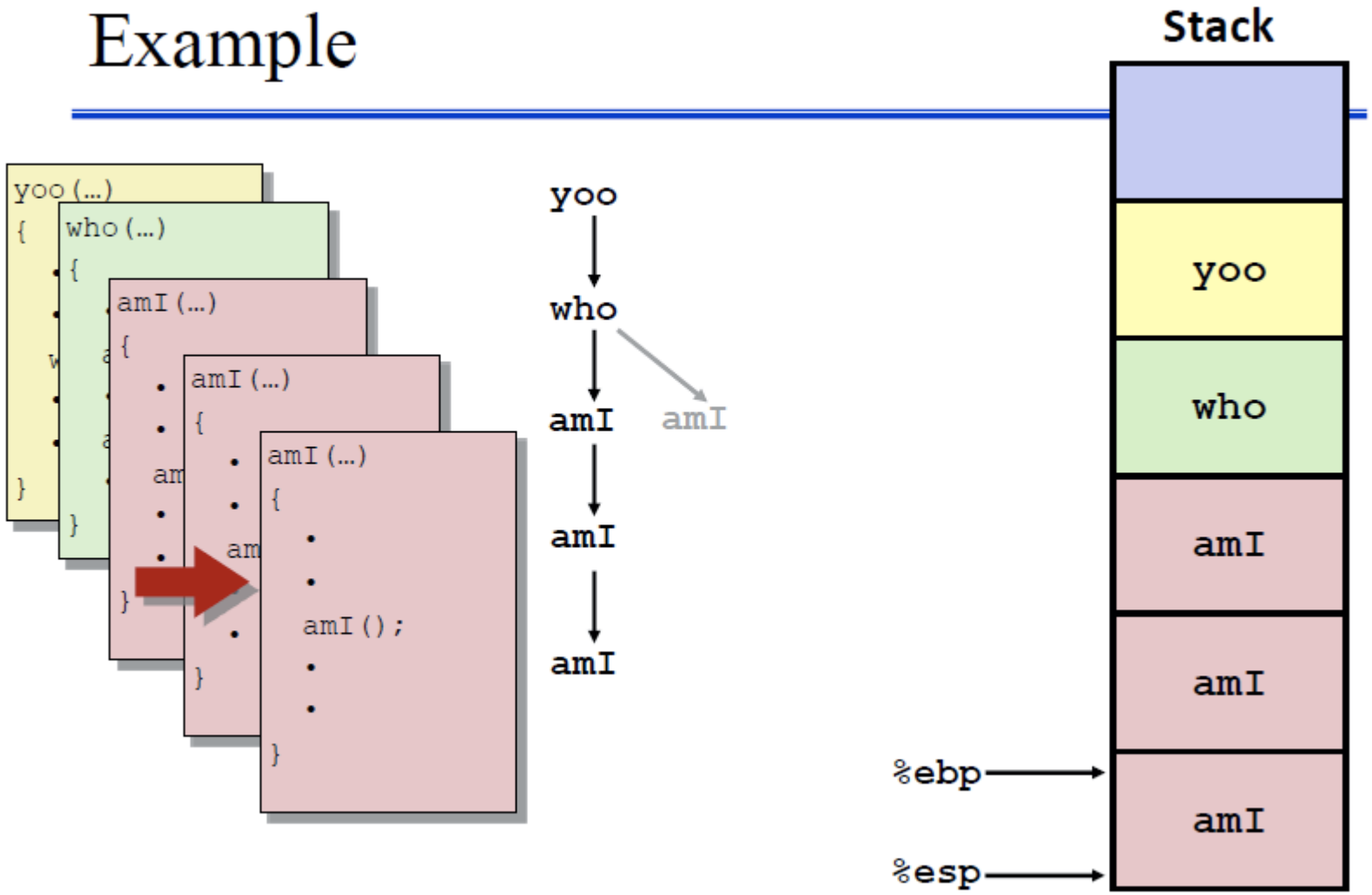
Stored-program architecture

- Instructions in a linear memory array
  - Instructions (as well as data) are in memory
- Instructions can be modified just like data
  - Both forms consist of 0s and 1s

Sequential instruction processing

- Program counter identifies the current instruction
- Instruction is fetched from memory and executed
- Program counter is advanced (according to instruction)
- Repeat

순서대로 메모리에


Example
yoo (...)
{
who (...)
{
amI (...)
{
amI (...)
{
amI (...)
{
amI();
}
yoo
who
amI
amI
amI
amI
Stack
yoo
who
amI
amI
amI
%ebp
%esp

메모리

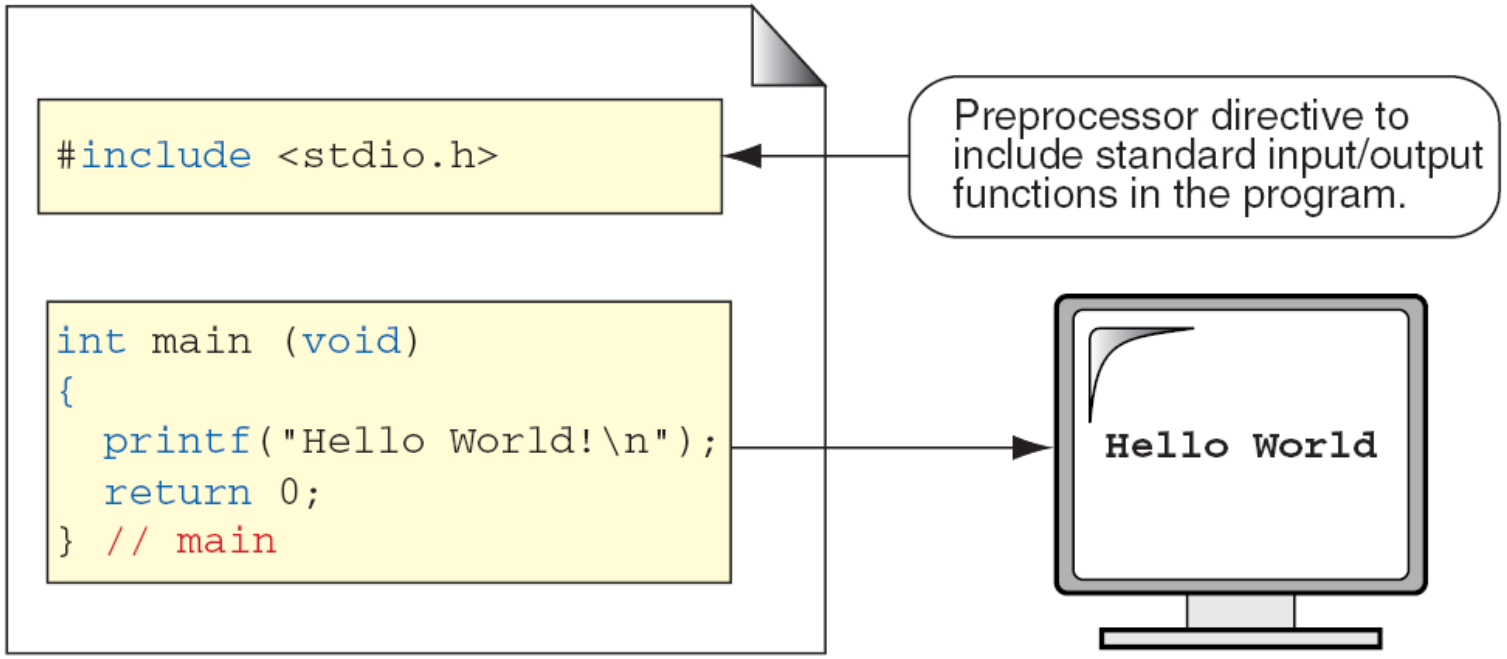
- 1 byte = 8 bits = 0101 0101
- 10^3 = Kilo / 10^6 = Mega / 10^9 = Tera
- 2^10 = 1024 = Kilo / 2^20 = Mega / 2^30 = Tera

