VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY

HOCHIMINH CITY INTERNATIONAL UNIVERSITY • HCM - IU •

SCHOOL OF ELECTRICAL ENGINEERING * IU - EE *

Programming for Engineers

Lecture 1B: Simple C Program and Arithmetic in C

Course ID: EE057IU

Lecture Outline

- Simple C Program
  - Comments
  - Preprocessor
  - Blank lines, Spaces, Tabs
  - Main function
  - Body of function
  - Output statement
  - Escape sequence

A Simple C Program

```
1   // fig02_01.c
2   // A first program in C.
3   #include <stdio.h>
4
5   // function main begins program execution
6   int main(void)
7   {
8       printf("Welcome to C!\n");
9   } // end function main
```

```
Welcome to C!
```

A Simple C Program - Comments

- **MATLAB:**
  - Single line: %
  - Multiple lines: %{ … %}
- **Python:**
  - Single line: #
  - Multiple lines: """…""" or ''' … '''
- Now **C Program**
  - Single line: //
  - Multiple lines: /* … */

➢**What is the purpose?**

- Explaining complex logic
- Improving readability
- Temporary disable the code without deleting the code
- Indicate author, date, and purpose of the code or functions

A Simple C Program - Preprocessor

- Line 3: #include <stdio.h>
  - ➢ Indicate a directive to the C preprocessor
  - ➢ stdio.h contains the declaration of "printf" function
- Include the contents of standard input/output header
- Help organize the code into multiple files and reuse common code

```
1   // fig02_01.c
2   // A first program in C.
3   #include <stdio.h>
4
5   // function main begins program execution
6   int main(void)
7   {
8       printf("Welcome to C!\n");
9   } // end function main
```

A Simple C Program – Blank Lines, Spaces, Tabs

- You use blank lines, spaces characters, and tab characters (i.e., “tabs”) to make the program easier to read

- Together, these characters are known as white space. White-space characters are normally ignored by the compiler.

```
1   // fig02_01.c
2   // A first program in C.
3   #include <stdio.h>
4
5   // function main begins program execution
6   int main(void)
7   {
8       printf("Welcome to C!\n");
9   } // end function main
```

A Simple C Program – Main Function

- Line 6: int main(void)
  - ➢ Part of every C program
  - ➢ Entry point of a program where the execution begins
  - ➢ "int" indicates that this function returns an integer
- C program contain one or more functions, but one must be "**main**"
- "**void**" specifies that function "**main**" does not take any parameters or arguments

```
1  // fig02_01.c
2  // A first program in C.
3  #include <stdio.h>
4
5  // function main begins program execution
6  int main(void)
7  {
8      printf("Welcome to C!\n");
9  } // end function main
```

A Simple C Program –Body of Function

- Lines 7 and 9:
  - Left bracket: {, begins of body of every function
  - Corresponding right bracket: }, ends each function
  - This pair of braces and the portion of the program between this braces is called a block

```
1  // fig02_01.c
2  // A first program in C.
3  #include <stdio.h>
4
5  // function main begins program execution
6  int main(void)
7  {
8      printf("Welcome to C!\n");
9  } // end function main
```

A Simple C Program – Output Statement

- Line 8: printf("Welcome to C!\n");
  - "**printf**" is executed to print the message on the screen
  - Welcome to C! with "..." is the message string
  - Entire line ends with a semicolon (;), is called a statement
- Instructs the computer to perform an action
- A string is sometimes called a character string, a message, or a literal

```
1   // fig02_01.c
2   // A first program in C.
3   #include <stdio.h>
4
5   // function main begins program execution
6   int main(void)
7   {
8       printf("Welcome to C!\n");
9   } // end function main
```

A Simple C Program – Escape sequence

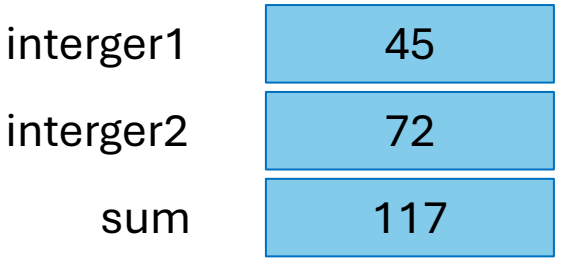
- \n new line
- \t horizontal tab (add 4 to 8 spaces)
- \r carriage return
- \a alert
- \\ backslash
- \” double quote

Simple C Program - Adding Two Integers

```
1   #include <stdio.h>
2
3   // function main begins program execution
4   int main(void) {
5       int integer1 = 0;                   // will hold first number user enters
6       int integer2 = 0;                   // will hold second number user enters
7       printf("Enter first integer: ");     // prompt
8       scanf("%d", &integer1);              // read an integer
9       printf("Enter second integer: ");                   // prompt
10      scanf("%d", &integer2);             // read an integer
11      int sum = 0;                        // variable in which sum will be stored
12      sum = integer1 + integer2;          // assign total to sum
13      printf("Sum is %d\n", sum);         // print sum
14  }                                       // end function main
```

Memory concepts

- Line 5: int integer1 = 0; ->destructive
- Line 6: int integer2 = 0; ->destructive
- Line 11: int sum = 0;
- Line 12: sum = integer1 + integer2;



- Different from MATLAB and Python, C variable requires name, type, and value
- Value 45 is stored to **integer1**'s memory location, value 72 is stored to **integer2**'s memory location, the summation value 117 is stored in **sum**'s memory location
  - Memory locations of **integer1**, **integer2**, and **sum** are not necessarily adjacent
  - When we place values to **integer1** and **integer2**, it is called destructive process
  - When we read values from **integer1** and **integer2**, it is called nondestructive process

