GEDT019
Basis and Practice in Programming

Chapter 3: Data and C

Prof. Tamer ABUHMED
College of Software

성균관대학교
SUNG KYUN KWAN UNIVERSITY(SKKU)

Review of the Precedent Lecture

- Introduced comments in C language
- Explained variable declaration and assignment
- Introduced printf() functions
- Introduced a simple program
- Gave several C keywords (include, int, return)

Lecture Objectives

- To explain the input/output functions
  - printf() and scanf()
- To explain how to define and treat variables of several data types
  - Integers, floating-point, and characters
- To introduce nonprinting sequences
- To give several programming examples
- To give several new C keywords

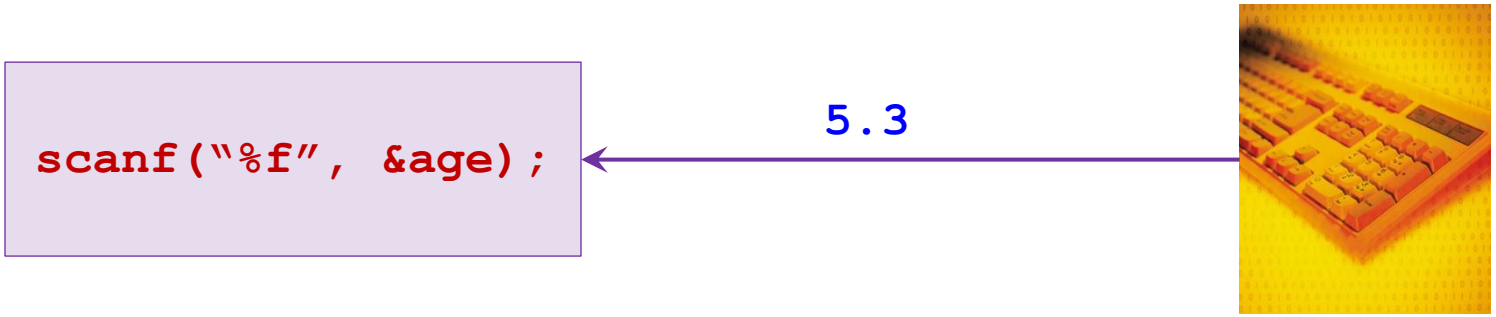
A Simple Program

```
# include <stdio.h>

void main()
{
    float age;                    // variable declaration
    float weight;

    printf("Please enter your age: ");     // display on the screen
    scanf("%f", &age);            // getting data from the keyboard

    printf("Please enter your weight: ");
    scanf("%f", &weight);

    // display on the screen
    printf("Your age is %f and your weight is %f \n", age, weight);

}
```

- What is new?
  - "float" data type: float can hold number with decimal points such as 3.42
  - "%f" specifier: This specifier is used to handle a floating-point variable
  - "scanf()" function: This function reads values or strings from the keyboard

A Simple Program – contd.