- **32bit** OS → 2^32 = 4GB = 램은 4GB까지
- 1 word = 4 bytes = 32 bits

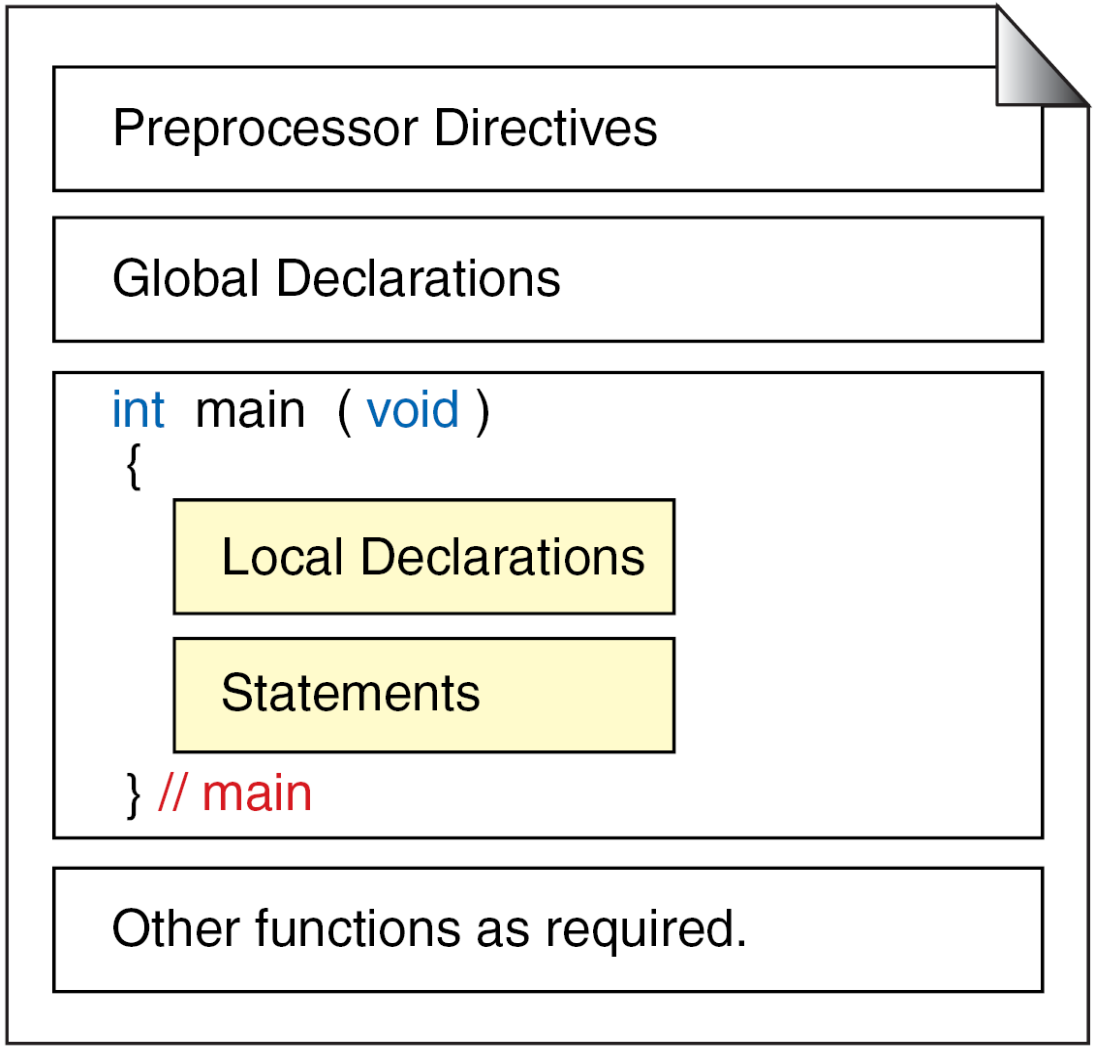
- **64bit** OS → 2^64 = 거의 무한대
- 1 word = 8 bytes = 64 bits

C 언어 개론

Hello World!


#include <stdio.h>
int main (void)
{
printf("Hello World!\n");
return 0;
} // main
Preprocessor directive to include standard input/output functions in the program.
Hello World

Scopes


Preprocessor Directives
Global Declarations
int main (void)
{
Local Declarations
Statements
} // main
Other functions as required.

주석 (Comments)