Arithmetic in C

C operation		Algebraic	C
Addition	(+)	f+7	f+7
Subtraction	(-)	p-c	p-c
Multiplication	(*)	bm	b*m
Division	(/)	x/y	x/y
Modulus	(%)	r mod s	r%s

- **Example 1:** 7/4 yields 1, 17/5 achieves 3 -> this is called integer division
- **Example 2:** 7%4 yields remainder 3, 17/5 achieves remainder 2

Arithmetic in C – Parentheses (Precedence Order)

- Parenthesis are used in C expressions in the same manner in algebraic expressions
- Parentheses: Highest priority to lowest priority
  - ()
  - *,/,%
  - +,-
  - =

- **Example:** algebra $m = \frac{a+b+c+d+e}{5}$

  C $m = (a + b + c + d + e)/5;$

Arithmetic in C – Parentheses (Precedence Order)

- **Example:** algebra $z = pr\%q + \frac{w}{x} - y$

  C $z = p * r \,\%\, q + w/x - y\,;$

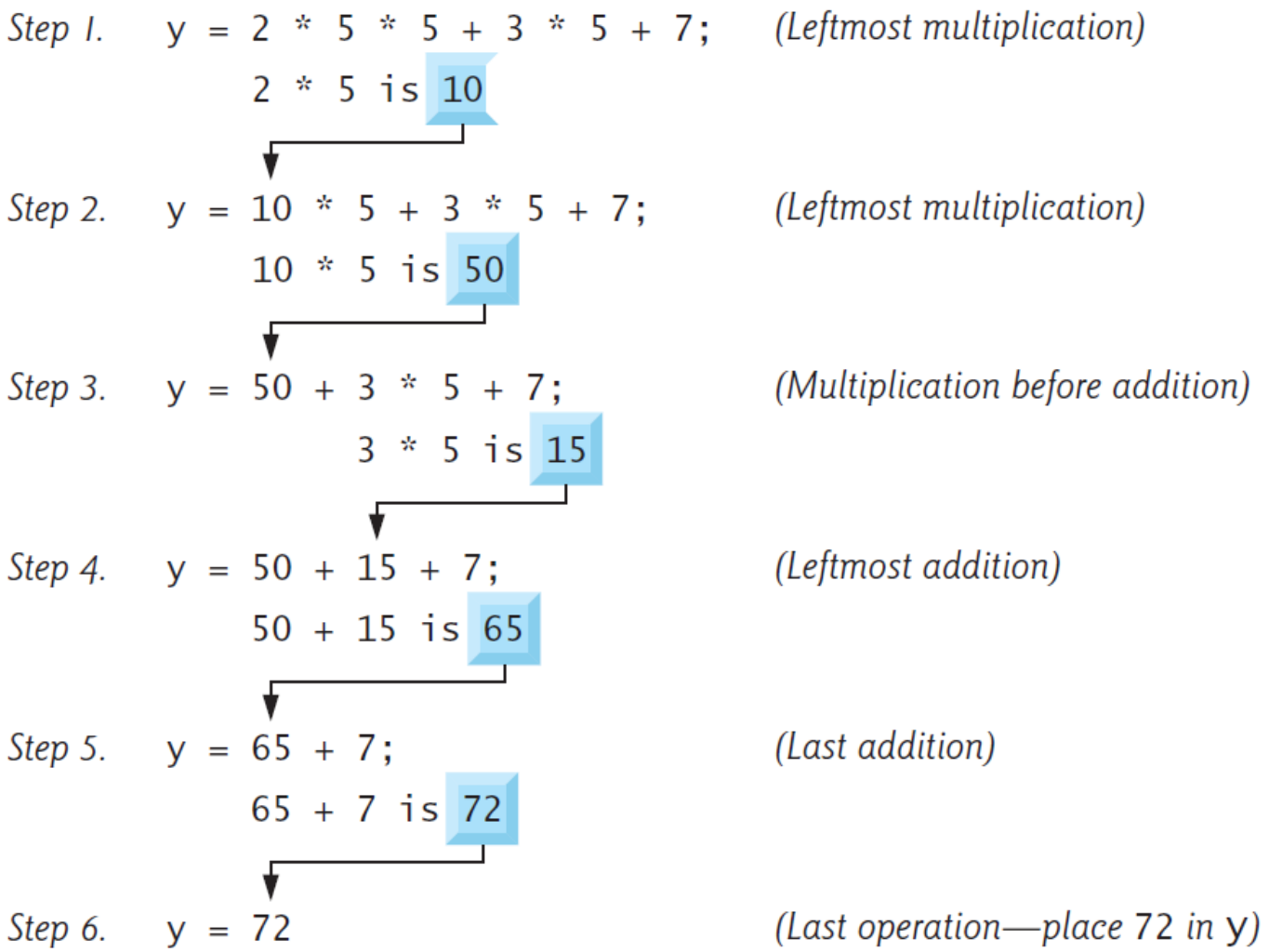
  6 1 2 4 3 5

- **Example:** algebra $a(b + c) + c(d + e)$

  C $a * (b + c) + c * (d + e);$

  3 1 5 4 2

Arithmetic in C – Parentheses (Precedence Order)



KEY TAKEAWAYS

- Basic Structure: Every C program must have a main function, which is the entry point for execution
- Preprocessor Directive: #include <stdio.h> is needed for input/output functions like *printf*
- *Output: The printf function displays* text *on the screen. Use escape sequences like \n for new lines and \" for quotes*
- *Variables: Must be declared with a name and data type (e.g., int sum;) before use*
- *Arithmetic: C uses standard operators (+, -, *, /, %).*

  Integer division (/) truncates the fractional part.

  Modulus (%) returns the remainder.

  Precedence: Operators follow a specific order. Use parentheses () to explicitly control the order of evaluation