I/O Issues in C

Jiani ZHANG

CSCI2100B Data Structures Tutorial 3

Content

- **Introduction**
- **Input/Output Functions**
  - **printf()**
  - **scanf()**
  - **getchar() and putchar()**
- **Constants**
  - **Constants**
  - **#define**
  - **String Literal**

Content

- **Introduction**
- **Input/Output Functions**
  - **printf()**
  - **scanf()**
  - **getchar() and putchar()**
- **Constants**
  - **Constants**
  - **#define**
  - **String Literal**

How to read the input data and write the output data?

Exercise 1.22 *You are asked to write a C program to find the number of digits within a string.*

Input *The input consists of the number of test cases, m, in the first line and followed by m test cases.*

Each test case consists of a string with less than 256 characters

An example is as follows,

```
3
I hate CUSIS, I have failed my course registration 5 times.
Sherlock Holmes is looking for a flatmate to share a flat at 221B Baker Street
I won 5,000,000.00 dollars on 25 July 2010
```

Output *The output should be m lines of integer value. Each line should be the number of digits within that string.*

```
1
3
15
```

What's on Judge System

```
config.txt  in1  in2  in3  in4  out1  out2  out3  out4
```

config.txt

```
1 CASE=4
2 INPUT=in%i
3 OUTPUT=out%i
4 TIME_LIMIT=1
5 MEM_LIMIT=32
```

int1

```
 1 23
 2 53102 57860
 3 12161 56143
 4 19600 26579
 5 19385 62725
 6 19133 9183
 7 5064 5997
 8 58801 33145
 9 18117 52016
10 29410 21300
11 9057 1659
12 30309 38452
13 40689 14163
14 19912 58454
15 61974 39943
16 32396 2740
17 60600 63742
18 54350 19962
19 46542 7226
20 4415 8414
21 17597 391
22 0 65535
23 0 0
24 1 10
```

out1

```
 1 8
 2 10
 3 7
 4 11
 5 7
 6 5
 7 6
 8 10
 9 7
10 6
11 6
12 7
13 9
14 7
15 7
16 5
17 8
18 9
19 7
20 7
21 16
22 0
23 3
```

C program skeleton

- In short, the basic skeleton of a C program looks like this:

```
#include <stdio.h>
int main(void)
{
    statement(s);
    return(0);
}
```

Preprocessor directives

Function main

Start of segment

End of segment

Input/Output Operations

- **Input operation**
  - an instruction that copies data from an input device into memory
- **Output operation**
  - an instruction that displays information stored in memory to the output devices (such as the monitor screen)

Input/Output Functions

- A C function that performs an input or output operation
- A few functions that are pre-defined in the header file **<stdio.h>** such as :
  - ***printf()***
  - ***scanf()***
  - ***getchar()*** & ***putchar()***

Content

- Introduction
- **Input/Output Functions**
  - **printf()**
  - scanf()
  - getchar() and putchar()
- Constants
  - Constants
  - #define
  - String Literal

The *printf()* function

- Used to send data to the standard output (usually the monitor) to be printed according to specific format.
- General format:
  - `printf("string literal");`
    - A sequence of any number of characters surrounded by double quotation marks.
  - `printf("format string", variables);`
    - Format string is a combination of **text**, **conversion specifier** and **escape sequence**.

The *printf()* function cont…

- Example:
  - `printf("Thank you\n");`

```
Thank you
```

  - `printf ("Total sum is: %d\n", sum);`

```
Total sum is: 50
```

Assuming that the value of sum is 50

- **%d** is a **placeholder** (conversion specifier)
  - marks the display position for a type integer variable
  - Common Conversion Identifier used in printf function
- **\n** is an **escape sequence**
  - moves the cursor to the new line

	printf
int	%d
float	%f
double	%f
char	%c
string	%s

Placeholder/Conversion Specifier

No	Conversion Specifier	Output Type	Output Example
1	%d	Signed decimal integer	76
2	%i	Signed decimal integer	76
3	%o	Unsigned octal integer	134
4	%u	Unsigned decimal integer	76
5	%x	Unsigned hexadecimal (small letter)	9c
6	%X	Unsigned hexadecimal (capital letter)	9C
7	%f	Integer including decimal point	76.0000
8	%e	Signed floating point (using e notation)	7.6000e+01
9	%E	Signed floating point (using E notation)	7.6000E+01
10	%g	The shorter between %f and %e	76
11	%G	The shorter between %f and %E	76
12	%c	Character	‘7’
13	%s	String	‘76'