- 코드의 가독성
- 프로그램 내에 삽입된 메모
- 실행결과에 영향을 미치지 않는다.
- Compile 대상에서 제외

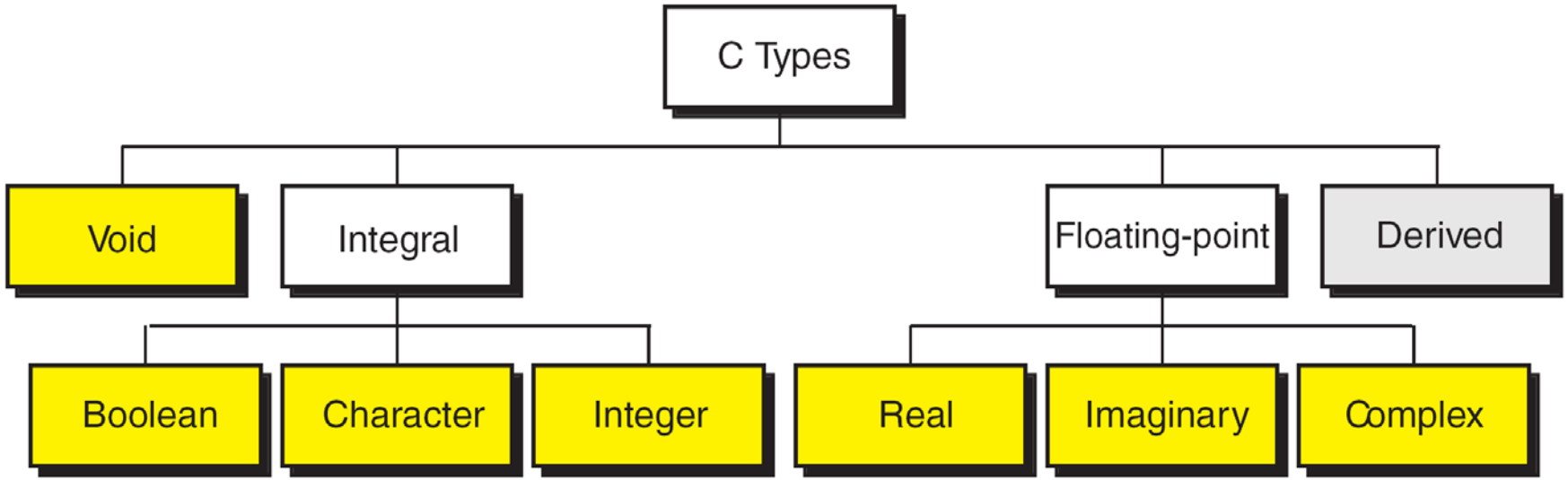
// 한 줄 주석

/*
 * 여러
 * 줄
 * 주석
 */

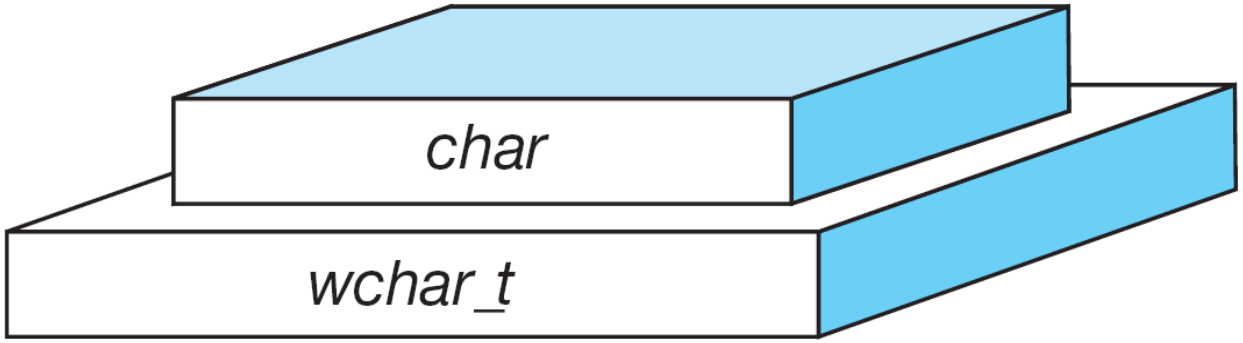
```
 1  /* The greeting program. This program demonstrates
 2     some of the components of a simple C program.
 3        Written by:  your name here
 4        Date:        date program written
 5  */
 6  #include <stdio.h>
 7
 8  int main (void)
 9  {
10  // Local Declarations
11
12  // Statements
13
14     printf("Hello World!\n");
15
16     return 0;
17  } // main
```

자료형 (Types)

자료형


C Types
Void
Integral
Floating-point
Derived
Boolean
Character
Integer
Real
Imaginary
Complex

Character Types

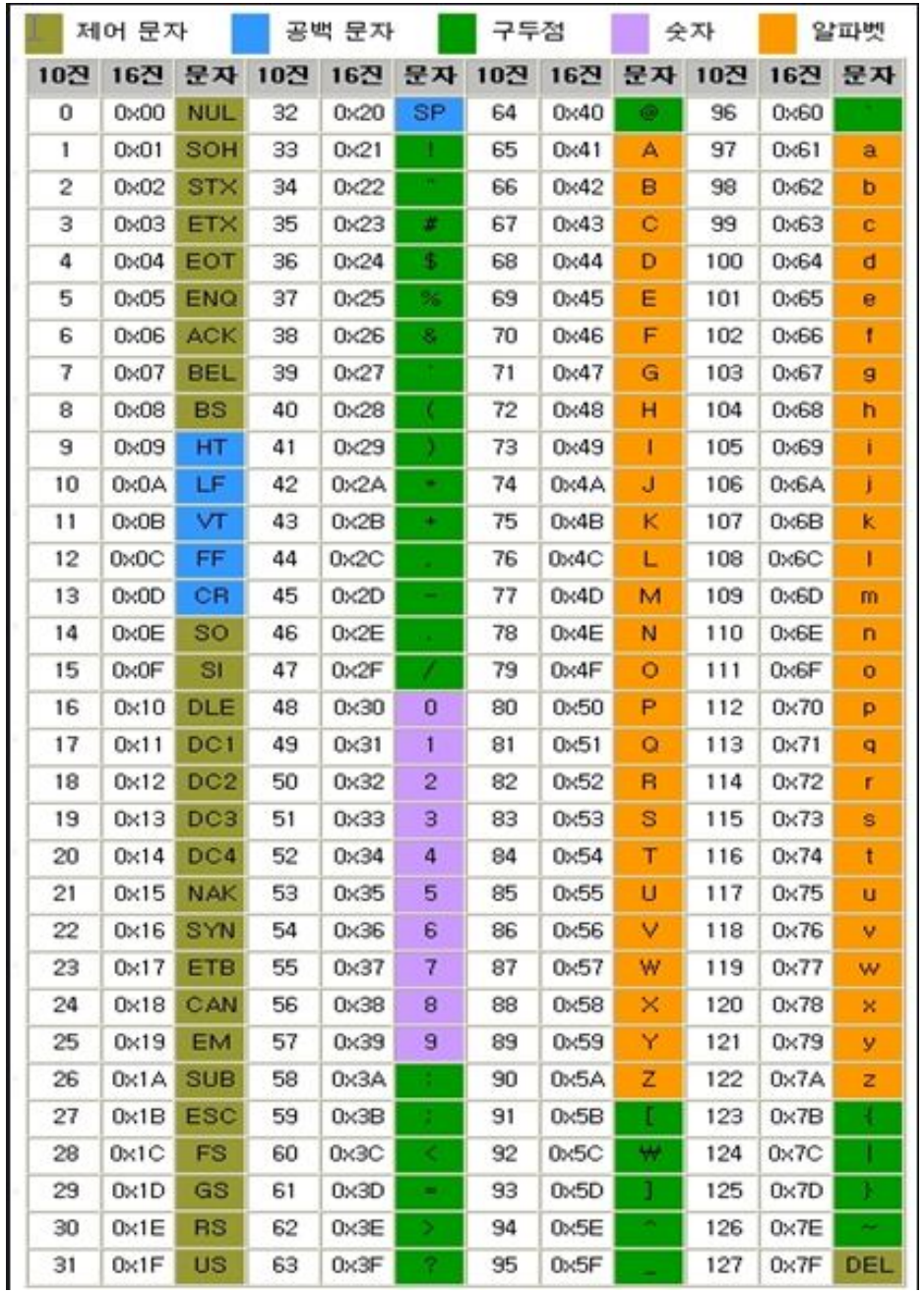


- 1byte = 2^8 = 256개의 문자
- 각각의 문자가 아스키 코드에 대응!

아스키(ASCII) 코드

- C의 char는 ASCII코드를 이용해 나타냅니다.
- 문자와 코드가 1대1 대응.

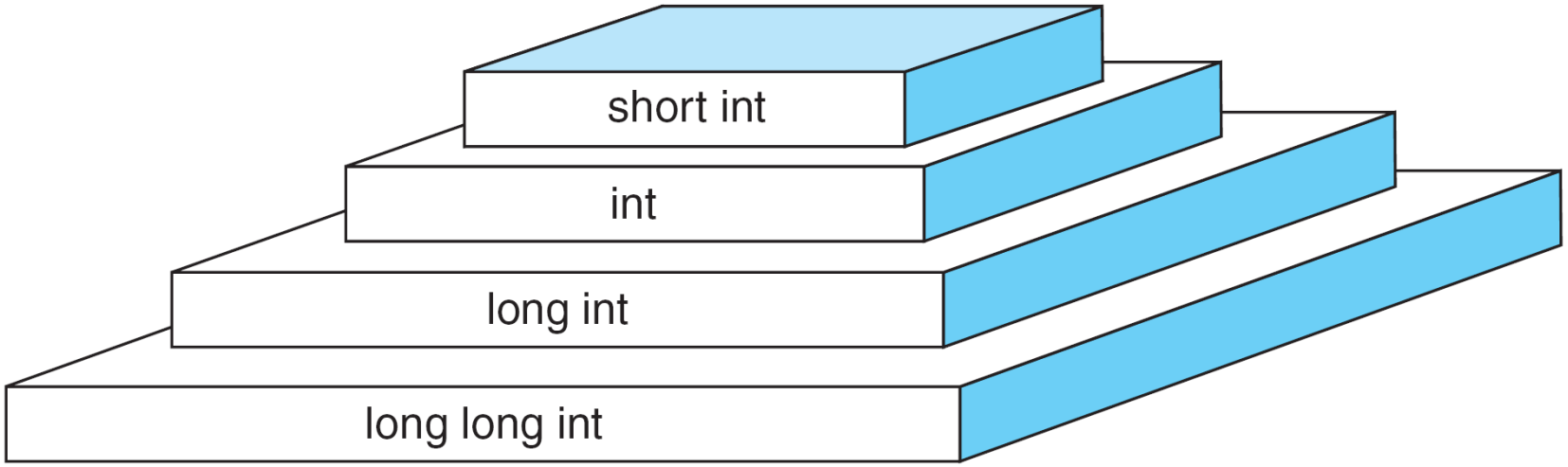
- 소문자 a(97) to z(112)
- 대문자 A(65) to Z(90)
- 숫자 0(48) to 9(57)
- 특수문자 &(38), *(42), +(43)



