

Structured Programming - Formatted I/O

Donglong Chen

Structured Programming

With **Thanks** to Dr. Xin Feng and Dr. Haipeng Guo

Outline

- Write output to screen - printf
- Read input from keyboard– scanf

printf

- **Format:** `printf(format_string, arg1, arg2, ..., argn);`
- Display the output

```
printf("The results are %d and %d\n", a, b);
```



Format_string - included in a pair of " "

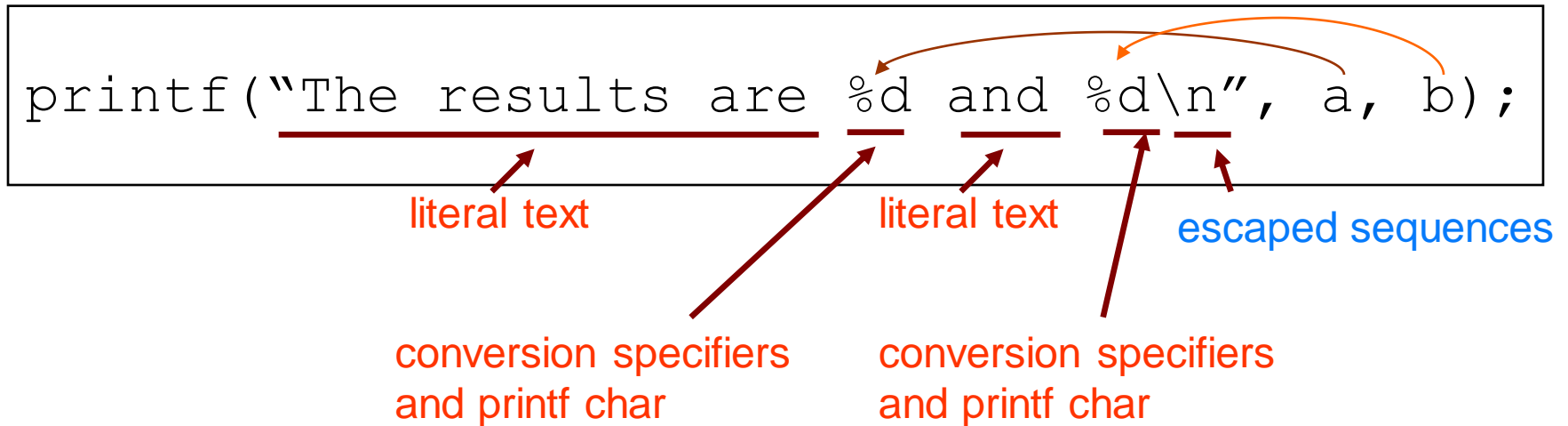
arg1

arg2

printf

- Format string contains (optional):
 - **literal text**: printed without any variation
 - **escape sequences**: preceeded by `\`, used to print special characters
 - **conversion specifiers** : `%` followed by a single character (printf char)
 - Conversion specifier indicates (usually) that **the value of a variable is to be displayed at this location**. The variables to be printed must appear as `arg1`, `arg2`, ...in the order that they appear in the format string.

printf



If `a = 10` and `b = 20`, we have the following output:

The results are 10 and 20

Conversion Specifiers

Conversion Specifier	Meaning
%c	Single character
%d	Signed decimal integer
%x	Hexadecimal number
%f	Decimal floating point number
%e	Floating point in "scientific notation"
%s	Character string (more on this later)
%u	Unsigned decimal integer
%%	Just print a % sign
%ld, %lld	long, and long long

More Conversion Specifiers

- More conversion specifiers are available.
- They can be found in the Internet and reference books.
- Feel free to search via Google or Bing. Better NOT use Baidu for scientific search.
- <https://www.tutorialspoint.com/format-specifiers-in-c>
- <https://www.geeksforgeeks.org/format-specifiers-in-c/>
- <http://crasseux.com/books/ctutorial/Formatted-output-conversion-specifiers.html>

Escape Sequences

Sequence	Meaning
<code>\a</code>	Bell (an alert sound)
<code>\b</code>	Backspace
<code>\n</code>	Newline
<code>\t</code>	Horizontal tab
<code>\\</code>	Backslash
<code>\'</code>	Single quote
<code>\"</code>	Double quotation
<code>\xhh</code>	ASCII char specified by hex digits hh
<code>\ooo</code>	ASCII char specified by octal digits oo

Exercises

- What is the output of the following program

```
#include <stdio.h>
int main()
{
    char c, d;
    float f;
    c = 'd';
    d = 97;
    f = 23.5;
    printf("c = %c, d = %c \nf = %f, f = %e", c,
           d, f, f);
}
```

Exercises

- What is the output of the following program

```
#include <stdio.h>
int main()
{
    char c, d;
    float f;
    c = 'd';
    d = 97;
    f = 23.5;
    printf("d = %d \tf = %d", d, f);
}
```

