EECS 10: Computational Methods in Electrical and Computer Engineering

Lecture 4

Rainer Dömer

doemer@uci.edu

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

Lecture 4.1: Overview

- Formatted Input
  - Format specifiers for **`scanf()`**
  - Detailed formatting of integral values
  - Detailed formatting of floating-point values
- Formatted Output
  - Format specifiers for **`printf()`**
  - Detailed formatting of integral values
  - Detailed formatting of floating-point values
- Example **`Formatting.c`**

Formatted Input

- Formatted input using **`scanf()`**
  - standard format specifier for integral values
    - (**`unsigned`**) **`long long`** **`%llu`** **`%lld`**
    - (**`unsigned`**) **`long`** **`%lu`** **`%ld`**
    - (**`unsigned`**) **`int`** **`%u`** **`%d`**
    - (**`unsigned`**) **`short`** **`%hu`** **`%hd`**
    - (**`unsigned`**) **`char`** **`%c`** (reads a character)
  - standard format specifier for floating point values
    - **`long double`** **`%Lf`**
    - **`double`** **`%lf`**
    - **`float`** **`%f`**

Formatted Output

- Formatted output using **`printf()`**
  - standard format specifier for integral values
    - (**`unsigned`**) **`long long`** **`%llu`** **`%lld`**
    - (**`unsigned`**) **`long`** **`%lu`** **`%ld`**
    - (**`unsigned`**) **`int`** **`%u`** **`%d`**
    - (**`unsigned`**) **`short`** **`%hu`** **`%hd`**
    - (**`unsigned`**) **`char`** **`%c`** (prints a character)
  - standard format specifier for floating point values
    - **`long double`** **`%Lf`**
    - **`double`** **`%f`**
    - **`float`** **`%f`**

Formatted Output

- Detailed formatting sequence for integral values
  - **% *flags width length conversion***
  - *flags*
    - (none) standard formatting (right-justified)
    - **-** left-justified output
    - **+** leading plus-sign for positive values
    - **0** leading zeros
  - field *width*
    - (none) minimum number of characters needed
    - integer width of field to be filled with output
  - *length* modifier
    - (none) **int** type
    - **h** **short int** type
    - **l** **long int** type
    - **ll** **long long int** type
  - *conversion* specifier
    - **d** signed decimal value
    - **u** unsigned decimal value
    - **o** (unsigned) octal value
    - **x** (unsigned) hexadecimal value using characters **0-9**, **a-f**
    - **X** (unsigned) hexadecimal value using characters **0-9**, **A-F**

Formatted Output

- Detailed formatting sequence for floating-point values
  - **% *flags width precision length conversion***
  - *flags*
    - (none) standard formatting (right-justified)
    - **-** left-justified output
    - **+** leading plus-sign for positive values
    - **0** leading zeros
  - field *width*
    - (none) minimum number of characters needed
    - integer width of field to be filled with output
  - *precision*
    - (none) default precision (e.g. 6)
    - **.**int number of digits after decimal point (for **f**, **e**, or **E**), maximum number of significant digits (for **g**, or **G**)
  - *length* modifier
    - (none) **float** or **double** type
    - **L** **long double** type
  - *conversion* specifier
    - **f** standard floating-point notation (fixed-point)
    - **e** or **E** exponential notation (using **e** or **E**)
    - **g** or **G** standard or exponential notation (using **e** or **E**)

Formatted Output

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

```
/* Formatting.c: formatted output demo          */
/* author: Rainer Doemer                        */
/* modifications:                               */
/* 10/19/04 RD  initial version                 */

#include <stdio.h>

/* main function */

int main(void)
{
  /* output section */
  printf("42 formatted as |%%d|:   |%d|\n", 42);
  printf("42 formatted as |%%8d|:  |%8d|\n", 42);
  printf("42 formatted as |%%-8d|: |%-8d|\n", 42);
  printf("42 formatted as |%%+8d|: |%+8d|\n", 42);
  printf("42 formatted as |%%08d|: |%08d|\n", 42);
  printf("42 formatted as |%%x|:   |%x|\n", 42);
  printf("42 formatted as |%%o|:   |%o|\n", 42);
...
```


Formatted Output

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

```
...
  printf("\n");
  printf("123.456 formatted as |%%f|:     |%f|\n", 123.456);
  printf("123.456 formatted as |%%e|:     |%e|\n", 123.456);
  printf("123.456 formatted as |%%g|:     |%g|\n", 123.456);
  printf("123.456 formatted as |%%12.4f|: |%12.4f|\n",
                                                  123.456);
  printf("123.456 formatted as |%%12.4e|: |%12.4e|\n",
                                                  123.456);
  printf("123.456 formatted as |%%12.4g|: |%12.4g|\n",
                                                  123.456);

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

/* EOF */
```

Formatted Output

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

```
% vi Formatting.c
% gcc Formatting.c -o Formatting -Wall -ansi
% Formatting
42 formatted as |%d|:   |42|
42 formatted as |%8d|:  |      42|
42 formatted as |%-8d|: |42      |
42 formatted as |%+8d|: |     +42|
42 formatted as |%08d|: |00000042|
42 formatted as |%x|:   |2a|
42 formatted as |%o|:   |52|

123.456 formatted as |%f|:     |123.456000|
123.456 formatted as |%e|:     |1.234560e+02|
123.456 formatted as |%g|:     |123.456|
123.456 formatted as |%12.4f|: |    123.4560|
123.456 formatted as |%12.4e|: |  1.2346e+02|
123.456 formatted as |%12.4g|: |       123.5|
%
```