제어 문자 | 공백 문자 | 구두점 | 숫자 | 알파벳

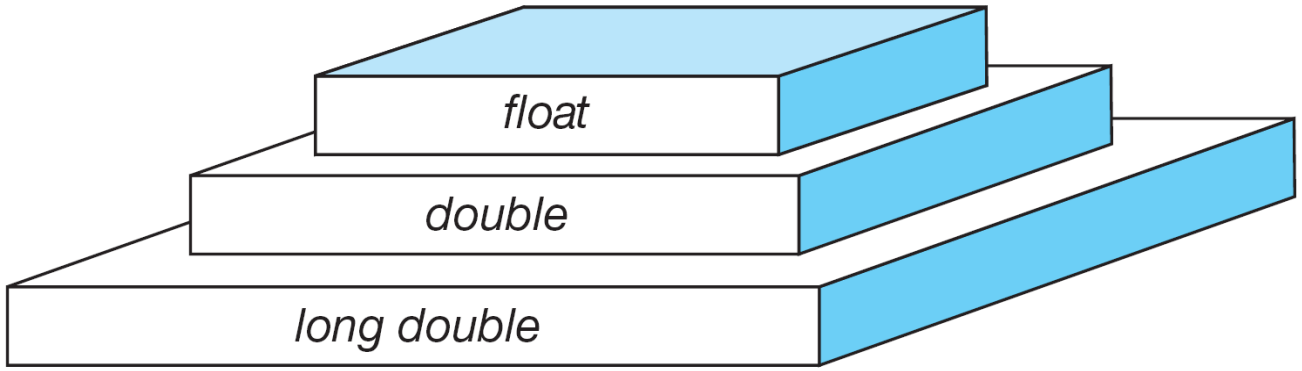
10진	16진	문자	10진	16진	문자	10진	16진	문자	10진	16진	문자
0	0x00	NUL	32	0x20	SP	64	0x40	@	96	0x60	`
1	0x01	SOH	33	0x21	!	65	0x41	A	97	0x61	a
2	0x02	STX	34	0x22	"	66	0x42	B	98	0x62	b
3	0x03	ETX	35	0x23	#	67	0x43	C	99	0x63	c
4	0x04	EOT	36	0x24	$	68	0x44	D	100	0x64	d
5	0x05	ENQ	37	0x25	%	69	0x45	E	101	0x65	e
6	0x06	ACK	38	0x26	&	70	0x46	F	102	0x66	f
7	0x07	BEL	39	0x27	'	71	0x47	G	103	0x67	g
8	0x08	BS	40	0x28	(	72	0x48	H	104	0x68	h
9	0x09	HT	41	0x29	)	73	0x49	I	105	0x69	i
10	0x0A	LF	42	0x2A	*	74	0x4A	J	106	0x6A	j
11	0x0B	VT	43	0x2B	+	75	0x4B	K	107	0x6B	k
12	0x0C	FF	44	0x2C	,	76	0x4C	L	108	0x6C	l
13	0x0D	CR	45	0x2D	-	77	0x4D	M	109	0x6D	m
14	0x0E	SO	46	0x2E	.	78	0x4E	N	110	0x6E	n
15	0x0F	SI	47	0x2F	/	79	0x4F	O	111	0x6F	o
16	0x10	DLE	48	0x30	0	80	0x50	P	112	0x70	p
17	0x11	DC1	49	0x31	1	81	0x51	Q	113	0x71	q
18	0x12	DC2	50	0x32	2	82	0x52	R	114	0x72	r
19	0x13	DC3	51	0x33	3	83	0x53	S	115	0x73	s
20	0x14	DC4	52	0x34	4	84	0x54	T	116	0x74	t
21	0x15	NAK	53	0x35	5	85	0x55	U	117	0x75	u
22	0x16	SYN	54	0x36	6	86	0x56	V	118	0x76	v
23	0x17	ETB	55	0x37	7	87	0x57	W	119	0x77	w
24	0x18	CAN	56	0x38	8	88	0x58	X	120	0x78	x
25	0x19	EM	57	0x39	9	89	0x59	Y	121	0x79	y
26	0x1A	SUB	58	0x3A	:	90	0x5A	Z	122	0x7A	z
27	0x1B	ESC	59	0x3B	;	91	0x5B	[	123	0x7B	{
28	0x1C	FS	60	0x3C	<	92	0x5C	₩	124	0x7C	|
29	0x1D	GS	61	0x3D	=	93	0x5D	]	125	0x7D	}
30	0x1E	RS	62	0x3E	>	94	0x5E	^	126	0x7E	~
31	0x1F	US	63	0x3F	?	95	0x5F	_	127	0x7F	DEL

Integer Types



Type	Byte Size	Minimum Value	Maximum Value
short int	2	–32,768	32,767
int	4	–2,147,483,648	2,147,483,647
long int	4	–2,147,483,648	2,147,483,647
long long int	8	–9,223,372,036,854,775,807	9,223,372,036,854,775,806

Floating Point Types (실수형)



Type Summary

Category	Type	C Implementation
Void	Void	*void*
Integral	Boolean	*bool*
	Character	*char, wchar_t*
	Integer	*short int, int, long int, long long int*
Floating-Point	Real	*float, double, long double*
	Imaginary	*float imaginary, double imaginary, long double imaginary*
	Complex	*float complex, double complex, long double complex*