- Input/output functions

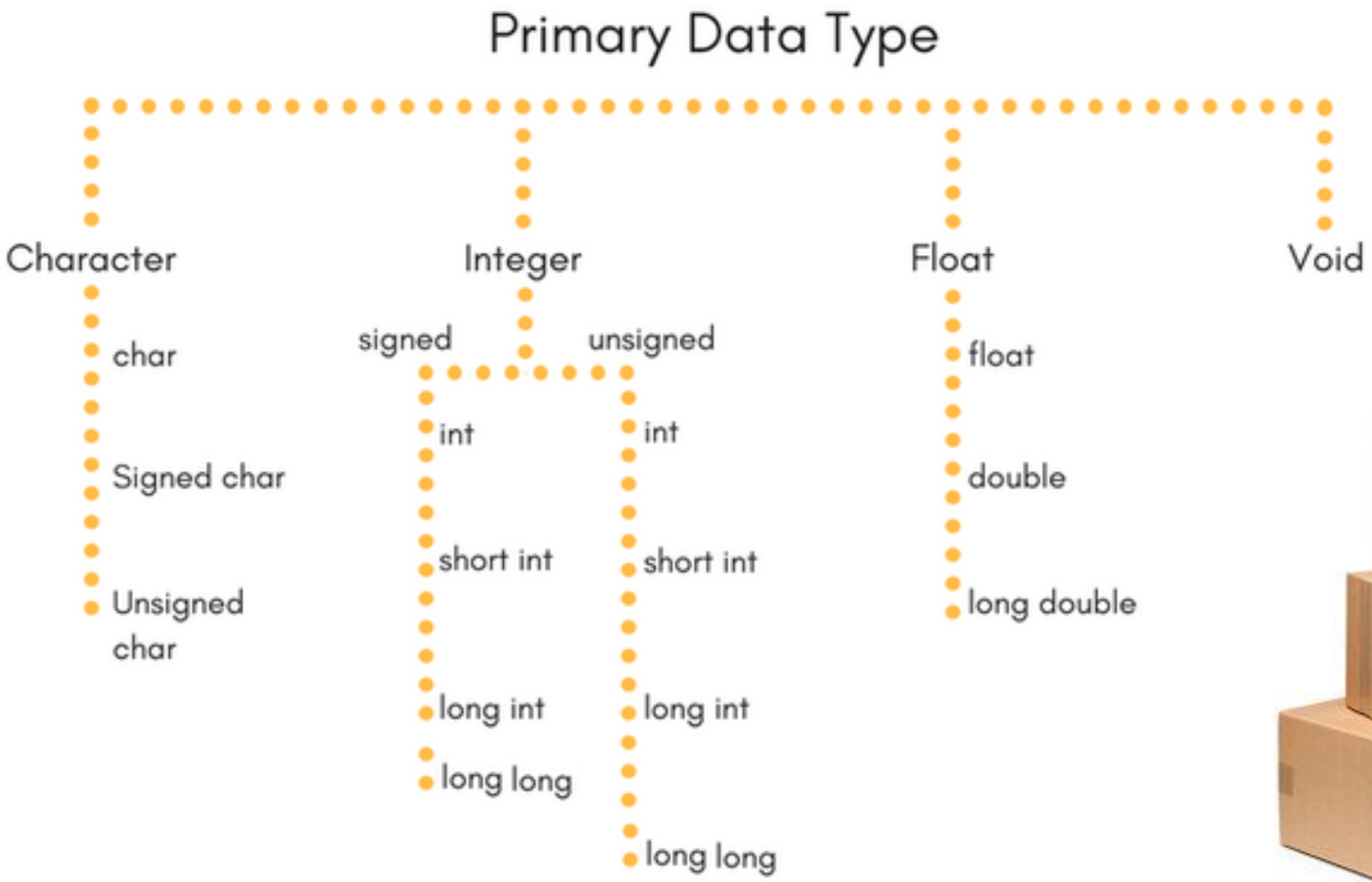


Data-type Keywords

- Data-type
  - The size of each data type might vary from a computer/OS to another
  - On Windows 32bits, MS Visual Studio, you get the following values

Data-type keywords		
Keyword	**Length (Bytes)**	**Example**
int	4	4, 5, 1520, -1526
long	4	4, 5, 1520, -1526
short	2	4, 5, 13, -15
unsigned	4	No sign,
char	1	‘A’, ‘a’, ‘$’, ‘5’, etc.
float	4	1.25, 5, -9.7, 5E-5
double	8	1.25, 5, -9.7, 5E-5

Data-type


Primary Data Type
Character
char
Signed char
Unsigned char
Integer
signed
int
short int
long int
long long
unsigned
int
short int
long int
long long
Float
float
double
long double
Void

Data-type Ranges

Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-128 to 127
int	4 bytes	-2,147,483,648 to 2,147,483,647
unsigned int	4 bytes	0 to 4,294,967,295
short	2 bytes	-32,768 to 32,767
unsigned short	2 bytes	0 to 65,535
long	4 bytes	-2,147,483,648 to 2,147,483,647
long long	8 bytes	-9223372036854775808 to 9223372036854775807
unsigned long long	8 bytes	0 to 18446744073709551615

Size of Data Types

- Variables and memory usage

```c
/* typesize.c -- prints out type sizes */
#include <stdio.h>

void main(void)
{
    printf("Type int has a size of %u bytes.\n", sizeof(int));
    printf("Type char has a size of %u bytes.\n", sizeof(char));
    printf("Type long has a size of %u bytes.\n", sizeof(long));
    printf("Type double has a size of %u bytes.\n", sizeof(double));

}
```

Output

```
Type int has a size of 4 bytes
Type char has a size of 1 byte
Type long has a size of 4 bytes
Type double has a size of 8 bytes
```