Exercises

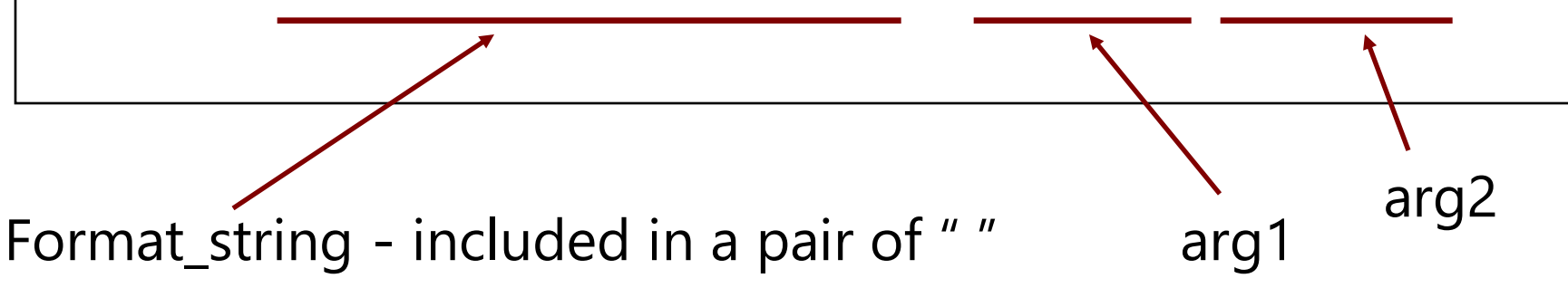
- What is the output of the following program

```
#include <stdio.h>
int main() {
    int ten = 10, x = 42;
    char ch1 = 'o', ch2 = 'f';
    printf("%d%% %c%c %d is %f\n",
           ten, ch1, ch2, x, 1.0 * x / ten );
    return 0;
}
```

scanf

- Format: `scanf(format_string, arg1, arg2, ..., argn);`
- Read the input
- Format string is similar to that in `printf`

```
scanf("value=%d, ratio=%f", &value, &ratio);
```



Format_string - included in a pair of " "

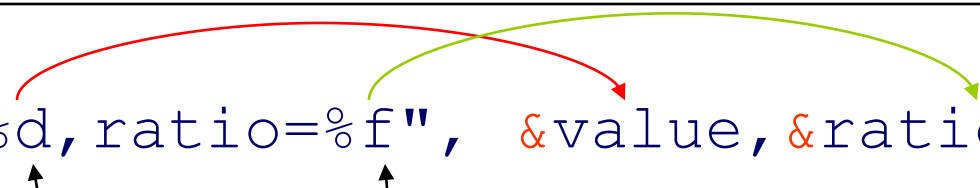
arg1

arg2

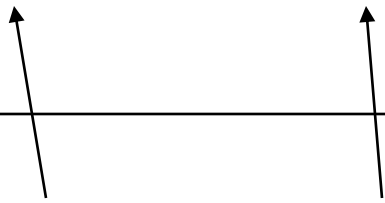
&: means "address of". Will be explained in the later chapter.

scanf

```
scanf("value=%d,ratio=%f",&value,&ratio);
```

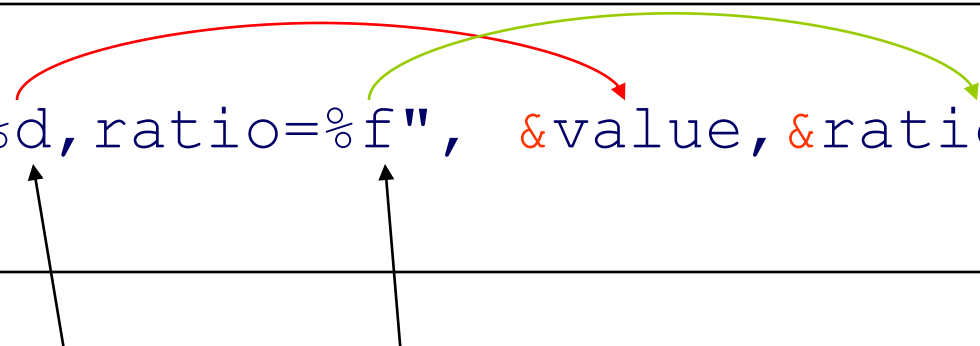


Input: value=10,ratio=15.9



scanf

```
scanf("value=%d,ratio=%f",&value,&ratio);
```



Input: value=10,ratio=15.9

```
scanf("%d%f",&value,&ratio);
```



Input: 10 15.9

Exercises

- What is the output of the following program

```
#include <stdio.h>
int main() {
    int percentage, x;
    char ch1, ch2;
    scanf("%d%c%c%d", &percentage, &ch1, &ch2, &x);
    printf("%d%% %c%c %d is %f\n",
           percentage, ch1, ch2, x, 1.0 * x *
           percentage / 100);
    return 0;
}
```

- what does this program do?
- What is the output if the input is 25 o f 60

Exercises Answer

```
#include <stdio.h>
int main() {
    int percentage, x;
    char ch1, ch2;
    scanf("%d%c%c%d", &percentage, &ch1, &ch2, &x);
}
```

- If the input is **25 o f 60**, the output is:

25% o -858993460 is -214748365.000000

- The %d conversion specifier expects the input text to be formatted as a decimal integer. If it isn't, the conversion fails and the character that caused the conversion to fail is left in the input stream.

Initialization

```
#include <stdio.h>
int main() {
    int percentage = 0, x = 0;
    char ch1 = 0, ch2 = 0;
    scanf("%d%c%c%d", &percentage, &ch1, &ch2, &x);
}
```

Therefore, it is recommended that all the variables should be **initialized**. It will be a good behavior for your **program debug**. If the variables are not initialized, the compiler will use **magic numbers** to initialize them.

https://www.asawicki.info/news_1292_magic_numbers_in_visual_c

What's More

```
#include <stdio.h>
int main() {
    int percentage=0, x=0;
    char ch1=0, ch2=0;
    scanf("%d %c %c %d", &percentage, &ch1, &ch2, &x);
}
```

Use spaces to separate the format string if you want to separate the inputs by using spaces.

Try input as 25 o f 60 again and you will find the output:

25% of 60 is 15.000000

https://www.asawicki.info/news_1292_magic_numbers_in_visual_c

Summary

- Make the output in a specific format
- How to read from keyboard