EECS 10: Computational Methods in Electrical and Computer Engineering

Lecture 2

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 2.1: Overview

- Review Quiz
- Our second C Program
  - Program structure
    - Input
    - Computation
    - Output
  - Example **`Addition.c`**
    - Variables
    - Value input
    - Calculation
    - Result output

Quiz: Question 1

- Which Linux command shows you the path to the current directory?
  a) **`cd`**
  b) **`pwd`**
  c) **`dir`**
  d) **`ls`**
  e) **`list`**

Quiz: Question 1

- Which Linux command shows you the path to the current directory?
  a) **`cd`**
  b) **`pwd`**
  c) **`dir`**
  d) **`ls`**
  e) **`list`**

Quiz: Question 2

- Which of the following Linux commands renames file “text1” into “homework1”?
  a) **`ren text1 homework1`**
  b) **`ren homework1 text1`**
  c) **`rm text1 homework1`**
  d) **`mv homework1 text1`**
  e) **`mv text1 homework1`**

Quiz: Question 2

- Which of the following Linux commands renames file “text1” into “homework1”?
  a) **`ren text1 homework1`**
  b) **`ren homework1 text1`**
  c) **`rm text1 homework1`**
  d) **`mv homework1 text1`**
  e) **`mv text1 homework1`**

Quiz: Question 3

- What is C *not*?
  a) a structured programming language
  b) a object-oriented programming language
  c) a compiled programming language
  d) a high-level programming language
  e) a portable programming language

Quiz: Question 3

- What is C *not*?
  a) a structured programming language
  b) a object-oriented programming language
  c) a compiled programming language
  d) a high-level programming language
  e) a portable programming language

Quiz: Question 4

- What is the meaning of the following code fragment?

```
/* printf("C programming is great!\n") */
```

a) it prints “C programming is boring!”
b) it prints “C programming is great!”
c) it is a syntax error because a semicolon is missing after the **`printf()`** statement
d) it is the main function of the C program
e) it is a comment ignored by the compiler

Quiz: Question 4

- What is the meaning of the following code fragment?

```
/* printf("C programming is great!\n") */
```

a) it prints “C programming is boring!”
b) it prints “C programming is great!”
c) it is a syntax error because a semicolon is missing after the **`printf()`** statement
d) it is the main function of the C program
e) it is a comment ignored by the compiler

Quiz: Question 5

- What is true about of the following compiler call? (Check all that apply!)

```
% gcc HelloWorld.c –Wall –ansi –o HelloWorld
```

a) the GNU C Compiler is called to generate an executable program called **`HelloWorld`**

b) the compiler will print warning and/or error messages about any non-ANSI compliance in the code

c) the compiler will ignore all warnings

d) the compiler will read the file **`HelloWorld.c`**

e) the compiler will overwrite the **`HelloWorld`** file if it already exists

Quiz: Question 5

- What is true about of the following compiler call? (Check all that apply!)

```
% gcc HelloWorld.c –Wall –ansi –o HelloWorld
```

a) the GNU C Compiler is called to generate an executable program called **`HelloWorld`**

b) the compiler will print warning and/or error messages about any non-ANSI compliance in the code

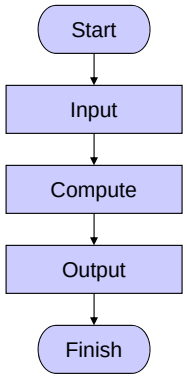
c) the compiler will ignore all warnings

d) the compiler will read the file **`HelloWorld.c`**

e) the compiler will overwrite the **`HelloWorld`** file if it already exists

Program Structure

- General Program Structure
  - Input
    - read input data
  - Computation
    - compute output data from input data
  - Output
    - write output data
- Examples
  - Calculator
    - Enter numbers, compute function, output result
  - Word processor
    - Type, format, print text
  - Database application
    - Enter data, process data, present data
  - etc.

