2

Introduction to C Programming

Objectives

In this chapter, you'll:

- Write simple C programs.
- Use simple input and output statements.
- Use the fundamental data types.
- Learn computer memory concepts.
- Use arithmetic operators.
- Learn the precedence of arithmetic operators.
- Write simple decision-making statements.
- Begin focusing on secure C programming practices.

Self-Review Exercises

2.1 Fill in the blanks in each of the following.

a) Every C program begins execution at the function _____.

ANS: `main`.

b) Every function's body begins with _____ and ends with _____.

ANS: left brace, right brace.

c) Every statement ends with a(n) _____.

ANS: semicolon.

d) The _____ standard library function displays information on the screen.

ANS: `printf`.

e) The escape sequence `\n` represents the _____ character, which causes the cursor to position to the beginning of the next line on the screen.

ANS: newline.

f) The _____ Standard Library function is used to obtain data from the keyboard.

ANS: `scanf`.

g) The conversion specifier _____ is used in a `scanf` format control string to indicate that an integer will be input and in a `printf` format control string to indicate that an integer will be output.

ANS: `%d`.

h) Whenever a new value is placed in a memory location, that value overrides the previous value in that location. This process is said to be _____.

ANS: destructive.

i) When a value is read from a memory location, the value in that location is preserved; this process is said to be _____.

ANS: nondestructive.

j) The _____ statement is used to make decisions.

ANS: `if`.

2.2 State whether each of the following is *true* or *false*. If *false*, explain why.

a) Function `printf` always begins printing at the beginning of a new line.

ANS: False. Function `printf` always begins printing where the cursor is positioned, and this may be anywhere on a line of the screen.

b) Comments cause the computer to print the text after the `//` on the screen when the program is executed.

ANS: False. Comments do not cause any action to be performed when the program is executed. They're used to document programs and improve their readability.

c) The escape sequence `\n` when used in a `printf` format control string causes the cursor to position to the beginning of the next line on the screen.

ANS: True.

d) All variables must be defined before they're used.

ANS: True.

e) All variables must be given a type when they're defined.

ANS: True.

f) C considers the variables `number` and `NuMbEr` to be identical.

ANS: False. C is case sensitive, so these variables are different.

g) Definitions can appear anywhere in the body of a function.

ANS: False. A variable's definition must appear before its first use in the code. In Microsoft Visual C++, variable definitions must appear immediately following the left brace that begins the body of `main`. Later in the book we'll discuss this in more depth as we encounter additional C features that can affect this issue.

h) All arguments following the format control string in a `printf` function must be preceded by an ampersand (`&`).

ANS: False. Arguments in a `printf` function ordinarily should not be preceded by an ampersand. Arguments following the format control string in a `scanf` function ordinarily should be preceded by an ampersand. We will discuss exceptions to these rules in Chapter 6 and Chapter 7.

i) The remainder operator (`%`) can be used only with integer operands.

ANS: True.

j) The arithmetic operators `*`, `/`, `%`, `+` and `-` all have the same level of precedence.

ANS: False. The operators `*`, `/` and `%` are on the same level of precedence, and the operators `+` and `-` are on a lower level of precedence.

k) A program that prints three lines of output must contain three `printf` statements.

ANS: False. A `printf` statement with multiple `\n` escape sequences can print several lines.

2.3 Write a single C statement to accomplish each of the following:

a) Define the variables `c`, `thisVariable`, `q76354` and `number` to be of type `int`.

ANS:
```
int c, thisVariable, q76354, number;
```

b) Prompt the user to enter an integer. End your prompting message with a colon (`:`) followed by a space and leave the cursor positioned after the space.

ANS:
```
printf( "Enter an integer: " );
```

c) Read an integer from the keyboard and store the value entered in integer variable `a`.

ANS:
```
scanf( "%d", &a );
```

d) If `number` is not equal to `7`, print `"The variable number is not equal to 7."`

ANS:
```
if ( number != 7 ) {
   printf( "The variable number is not equal to 7.\n" );
}
```

e) Print the message `"This is a C program."` on one line.

ANS:
```
printf( "This is a C program.\n" );
```

f) Print the message `"This is a C program."` on two lines so that the first line ends with `C`.

ANS:
```
printf( "This is a C\nprogram.\n" );
```

g) Print the message `"This is a C program."` with each word on a separate line.

ANS:
```
printf( "This\nis\na\nC\nprogram.\n" );
```

h) Print the message `"This is a C program."` with the words separated by tabs.

ANS:
```
printf( "This\tis\ta\tC\tprogram.\n" );
```

2.4 Write a statement (or comment) to accomplish each of the following:

a) State that a program will calculate the product of three integers.