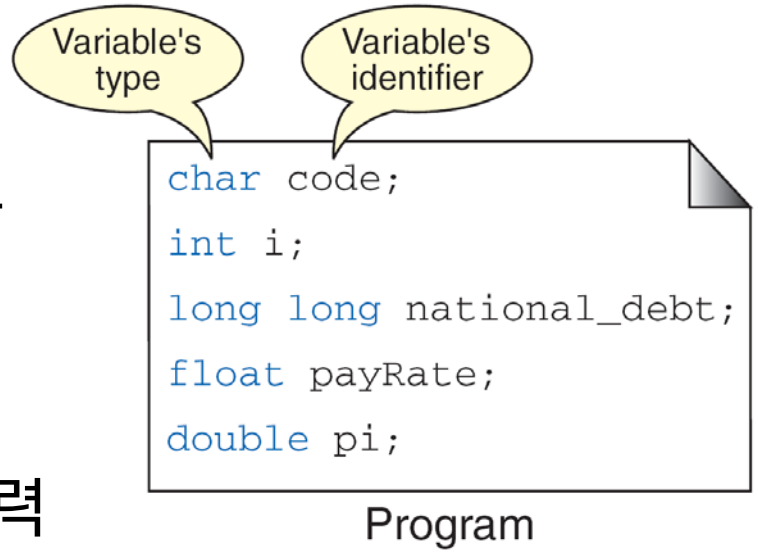
기본 자료형(Primitives)의 종류와 데이터의 표현 범위

자료형		크기	값의 표현범위
정수형	char	1byte	-128 ~ 127
	unsigned char	1byte	0~255
	short	2byte	-32768~32767
	unsigned short	2byte	0~65535
	int	4byte	-2,147,483,648~2,147,483,647
	unsigned int	4byte	0~4,294,967,294
	long	4byte	-2,147,483,648~2,147,483,647
	unsigned long	4byte	0~4,294,967,294
	long long	8byte	-9,223,372,036,854,775,808~9,223,372,036,854,775,808
	unsigned long long	8byte	0~위 숫자의 2배
실수형	float	4byte	$\pm 3.4 \times 10^{-37}$ ~ $\pm 3.4 \times 10^{+38}$
	double	8byte	$\pm 1.7 \times 10^{-307}$ ~ $\pm 1.7 \times 10^{+308}$
	long double	8byte이상	double이상의 표현범위

변수(Variable)란?

- 값을 저장할 수 있는 메모리 공간에 붙여진 이름
- 변수를 선언하면 메모리 공간이 할당되고 그 공간에 이름이 붙는다.
- 일반 변수와 포인터 변수
- const 키워드: 변수를 상수화시킨다.

```
int num; // int : 정수의 저장을 위한 메모리 공간의 할당
         num : 할당된 메모리 공간의 이름(declare)
num=20; //변수 num에 접근하여 20을 저장(assign)
printf("%d", num); // num의 값을 10진 정수로 출력
```



식별자(Identifier) 이름규칙

- 변수의 이름은 알파벳, 숫자, 언더바(_)로 구성
- 대소문자 구분
- 숫자로 시작할 수 없고, 키워드를 변수의 이름으로 사용할 수 없다.
- 이름 사이에 공백 넣을 수 없다.

잘된 예	잘못된 예
`int  ab3;`	`int 4ab;`
`char num_char;`	`char num-char;`
`double AB3;`	`doouble ?q;`

C언어의 표준 키워드

아래의 단어들은 기능적 의미가 정해진, C언어의 문법을 구성하는 단어들(키워드)이므로 변수이름으로 사용할 수 없다.

auto	double	int	struct	break
else	long	switch	case	enum
register	typedef	char	extern	return
union	const	float	short	unsigned
continue	for	signed	void	default
goto	sizeof	volatile	do	if
static	while			

상수(Constant)란?

- 숫자 상수
- 문자(Character) 상수
- 문자열(String) 상수
- 매크로(Macro) 상수

#define

- 전처리(Preprocessor Directive) 부분에 선언
- **컴파일** 시 자동으로 대체(substitution)되어 들어간다.

```
#define PI 3.141592     // 세미콜론 안 붙임
…
float pi = PI;      // float pi = 3.141592;

#define square(x) (x * x)
…
int a = square(2)       // int a = 2*2;
```

3-ways to use constants

```
 5  #include <stdio.h>
 6  #define PI 3.1415926536
 7
 8  int main (void)
 9  {
10  // Local Declarations
11     const double cPi = PI;
12
13  // Statements
14     printf("Defined constant PI: %f\n", PI);
15     printf("Memory constant cPi: %f\n",cPI);
16     printf("Literal constant:    %f\n", 3.1415926536);
17     return 0;
18  }  // main
```

```
Results:
Defined constant PI:  3.141593
Memory constant cPi:  3.141593
Literal constant:     3.141593
```

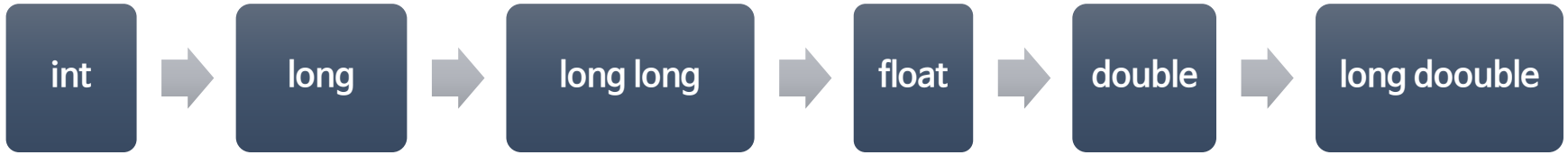
형 변환 (Type Conversion)

- **Implicit Conversion**
  - 컴파일러에 의해 사용자 모르게 자동으로 이루어 지는 형변환
  - Promotion, Demotion : 연산시 우선순위 높은 거에 맞춰서 다른 데이터 변환 (bool 〈 char 〈 integer 〈 real number)
  - 최종적으로 값 대입(assing)시 저장되는 공간의 타입에 따라 결정
  - `int a = 3.14;  // a = ??`

- **Explicit Conversion(casting)**
  - 프로그래머가 인위적으로 타입을 바꿈. 사용시 주의!
  - `double a = (double) 4/3  // a = ??`
  - `double b = (double) 4 / (double) 3  // b = ??`

자동(implicit) 형 변환

```
double a=1.7 + 30; // a=31.7
int b=1.7 + 30; // b=31

int main(void) {
        int a = -1;
        unsigned int b = 100;
        if(a > b)
                printf(“a is bigger”);
        else
                printf(“b is bigger”);
}
```