Handling Integers

- Integer
  - An integer does not have a fractional part
  - Identifiers: int, short int, signed int, unsigned int, short, signed, unsigned
  - "%d" notation is used to indicate an integer value. More to check on fprint %codes

```c
/* print1.c-displays some properties of printf() */
#include <stdio.h>

int main(void)
{
    int ten = 10;
    int two = 2;

    printf("Doing it right: ");
    printf("%d minus %d is %d\n", ten, 2, ten - two );
    printf("Doing it wrong: ");
    printf("%d minus %d is %d\n", ten );  // forgot 2 arguments

    return 0;
}
```

Output

```
Doing it right: 10 minus 2 is 8
Doing it wrong: 10 minus 1278206344 is 213901120
```

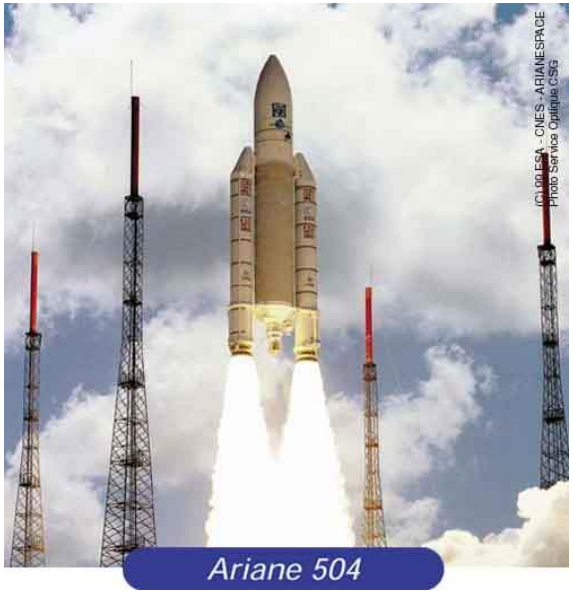
Handling Integers – contd.

- Other types of integers
  - Wrong specifications might produce unexpected results.

```
/* print2.c-more printf() properties */
#include <stdio.h>

void main(void)
{
    unsigned int un = 3000000000;
    short end = 200;                    /* and 16-bit short */
    long big = 65537;

    printf("un = %u and not %d\n", un, un);
    printf("end = %hd and %d\n", end, end);
    printf("big = %ld and not %hd\n", big, big);

}
```

Output

```
un = 3000000000 not -1294967296
end = 200 and 200
big = 65537 not 1
```



Double Int kills Ariane 5

- The EU Space Agency spent ten years and $7B to produce Ariane 5, a giant rocket capable of putting a pair of three-ton satellites into orbit with each launch and intended to give Europe supremacy in the commercial space business.
- All it took to explode the rocket less than a minute into its maiden voyage in 1996 was a small computer program trying to stuff a 64-bit number into a 16-bit space.

Octal & Hexadecimal

- Note
  - Decimal numbers (i.e., integers) are base 10, octal are base 8, and hexadecimal are base 16
  - Example:
    - 100 decimal: $1x10^2 + 0x10^1 + 0x10^0 = 100$
    - 100 decimal = $1x8^2 + 4x8^1 + 4x8^0 = 144$ octal
    - 100 decimal = $6x16^1 + 4\ x16^0 = 64$ hexadecimal

```
/* bases.c--prints 100 in decimal, octal, and hex */
#include <stdio.h>

void main(void)
{
    int x = 100;

    printf("dec = %d; octal = %o; hex = %x\n", x, x, x);
    printf("dec = %d; octal = %#o; hex = %#x\n", x, x, x);
}
```

Output

```
dec = 100; octal = 144, hex = 64
dec = 100, octal = 0144, hex = 0x64
```

Character

- Character representation
  - Each character is represented in computer using the ASCII code
    - For example: the ASCII 65 is equivalent to the letter A
  - The ASCII runs from 0 to 127
    - Therefore, 1 Byte is more than enough to encompass the standard ASCII code

```
char mid;
mid = A;       // No! Compiler thinks A is a variable
mid = "A";     // No! compiler considers "A" as string

mid = 'A';     // Okay.
```

Character – contd.