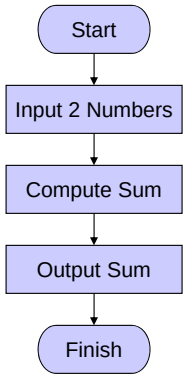


C Program Structure

- Initialization section
  - Definition of variables (storage elements)
    - Name, type, and initial value
- Input section
  - read values from input devices into variables
    - standard input functions
- Computation section
  - perform the necessary computation on variables
    - assignment statements
- Output section
  - write results from variables to output devices
    - standard output functions
- Exit section
  - clean up and exit

Our second C Program

- Program Example: Addition
  - Input
    - Let the user enter two whole numbers
  - Computation
    - Compute the sum of the two numbers
  - Output
    - Display the sum



Our second C Program

- Program example: **`Addition.c`** (part 1/2)

```
/* Addition.c: adding two integer numbers       */
/*                                              */
/* author: Rainer Doemer                        */
/*                                              */
/* modifications:                               */
/* 09/30/04 RD  initial version                 */

#include <stdio.h>

/* main function */

int main(void)
{
    /* variable definitions */
    int i1 = 0;       /* first integer */
    int i2 = 0;       /* second integer */
    int sum;          /* result */
...
```

Our second C Program

- Program example: **`Addition.c`** (part 2/2)

```
...
    /* input section */
    printf("Please enter an integer:      ");
    scanf("%d", &i1);
    printf("Please enter another integer: ");
    scanf("%d", &i2);

    /* computation section */
    sum = i1 + i2;

    /* output section */
    printf("The sum of %d and %d is %d.\n", i1, i2, sum);

    /* exit */
    return 0;
} /* end of main */

/* EOF */
```


Our second C Program

- Variable definition and initialization

```
    /* variable definitions */
    int i1 = 0;       /* first integer */
    int i2 = 0;       /* second integer */
    int sum;          /* result */
```

  - Variable type: **`int`**
    - integer type, stores whole numbers (e.g. -5, 0, 42)
    - many other types exist (**`float`**, **`double`**, **`char`**, ...)
  - Variable name: **`i1`**
    - valid identifier, i.e. name composed of letters, digits
    - variable name should be descriptive
  - Initializer: **`= 0`**
    - specifies the initial value of the variable
    - optional (if omitted, initial value is undefined)

Our second C Program

- Data input using **`scanf()`** function

```
/* input section */
printf("Please enter an integer:      ");
scanf("%d", &i1);
```

  - Function **`scanf()`** is defined in standard I/O library
    - declared in header file **`stdio.h`**
  - … reads data from the standard input stream **`stdin`**
    - **`stdin`** usually means the keyboard
  - … converts input data according to format string
    - **`"%d"`** indicates that a decimal integer value is expected
  - … stores result in specified location
    - **`&i1`** indicates to store at the *address of* variable **`i1`**

Our second C Program

- Computation using assignment statements

```
/* computation section */
sum = i1 + i2;
```

  - Operator **`=`** specifies an assignment
    - value of the right-hand side (**`i1 + i2`**) is assigned to the left-hand side (**`sum`**)
    - left-hand side is usually a variable
    - right-hand side is a simple or complex expression
  - Operator **`+`** specifies addition
    - left and right arguments are added
    - result is the sum of the two arguments
  - Many other operators exist
    - For example, **`-`**, **`*`**, **`/`**, **`%`**, **`<`**, **`>`**, **`==`**, **`^`**, **`&`**, **`|`**, ...

Our second C Program

- Data output using **`printf()`** function

```
/* output section */
printf("The sum of %d and %d is %d.\n", i1, i2, sum);
```

  - Function **`printf()`** is defined in standard I/O library
    - declared in header file **`stdio.h`**
  - … writes data to the standard output stream **`stdout`**
    - **`stdout`** usually means the monitor
  - … converts output data according to format string
    - text (**"The sum…"**) is copied verbatim to the output
    - **"%d"** is replaced with a decimal integer value
  - … takes values from specified arguments (in order)
    - **`i1`** indicates to use the value of the variable **`i1`**

Our second C Program

- Example session: **`Addition.c`**

```
% vi Addition.c
% ls -l
-rw-------   1 doemer   faculty      702 Sep 30 14:17 Addition.c
% gcc -Wall -ansi Addition.c -o Addition
% ls -l
-rwx------   1 doemer   faculty     6628 Sep 30 16:44 Addition*
-rw-------   1 doemer   faculty      702 Sep 30 14:17 Addition.c
% Addition
Please enter an integer:      27
Please enter another integer: 15
The sum of 27 and 15 is 42.
% Addition
Please enter an integer:      123
Please enter another integer: -456
The sum of 123 and -456 is -333.
%
```