

EECS 10: Computational Methods in Electrical and Computer Engineering

Lecture 5

Rainer Dömer

doemer@uci.edu

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

Lecture 5: Overview

- Warm-up Quiz
- Basic Types in C
 - Integer types
 - Floating point types
- Arithmetic Operations in C
 - Arithmetic operators
 - Evaluation order
- Arithmetic Example
 - Cosine approximation
 - Example `cosine.c`

Quiz: Question 1


- Which Unix command shows you the contents of the current directory?
 - a) `pwd`
 - b) `ls`
 - c) `dir`
 - d) `list`
 - e) `cd`

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

3

Quiz: Question 1

- Which Unix command shows you the contents of the current directory?
 - a) `pwd`
 -  b) `ls`
 - c) `dir`
 - d) `list`
 - e) `cd`

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

4

Quiz: Question 2


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

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

5

Quiz: Question 2

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

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

6

Quiz: Question 3


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

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

7

Quiz: Question 3

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

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

8

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

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

Quiz: Question 5

- What is *not* true about of the following compiler call?

```
% gcc -Wall -ansi HelloWorld.c -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 read the file `HelloWorld.c`
- d) the compiler will ignore all warnings
- e) the compiler will assume that `HelloWorld.c` is an ANSI-compliant C program

EECS10: Computational Methods in ECE, Lecture 5


(c) 2006 R. Doemer

11

Quiz: Question 5

- What is *not* true about of the following compiler call?

```
% gcc -Wall -ansi HelloWorld.c -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 read the file `HelloWorld.c`
-  d) the compiler will ignore all warnings
- e) the compiler will assume that `HelloWorld.c` is an ANSI-compliant C program

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

12

Basic Types in C

- Integer types
 - **char** Character, e.g. `'a'`, `'b'`, `'1'`, `'*'`
 - typical range [-128,127]
 - **short int** Short integer, e.g. -7, 0, 42
 - typical range [-32768,32767]
 - **int** Integer, e.g. -7, 0, 42
 - typical range [-2147483648,2147483647]
 - **long int** Long integer, e.g. -991, 91, 1231
 - typical range [-2147483648,2147483647]
 - **long long int** Very long integer, e.g. 1234511
 - typical range [-9223372036854775808, 9223372036854775807]
- Integer types can be
 - **signed** negative and positive values (and 0)
 - **unsigned** positive values only (and 0)

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

13

Basic Types in C

- Floating point types
 - **float** Floating point with single precision
 - Example 3.5f, -0.234f, 10E8f
 - **double** Floating point with double precision
 - Example 3.5, -0.23456789012, 10E88
 - **long double** Floating point with high precision
 - Example 12345678.123456E1231
- Floating point values are in many cases *approximations* only!
 - Storage size of floating point values is fixed
 - Many values can only be represented as approximations
 - Example: $1.0/3.0 = .333333$

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

14

Conversion Specifiers for Basic Types

Type	printf()	scanf()
• long double	%Lf	%Lf
• double	%f	%lf
• float	%f	%f
• unsigned long long	%llu	%llu
• long long	%lld	%lld
• unsigned long	%lu	%lu
• long	%ld	%ld
• unsigned int	%u	%u
• int	%d	%d
• short	%hd	%hd
• char	%c	%c

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

15

Arithmetic Operations in C

- Arithmetic Operators
 - parentheses (,)
 - unary plus, minus +, -
 - multiplication, division, modulo *, /, %
 - addition, subtraction +, -
 - shift left, shift right <<, >>
- Evaluation order of expressions
 - usually left to right
 - by operator precedence
 - ordered as in table above (higher operators are evaluated first)
- Arithmetic operators are available
 - for integer types: all
 - for floating point types: all except %, <<, >>

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

16

Example Program

- Cosine function approximation
 - Task
 - Design a program to compute the cosine function!
 - In your program, use only the four basic operations addition, subtraction, multiplication, and division.
 - Approach
 - The cosine function can be algebraically approximated using an infinite sum

$$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n}}{(2n)!} \approx 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$$

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

17

Example Program

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

```

/* Cosine.c: cosine function approximation */
/* */
/* author: Rainer Doemer */
/* */
/* modifications: */
/* 10/02/05 RD initial version */

#include <stdio.h>

/* main function */
int main(void)
{
    /* variable definitions */
    double x, y;

    /* input section */
    printf("Please enter real value x: ");
    scanf("%lf", &x);
    ...

```

EECS10: Computational Methods in ECE, Lecture 5

(c) 2006 R. Doemer

18

Example Program

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

```

...

/* computation section */
y = 1 - (x*x)/(2.0*1.0)
  + (x*x*x*x)/(4.0*3.0*2.0*1.0)
  - (x*x*x*x*x*x)/(6.0*5.0*4.0*3.0*2.0*1.0);

/* output section */
printf("cos(%f) is approximately %f\n", x, y);

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

/* EOF */

```

Example Program

- Example session: `Cosine.c`

```

% vi Arithmetic.c
% vi Cosine.c
% gcc -Wall -ansi Cosine.c -o Cosine
% Cosine
Please enter real value x: 0.0
cos(0.000000) is approximately 1.000000
% Cosine
Please enter real value x: 0.1
cos(0.100000) is approximately 0.995004
% Cosine
Please enter real value x: 1.57079
cos(1.570790) is approximately -0.000888
% Cosine
Please enter real value x: 3.1415927
cos(3.141593) is approximately -1.211353
%

```