ANS:
```
// Calculate the product of three integers
```

b) Prompt the user to enter three integers.

ANS:
```
printf( "Enter three integers: " );
```

c) Define the variables `x`, `y` and `z` to be of type `int`.

ANS:
```
int x, y, z;
```

d) Read three integers from the keyboard and store them in the variables `x`, `y` and `z`.

ANS:
```
scanf( "%d%d%d", &x, &y, &z );
```

e) Define the variable `result`, compute the product of the integers in the variables `x`, `y` and `z`, and use that product to initialize the variable `result`.

ANS:
```
int result = x * y * z;
```

f) Print `"The product is"` followed by the value of the integer variable `result`.

ANS:
```
printf( "The product is %d\n", result );
```

2.5 Using the statements you wrote in Exercise 2.4, write a complete program that calculates the product of three integers.

2.1 See below.

```
1  // Calculate the product of three integers
2  #include <stdio.h>
3
4  int main( void )
5  {
6     printf( "Enter three integers: " ); // prompt
7
8     int x, y, z; // declare variables
9     scanf( "%d%d%d", &x, &y, &z ); // read three integers
10
11    int result = x * y * z; // multiply values
12    printf( "The product is %d\n", result ); // display result
13 } // end function main
```

2.6 Identify and correct the errors in each of the following statements:

a) `printf( "The value is %d\n", &number );`

ANS: Error: `&number`.
Correction: Eliminate the `&`. We discuss exceptions to this later.

b) `scanf( "%d%d", &number1, number2 );`

ANS: Error: `number2` does not have an ampersand.
Correction: `number2` should be `&number2`. Later in the text we discuss exceptions to this.

c)
```
if ( c < 7 );{
   printf( "C is less than 7\n" );
}
```

ANS: Error: Semicolon after the right parenthesis of the condition in the `if` statement.
Correction: Remove the semicolon after the right parenthesis. [*Note:* The result of this error is that the `printf` statement will be executed whether or not the condition in the `if` statement is true. The semicolon after the right parenthesis is considered an empty statement—a statement that does nothing.]

d)
```
if ( c => 7 ) {
   printf( "C is greater than or equal to 7\n" );
}
```

ANS: Error: => is not an operator in C.
Correction: The relational operator => should be changed to >= (greater than or equal to).

Exercises

2.7 Identify and correct the errors in each of the following statements. (*Note:* There may be more than one error per statement.)

a) `scanf( "d", value );`
ANS: `scanf( "%d", &value );`
b) `printf( "The product of %d and %d is %d"\n, x, y );`
ANS: `printf( "The product of %d and %d is %d\n", x, y, x * y );`
c) `firstNumber + secondNumber = sumOfNumbers`
ANS: `sumOfNumbers = firstNumber + secondNumber;`

d) `if ( number => largest )`
`largest == number;`

ANS:
```
if ( number >= largest )
   largest = number;
```

e) `*/ Program to determine the largest of three integers /*`

ANS: `/* Program to determine the largest of three integers */`

f) `Scanf( "%d", anInteger );`

ANS: `scanf( "%d", &anInteger );`

g) `printf( "Remainder of %d divided by %d is\n", x, y, x % y );`

ANS: `printf( "Remainder of %d divided by %d is %d\n", x, y, x % y );`

h) `if ( x = y );`
`printf( %d is equal to %d\n", x, y );`

ANS:
```
if ( x == y )
   printf( "%d is equal to %d\n", x, y );
```

i) `print( "The sum is %d\n," x + y );`

ANS: `printf( "The sum is %d\n", x + y );`

j) `Printf( "The value you entered is: %d\n, &value );`

ANS: `printf( "The value you entered is: %d\n", value );`

2.8 Fill in the blanks in each of the following:

a) _____ are used to document a program and improve its readability.

ANS: comments.

b) The function used to display information on the screen is _____.

ANS: `printf`.

c) A C statement that makes a decision is _____.

ANS: `if`.

d) Calculations are normally performed by _____ statements.

ANS: assignment.

e) The _____ function inputs values from the keyboard.

ANS: `scanf`.

2.9 Write a single C statement or line that accomplishes each of the following:

a) Print the message "`Enter two numbers.`"

ANS: `printf( "Enter two numbers\n" );`

b) Assign the product of variables `b` and `c` to variable `a`.

ANS: `a = b * c;`

c) State that a program performs a sample payroll calculation (i.e., use text that helps to document a program).

ANS: `// Sample payroll calculation program`

d) Input three integer values from the keyboard and place these values in `int` variables `a`, `b` and `c`.

ANS: `scanf( "%d%d%d", &a, &b, &c );`

2.10 State which of the following are *true* and which are *false*. If *false*, explain your answer.

a) C operators are evaluated from left to right.