Escape Sequence

Escape Sequence	Effect
\a	Beep sound
\b	Backspace
\f	Formfeed (for printing)
\n	New line
\r	Carriage return
\t	Tab
\v	Vertical tab
\\	Backslash
\”	“ sign
\o	Octal decimal
\x	Hexadecimal
\0	NULL

Content

- Introduction
- Input/Output Functions
  - printf()
  - **scanf()**
  - getchar() and putchar()
- Constants
  - Constants
  - #define
  - String Literal

The scanf() function

- Read data from the standard input device (usually keyboard) and store it in a variable.
- General format:
  - `scanf("format string", &variable);`
- Notice ampersand (&) operator :
  - C address of operator
  - it passes the address of the variable instead of the variable itself
  - tells the scanf() where to find the variable to store the new value
- Format string is a combination of conversion specifier and escape sequence (if any).

The scanf() function cont…

- Common Conversion Identifier used in printf and scanf functions.

	printf	scanf
int	%d	%d
float	%f	%f
double	%f	%lf
char	%c	%c
string	%s	%s

- Example :

```
int age;
printf("Enter your age:");
scanf("%d", &age);
```

The scanf() function cont…

- If you want the user to enter more than one value, you serialize the inputs.
- Example:

```
float height, weight;

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

Content

- Introduction
- **Input/Output Functions**
  - printf()
  - scanf()
  - **getchar() and putchar()**
- Constants / #define
- String Literal

getchar() and putchar()

- ***getchar()*** - read ***a*** character from standard input
- ***putchar()*** - write ***a*** character to standard output
- Example:

```
Please type a character: h
You have typed this character: h
```

```
#include <stdio.h>
int main(void)
{
   char my_char;
   printf("Please type a character:");
   my_char = getchar();
   printf("You have typed this character: ");
   putchar(my_char);
   return (0);
}
```

getchar() and putchar() cont

- Alternatively, you can write the previous code using normal ***printf*** / ***scanf*** and %c placeholder.
- Example:

```
Please type a character: h
You have typed this character: h
```

```
#include <stdio.h>
int main(void)
{
  char my_char;
  printf("Please type a character: ");
  scanf("%c",&my_char);
  printf("You have typed this character: %c", my_char);
  return(0);
}
```

Content

- Introduction
- Input/Output Functions
  - printf()
  - scanf()
  - getchar() and putchar()
- **Constants**
  - Constants
  - #define
  - String Literal

Content

- Introduction
- Input/Output Functions
  - printf()
  - scanf()
  - getchar() and putchar()
- Constants
  - **Constants**
  - #define
  - String Literal

Constants

- **Character constants**
  - A character enclosed in a single quotation mark
  - Example:
    - `const char letter = 'n';`
    - `const char number = '1';`
    - `printf("%c", 'S');`
- **Enumeration**
  - Values are given as a list
  - Example:
    - `enum Days { Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday };`

Constant example – volume of a cone

```
#include <stdio.h>

int main(void)
{
   const double pi = 3.412;
   double height, radius, base, volume;

   printf("Enter the height and radius of the cone:");
   scanf("%lf %lf", &height, &radius);

   base = pi * radius * radius;
   volume = (1.0/3.0) * base * height;

   printf("The volume of a cone is %f ", volume);
   return (0);
}
```

Content

- Introduction
- Input/Output Functions
  - printf()
  - scanf()
  - getchar() and putchar()
- Constants
  - Constants
  - **#define**
  - String Literal

#define

```
#include <stdio.h>
#define pi 3.142

int main(void)
{
   double height, radius, base, volume;

   printf("Enter the height and radius of the
   cone:");
   scanf("%lf %lf", &height, &radius);

   base = pi * radius * radius;
   volume = (1.0/3.0) * base * height;

  printf("The volume of a cone is %f ", volume);
   return (0);
}
```

Content

- Introduction
- Input/Output Functions
  - printf()
  - scanf()
  - getchar() and putchar()
- Constants
  - Constants
  - #define
  - **String Literal**

String Literal

- A sequence of any number of characters surrounded by double quotation marks “ ”.
- Example of usage in C program:

```
printf(“What a beautiful day.\n”);
```

```
What a beautiful day.
```

- To have double quotation marks as part of the sentence, precede the quote with backslash

```
printf(“He shouted \“stop!\” to the thief.\n”);
```

```
He shouted “stop!” to the thief.
```

Thanks!

Q&A