

Foundations of C Programming (Structured Programming) - Formatted I/O

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);
```



The diagram shows the `printf` function call with annotations. A red line underlines the format string `"The results are %d and %d\n"`. A red arrow points from the text `Format_string - included in a pair of " "` to this underline. Another red line underlines the variable `a`, with a red arrow pointing from the label `arg1` below it. A third red line underlines the variable `b`, with a red arrow pointing from the label `arg2` below it.

Format_string - included in a pair of " "

arg1

arg2

printf

- Format string contains (optional):
 - **literal text**: printed without any variation
 - **escape sequences**: preceded 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

```
int i = 10, j = 20;  
printf("The results are %d and %d\n", i, j);
```

The diagram illustrates the components of the `printf` format string `"The results are %d and %d\n"`. Annotations include:

- literal text**: Points to the string `"The results are"`.
- conversion specifiers and printf char**: Points to the `%d` conversion specifier.
- literal text**: Points to the string `and "`.
- conversion specifiers and printf char**: Points to the second `%d` conversion specifier.
- escaped sequences**: Points to the `\n` escape sequence.

Orange curved arrows indicate the mapping of arguments `i` and `j` to their respective `%d` specifiers.

The output is as follows:

```
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	long int

*More conversion specifiers are available. Refer to the Internet and reference books.

Escape Sequences

•	Sequence	Meaning
–	<code>\a</code>	Bell (alert)
–	<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 <i>hh</i>
–	<code>\0oo</code>	ASCII char specified by octal digits <i>oo</i>
–	<code>\1oo</code>	ASCII char specified by octal digits <i>1oo</i>

Class 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\n", c, d, f, f);
    printf("d = %d \tf = %d", d, f);
}
```


Class 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`

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



Format_string - included in a pair of ""

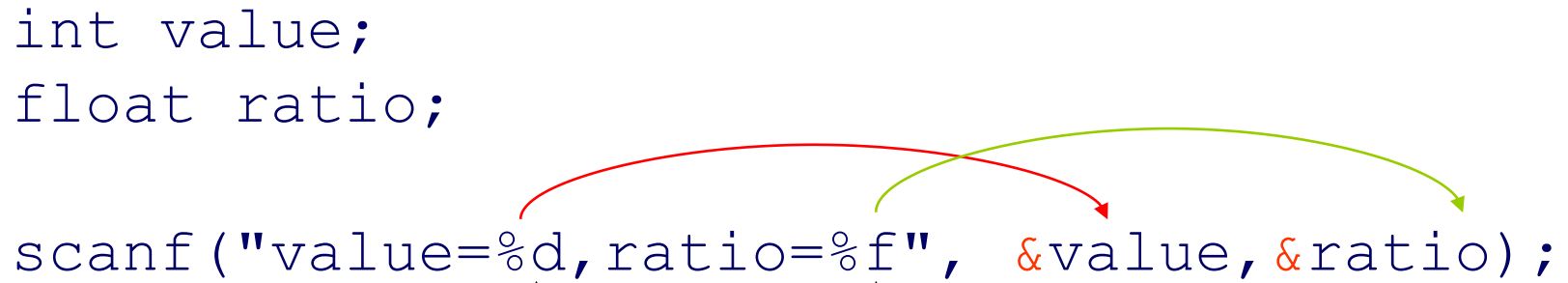
arg1

arg2

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

scanf

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

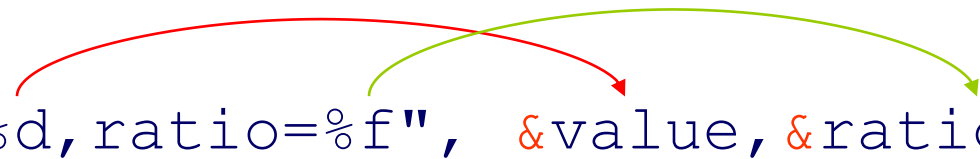


The diagram shows the variable declarations and the scanf call. A red curved arrow points from the format string "%d" to the variable &value. A green curved arrow points from the format string "%f" to the variable &ratio. Two black arrows point from the input string "value=10,ratio=15.9" to the format string "%d" and "%f" respectively.

Input: value=10,ratio=15.9

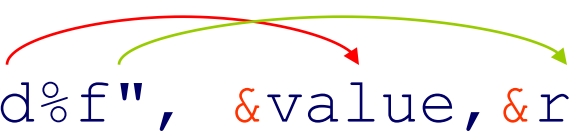
scanf

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



Input: value=10,ratio=15.9

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



%c and space (空格)

- Space in the input means reading next data. It is skipped if the next data is not of char type.

Input from keyboard: 10 20.1

<code>scanf ("%d%d", &a, &b);</code>	<code>a = 10, b = 20</code>
<code>scanf ("%d%c%d", &a, &c, &b);</code>	<code>a = 10, c = ' ', b = 20</code>
<code>scanf ("%d%d%c", &a, &b, &c);</code>	<code>a = 10, b = 20, c = '.'</code>
<code>scanf ("%d%f%c", &a, &f, &c);</code>	<code>a = 10, f = 20.1, c = '\n'</code>

Class Exercises

- What is the output of the following program (Percentage.c)

```
#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 is the output if the input is 25of60
- What is the output if the input is 25 o f 60

Attention: *printf* statement should be in one line except that the line is automatically broken in a code editor. You should not break the line by yourself.

Summary

- Make the output in a specific format
- Data can be inputted from keyboard in a sequence.