- Example
  - Try with other characters

```
/* charcode.c-displays code number for a character */
#include <stdio.h>

void main(void)
{
    char ch;

    printf("Please enter a character.\n");
    scanf("%c", &ch);   /* user inputs character */
    printf("The code for %c is %d.\n", ch, ch);

}
```

Output 1

```
Please enter a character.
{
The code for { is 123.
```

Output 2

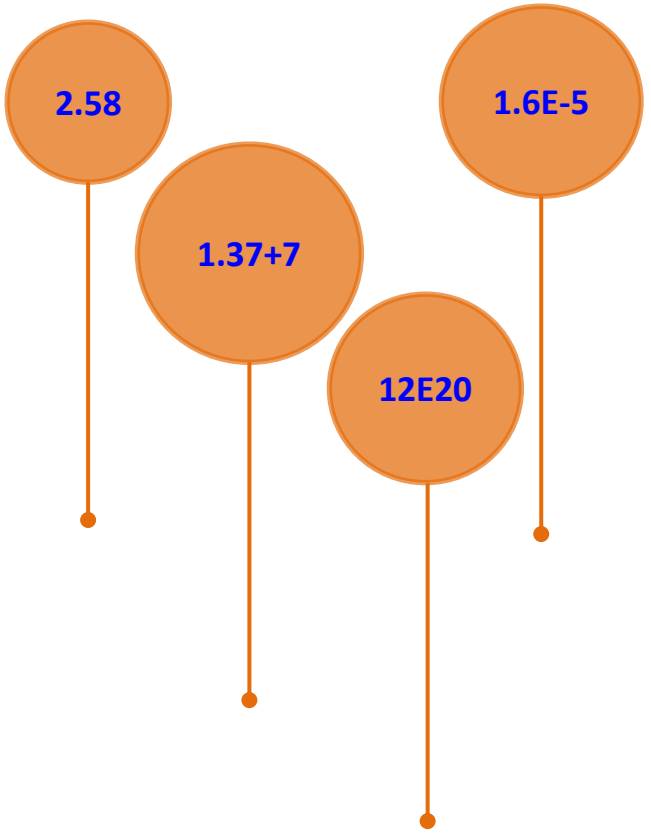
```
Please enter a character.
@
The code for @ is 64.
```

Nonprinting Characters

Nonprinting characters			
Sequence	**Meaning**	**Sequence**	**Meaning**
\a	Alert	\\	Backslash (\)
\b	Backspace	\’	Single quote (‘)
\f	Form feed	\’’	Double quote
\n	Newline	\?	Question mark
\r	Carriage return	\0oo	Octal value (oo)
\t	Horizontal tab	\xhh	Hexadecimal value (hh)
\v	Vertical tab		

Types float and double

- Representation
  - 1.4E15:
    - 1.4 is the significand and 15 is the exponent
  - float data-type
    - 24 bits are used for the significand and its sign
    - 8 bits are used for the exponent and its sign
  - double data-type (in general)
    - 56 bits are used for the significand and its sign
    - 8 bits are used for the exponent and its sign



Types float and double – contd.

- Better printout

```
/* showf_pt.c -- displays float value in two ways */
#include <stdio.h>

void main(void)
{
    float aboat = 32000.0;
    double abet = 2.14e9;
    double dip = 5.32e-5;

    printf("%f can be written %e\n", aboat, aboat);
    printf("%f can be written %e\n", abet, abet);
    printf("%f can be written %e\n", dip, dip);

}
```

Output

```
3200.000000 can be written 3.200000e+004
2140000000.000000 can be written 2.140000e+009
0.000053 can be written 5.320000e-005
```

Example: Escape Sequences

- What is new?
  - \a (alert), \r (carriage return), \b (backspace)
  - "%.2f": Prints the non-fractional part and only the first two digits of the fractional part

```
/* escape.c -- uses escape characters */
# include <stdio.h>
void main(void)
{
    float salary;

    printf("\aEnter your desired monthly salary:");
    printf(" $_______\b\b\b\b\b\b\b");
    scanf("%f", &salary);
    printf("\n\t$%.2f a month is $%.2f a year.",
              salary, salary * 12.0);
    printf("\rGee!\n");

}
```

Output

```
Enter your desired monthly salary: $2000____

      $150.00 a month is $1800.00 a year.
```

Lecture Keywords

- Keywords
  - Input/output functions (printf() and scanf())
  - Data types: integer- and float-precision
  - sizeof(): size of data types
  - Decimal, octal, hexadecimal systems
  - Nonprinting sequences
  - Escape sequences

Lecture Summary

- Explained the input/output functions
  - printf() and scanf()
- Explained how to define and treat variables of several data types
  - Integers, floating-point, and characters
- Introduced nonprinting sequences
  - \n \t \b \r
- Gave several programming examples
- Gave several C keywords
  - Data types keywords (int, char, float, double, long, unsigned, signed, etc.)