ANS: False. Some operators are evaluated left to right and others are evaluated from right to left depending on their associativity (see Appendix A).

b) The following are all valid variable names: `_under_bar_`, `m928134`, `t5`, `j7`, `her_sales`, `his_account_total`, `a`, `b`, `c`, `z`, `z2`.

ANS: True.

c) The statement `printf("a = 5;");` is a typical example of an assignment statement.

ANS: False. The statement prints `a = 5;` on the screen.

d) A valid arithmetic expression containing no parentheses is evaluated from left to right.
ANS: False. Multiplication, division, and modulus are all evaluated first from left to right, then addition and subtraction are evaluated from left to right.
e) The following are all invalid variable names: `3g`, `87`, `67h2`, `h22`, `2h`.
ANS: False. Only those beginning with a number are invalid.

2.11 Fill in the blanks in each of the following:
a) What arithmetic operations are on the same level of precedence as multiplication? _____.
ANS: division, modulus.
b) When parentheses are nested, which set of parentheses is evaluated first in an arithmetic expression? _____.
ANS: The innermost pair of parentheses.
c) A location in the computer's memory that may contain different values at various times throughout the execution of a program is called a _____.
ANS: variable.

2.12 What, if anything, prints when each of the following statements is performed? If nothing prints, then answer "Nothing." Assume `x = 2` and `y = 3`.

a) `printf( "%d", x );`
ANS: `2`
b) `printf( "%d", x + x );`
ANS: `4`
c) `printf( "x=" );`
ANS: `x=`
d) `printf( "x=%d", x );`
ANS: `x=2`
e) `printf( "%d = %d", x + y, y + x );`
ANS: `5 = 5`
f) `z = x + y;`
ANS: Nothing. Value of `x + y` is assigned to `z`.
g) `scanf( "%d%d", &x, &y );`
ANS: Nothing. Two integer values are read into the location of `x` and the location of `y`.
h) `// printf( "x + y = %d", x + y );`
ANS: Nothing. This is a comment.
i) `printf( "\n" );`
ANS: A newline character is printed, and the cursor is positioned at the beginning of the next line on the screen.

2.13 Which, if any, of the following C statements contain variables whose values are replaced?

a) `scanf( "%d%d%d%d%d", &b, &c, &d, &e, &f );`
b) `p = i + j + k + 7;`
c) `printf( "%s", Values are replaced." );`
d) `printf( "a = 5" );`
ANS: a and b.

2.14 Given the equation $y = ax^3 + 7$, which of the following, if any, are correct C statements for this equation?

a) `y = a * x * x * x + 7;`
b) `y = a * x * x * ( x + 7 );`
c) `y = ( a * x ) * x * ( x + 7 );`
d) `y = ( a * x ) * x * x + 7;`
e) `y = a * ( x * x * x ) + 7;`
f) `y = a * x * ( x * x + 7 );`
ANS: a, d, and e.

2.15 State the order of evaluation of the operators in each of the following C statements and show the value of x after each statement is performed.

a) `x = 7 + 3 * 6 / 2 - 1;`

ANS: * is first, / is second, + is third, - is fourth and = is last. Value of x is 15.

b) `x = 2 % 2 + 2 * 2 - 2 / 2;`

ANS: % is first, * is second, / is third, + is fourth, - is fifth and = is last. Value of x is 3.

c) `x = ( 3 * 9 * ( 3 + ( 9 * 3 / ( 3 ) ) ) );`

ANS: `5 6 4 2 3 1.` The = evaluates last. Value of x is 324.

2.17 *(Printing Values with `printf`)* Write a program that prints the numbers 1 to 4 on the same line. Write the program using the following methods.

a) Using one `printf` statement with no conversion specifiers.

ANS:
```
printf( "1 2 3 4\n\n" ); // part a
```

b) Using one `printf` statement with four conversion specifiers.

ANS:
```
printf( "%d %d %d %d\n\n", 1, 2, 3, 4 ); // part b
```

c) Using four `printf` statements.

ANS:
```
printf( "1 " ); // part c
printf( "2 " );
printf( "3 " );
printf( "4\n" );
```

2.22 What does the following code print?

```
printf( "*\n**\n***\n****\n*****\n" );
```

ANS:

```
*
**
***
****
*****
```

2.28 Distinguish between the terms fatal error and nonfatal error. Why might you prefer to experience a fatal error rather than a nonfatal error?

ANS: A fatal error causes the program to terminate prematurely. A nonfatal error occurs when the logic of the program is incorrect, and the program does not work properly. A fatal error is preferred for debugging purposes. A fatal error immediately lets you know there is a problem with the program, whereas a nonfatal error can be subtle and possibly go undetected.