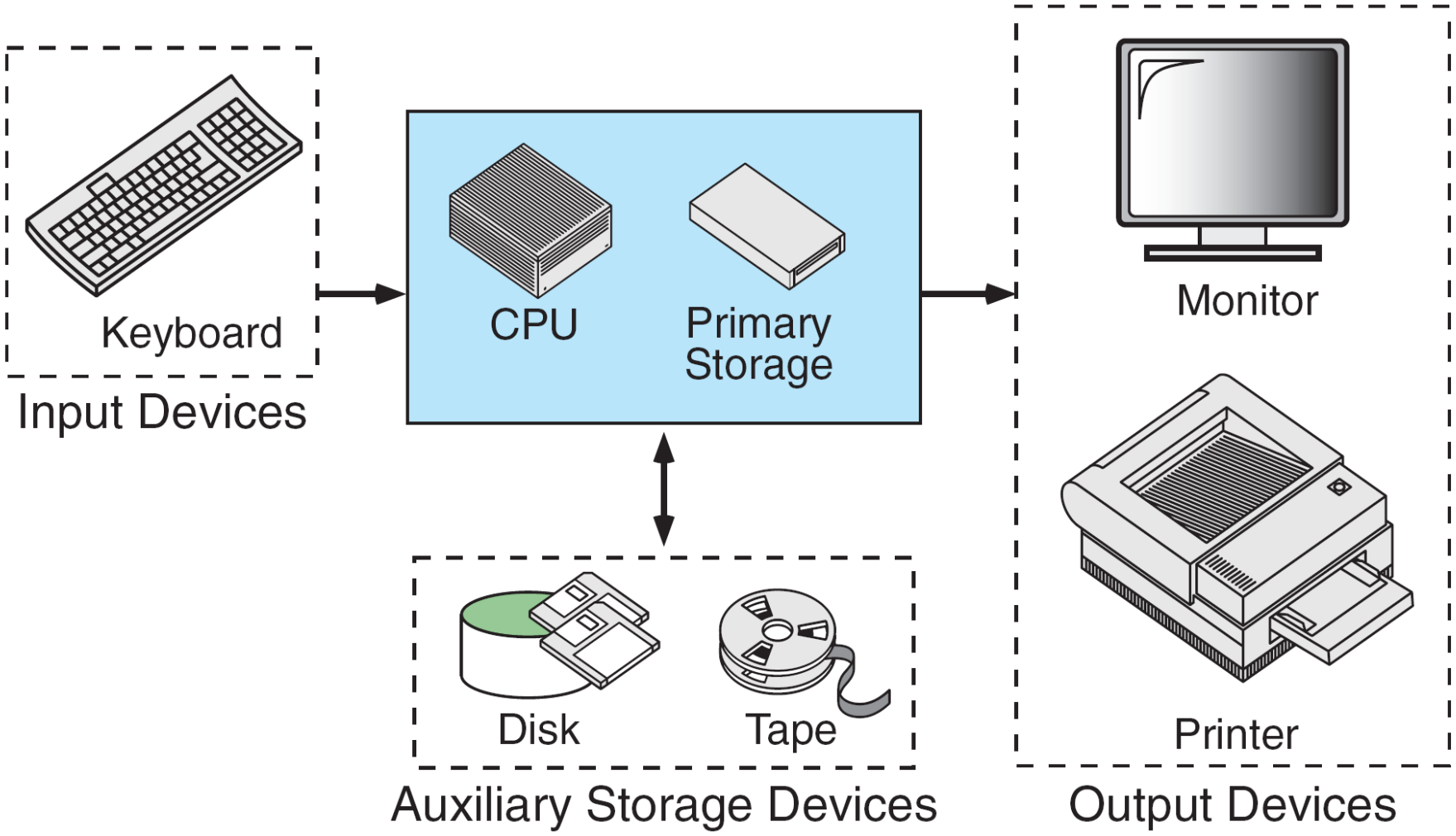
강제(explicit) 형 변환

```
#include <stdio.h>

int main(void) {
	int num1 = 3;
	int num2 = 4;
	double result = num1/num2;

	printf(“Result: %f \n", result); // 0.000000
}
```

I/O

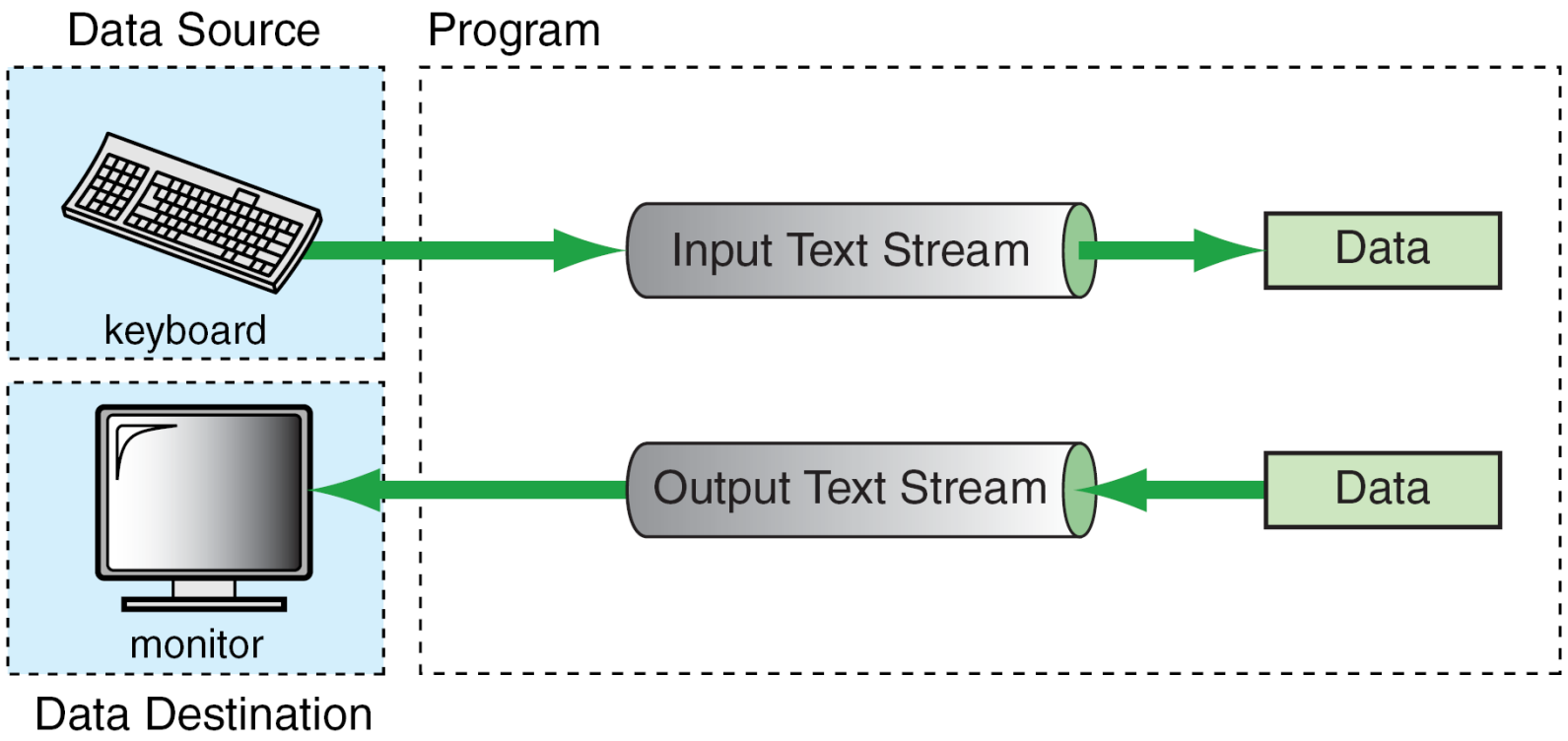
입출력

Overview

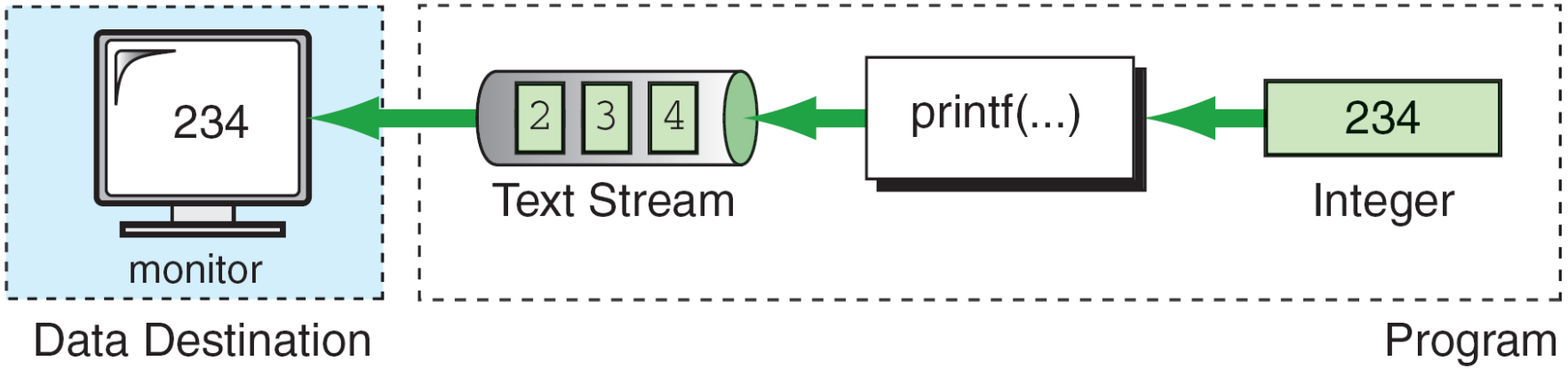

Keyboard
Input Devices
CPU
Primary Storage
Disk
Tape
Auxiliary Storage Devices
Monitor
Printer
Output Devices

