C# Programming: From Problem Analysis to Program Design, 5th edition

Chapter 2

1. d. sampleValue

2. c. floating-point

3. c. string

4. e. all of the above

5. c. using

6. d. double does not require suffixing a numeric literal with a value such as ‘m’ or ‘f’.

7. b. string nameOfBook;

8. a. classes

9. b. 417

10. c. bool upperLimit;

11. a. places a value in memory that cannot be changed

12. d. result += 15;

13. c. 8

14. c. 28

15. d. all of the above

16. a. ans = value1 + value2 \* value3 – (value4 + 20 / 5 % 2) \* 7;

8 6 4 7 3 1 2 5

b. ans += value1-- \* 10;

3 2 1

c. ans = (((value1 + 7) – 6 \* value2) / 2);

5 1 3 2 4

d. ans = value1 + value2 / value3 \* value4--;

5 4 1 2 3

17. a. valid

b. invalid; # is an invalid character

c. invalid; has embedded spaces

d. valid

e. valid

18. a. int noOfCorrectResponses;

b. decimal amountOwed;

or double amountOwed;

c. string homeTown;

d. int examScore;

e. char finalGrade;

19. a. int noOfCorrectResponses = 0;

b. decimal amountOwed = 0m;

or double amountOwed = 0;

c. string homeTown = “Jacksonville”;

d. int examScore = 100;

e. char finalGrade = ‘A’;

20. a. x = 2; y = 7; z = 11

b. x = 10; y = 6; z = 10

c. x = 2; y = 6; z = 9

d. x = 2; y = 13; z = 10

21. a. x = 2.5; y = 7.9; z = 69

b. x = 19.666666666667; y = 5.9; z = 10

c. x = 2.5; y = 6.9; z = 4.16666666666667;

d. x = 2.5; y = 6.9; z = 4.225;

22. a. Result is $67.00

b. Number 1 is $3.00

c. 1 - $6.00

// The 1 is followed by 9 spaces, the hyphen, and then 5 more spaces before $6.00…negative width specifier left justifies and pads spaces to the right.

d. 1 result xyz 25.00

23. Variables and constants represent areas in memory where values of a particular data type can be stored. When you declare a variable or constant, you allocate memory for that data item in your program. Declaration of constants requires that you use the const keyword as part of the declaration. This forces the functionality of not allowing the value to be changed. Thus, variables can change; constants cannot.

24. Format specifier is useful if you want to control the alignment of items. Following the index ordinal before the colon, add a comma and then add a number representing how many positions the value should be use for display. The formatted data in the field is right-aligned if alignment number value is positive and left-aligned if alignment number value is negative.

25. a. needs a closing parenthesis for the Main method argument list

b. main should begin with an upper case character (Main)

c. const should be lowercase

d. declaration for y needs a type or replace the semicolon in the declaration for x with a comma.

e The ans argument to the Write( ) should not be enclosed in double quotes.

f. If using static System.Console; is added, the System.write should be changed to Write.

g. change the declaration of z to a double or suffix the value with an F or f.

i. the format string is reversed. It should be {0:f2}

j. instead of using the plus to concatenate the string with the result, use a comma to separate the arguments—because the format string is included inside the string argument.

k. need an extra closing brace

Style issues include

a. identifier for inches should be in uppercase, since it is a constant

b. start the name of the class identifier with an uppercase character

c. second and subsequent declarations should be on separate lines and indented under the previous declaration.

d. place a space before and a space after each of the arithmetic operators for readability

e. add internal documentation explaining the purpose of the application.

f. match beginning and ending curly braces

g. if a line spans to the next line, indent second and subsequent lines

Below is the working solution with the above-mentioned modifications.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This program demonstrates use of arithmetic operators.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

using System;

namespace Chapter2

{

class Converter

{

static void Main( )

{

const int INCHES = 12;

int x = 100,

y = 10;

float z = 22.45f;

double ans;

ans = INCHES + z \* x % y;

System.Console.Write("The result is {0:f2} ", ans);

}

}

}