Standard I/O Stream

- Header **`<stdio.h>`**: Standard input/output libraries



Output Formatting Concept



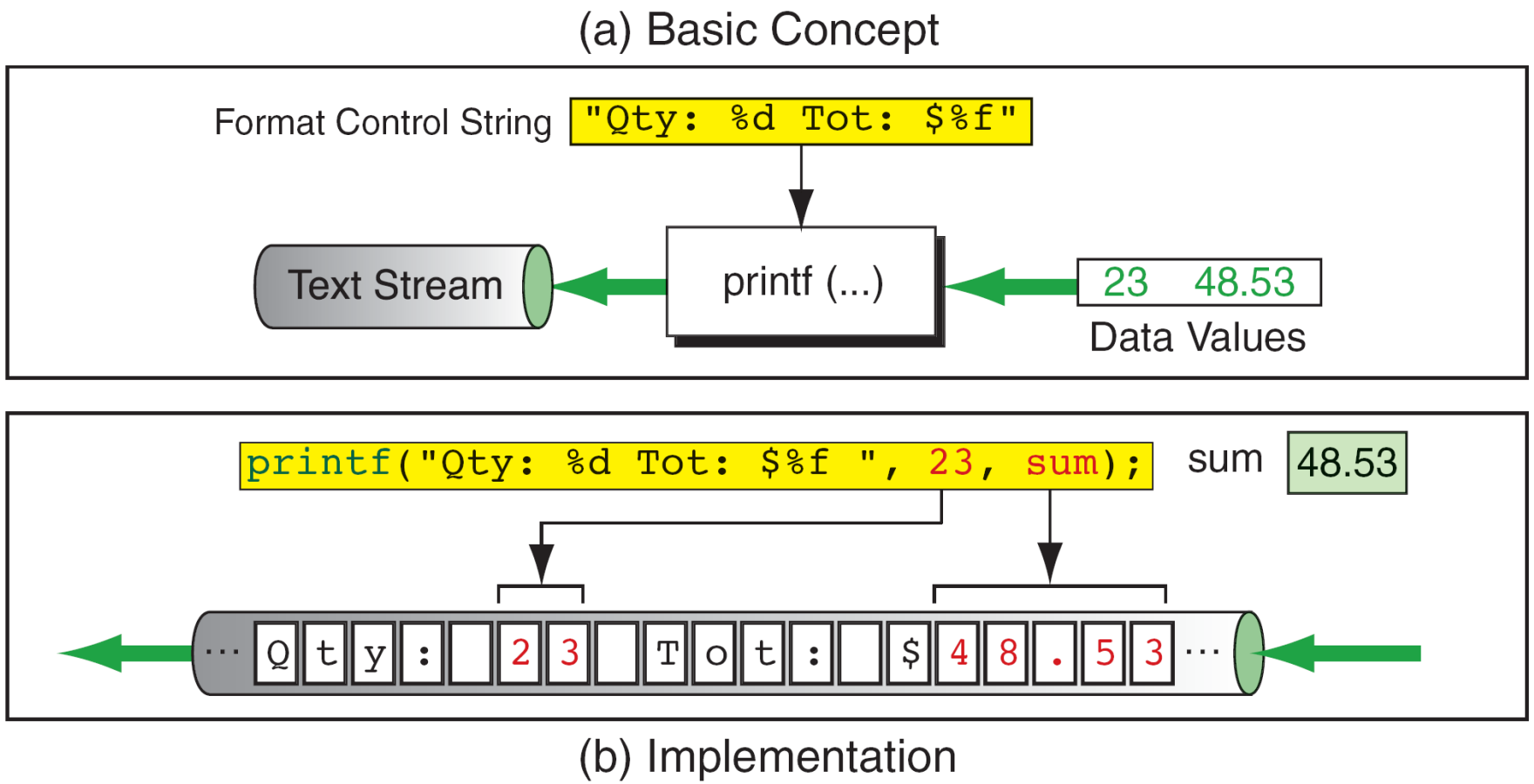
printf

```
int a = 1;
int b = 2;
int c = 3;

printf(“%d  %d  %d\n”, a, b, c);
```

서식문자(Format String) : 데이터의 출력 형태를 지정

ex. %d : 부호 있는 10진 정수로 출력하라!

Output Stream Formatting Example



Format String

출력형식	설 명
%d	10진수 정수형
%o	8진수 정수형
%x, %X	16진수 정수형
%u	부호 없는 10진수 정수형
%c	한 문자
%s	문자열
%f	10진수 방식의 부동소수점 실수형(float형, double형)
%Lf	10진수 방식의 부동소수점 실수형(long double형)
%e, %E	e방식의 부동소수점 실수형
%g, %G	%f와 %e중에서 출력 자릿수를 덜 차지하는 형태로 출력

특수문자

특수문자	설 명
\n	개행
\t	탭
\’	작은 따옴표 출력
\”	큰 따옴표 출력
\?	물음표 출력
\\	₩(역슬래시)출력

```
printf(“C\t언어\n”);                    // C   언어
printf(“\”Hello World\”\n”);          // “Hello World”
```

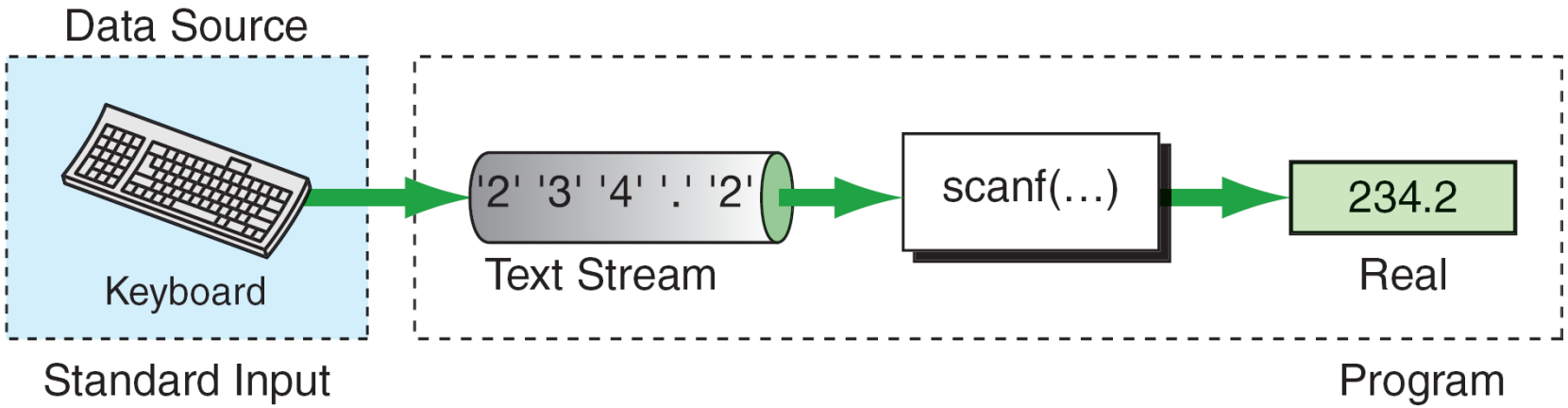
printf

```
#include <stdio.h>
int main(void) {
        int a=30, b=10, c=20;
        double d1=1.23e-4, d2=1.23e-5;
        printf(”%d %x\n”, a, a);        // 30, 1e
        printf(“%o %#o\n”, b, b);       // 12, 012
        printf(“%x %#x\n”, c, c);       // 14, 0x14
        printf(“%f \n”, 0.12345678);    // 0.123457
        printf(“%e\n”, 0.12345678);     // 1.234568e-001
        printf(“%g \n”, d1);            // 0.000123 =>  0.000123
        printf(“%g \n”, d2);            // 0.0000123 => 1.23e-005
}
```

printf

```
#include <stdio.h>

int main(void) {
  printf(“%c %d %s \n”, ‘A’, ‘A’, “ABC”);
  printf(“%10d \n”,123);                  // 오른쪽정렬
  printf(“%-10d \n”,123);                 // 왼쪽 정렬
  printf(“%10.2f \n”,123.456789);         // 왼쪽 정렬 ****123.46(*은 빈칸)
  printf(“%+f %+f\n”,123.45,-123,45);     // +123.45 -123.45
  printf(“% f % f\n”, 123.45,-123,45);    // *123.45 -123.45(*은 빈칸)
  printf(“%-6s %6s \n", “AAA", “BBB");
}
// 소수점 아래는 6자리까지만 문제가 없고 , 7자리부터 문제 발생
```

Formatting Text from an Input Stream



scanf

```
int a;
char b;

scanf(“%d %c”, &a, &b);
```

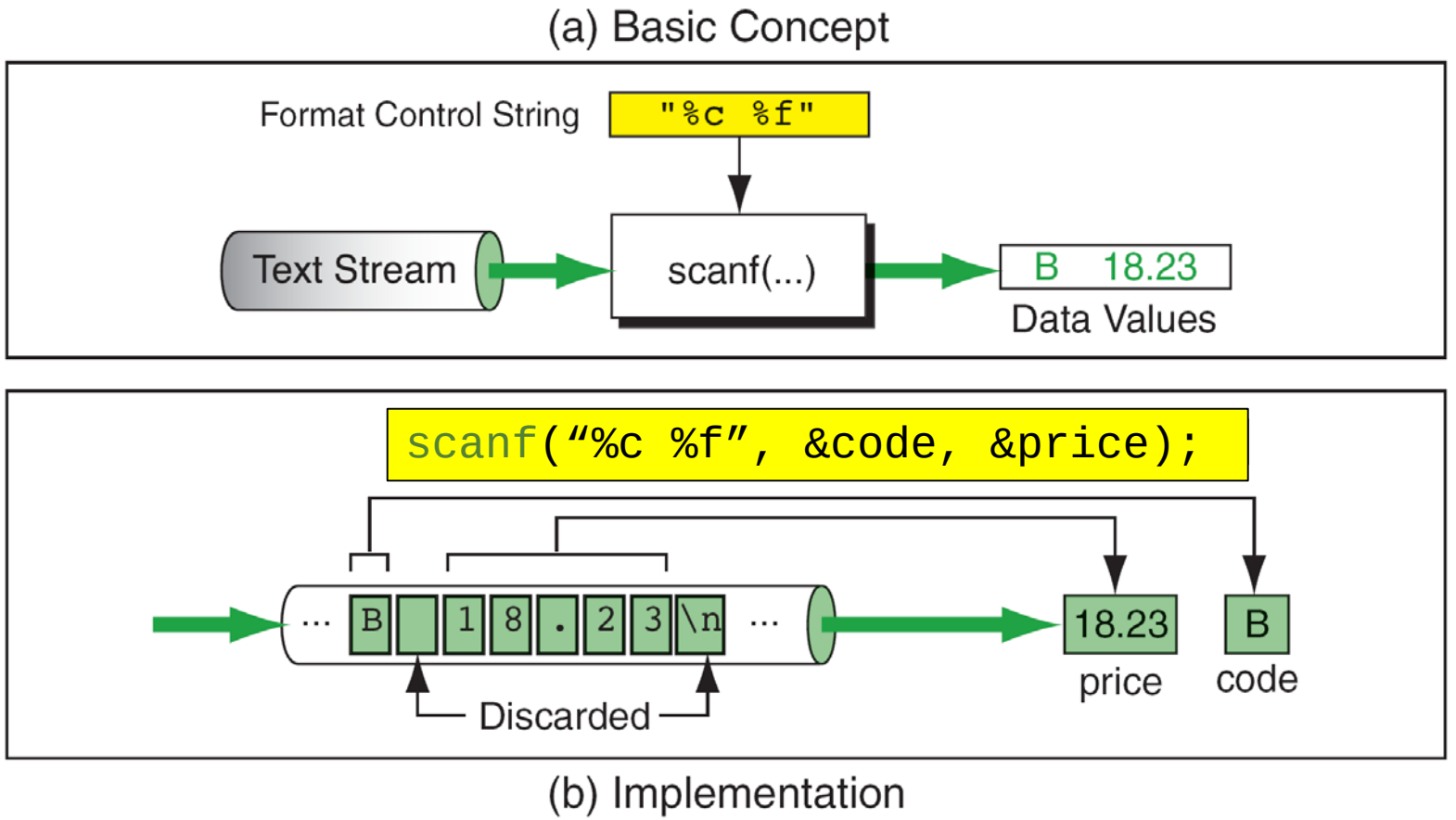
scanf로 값을 받기 위해서는 받는 위치, 즉 주소값을 인자로 주어야 한다.
이유는 다음 시간에. 일단 외우자.

& 가 변수 이름 앞에 붙을 시 그 변수가 가리키는 주소를 의미한다.

A : 변수 이름 / &a : 변수 a 의 주소

서식문자 옵션은 **precision, flag, size** 값 줄 수 없다. 오직 width만 가능

Input Stream Formatting Example



scanf

- scanf(“Format string”, variable a, b, …);
  // 사용자로부터 임의의 데이터를 입력받아 변수에 저장

- 출력할 때
  float, double, long double : %f %f %Lf

- 입력할 때
  float, double, long double : %f %lf %Lf

scanf

- 첫번 째 인자의 서식문자와 뒤의 변수가 대응해야 한다.
  - 갯수, 타입
- 변수에 값을 넣기위해서는 변수의 주소! 를 주어야 한다.
  - &a
- 공백, 서식문자를 제외한 문자가 있으면 입력시 형태를 맞춰줘야 한다.
  - “%d-%d” → 1-2
- Format string 마지막에 공백이 있으면 안됨!
  - “%d %d ” X → “%d %d” O

Usage of scanf

```
#include <stdio.h>

int main(void) {
  int num1, num2, num3;
  printf("세 개의 정수 입력: ");
  scanf("%d %o %x", &num1, &num2, &num3); //12 12 12

  printf("입력된 정수 10진수 출력: ");
  printf("%d %d %d\n", num1, num2, num3); // 12 10 18
  return 0;
}
```

마치기 전에

Tips

- 모르는 것 있으면 물어보기
- 스크린샷이 있으면 더 좋음

- 이런 것도 될까? 싶을 때는 해보는게 최고!
- 책에 있는 예제 코드를 한번씩 직접 코딩해 보자. 재미있잖아 ㅋ

다음 시간

- 연산자(Operators)
- 함수(Functions)