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

Chapter 8

1. d. columns

2. d. int [ , ] temp = new temp [7, 2];

3. b. for(x = 0; x < 4; ++x) num [x, 2] = 100;

4. b. establishes the maximum number of rows as 4

5. c. the row index of the smallest element of array anArray

6. e. there is no limit

7. d. g

8. a. all have the same type

9. e. none of the above

10. b. x[1, 0] += 95;

11. d. 6

12. d. values[2,1] = 0;

13. d. anArray [2, 1] \*= 2;

14. d. replace the 800 amount with 810.

15. c. replace the 600 with 200.

16. b. display 1300.

17. c. a copy of the value in the element of the ArrayList

18. b. the address of the ArrayList

19. d. ToLower( )

20. c. Add( )

21. e. Capacity

22. d. ArrayList

23. a. Write(result(anArray, 3));

24. a. only be used for read-only access

25 c. Dequeue( )

26. a. string aValue;

sValue = sValue.ToLower();

aValue = sValue.Replace(" day", " DAY"); // Note space " day"

riteLine(aValue);

b. string [ ] stringArray = sValue.Split(' ');

c. stringArray[stringArray.Length - 1] = stringArray[stringArray.Length - 1].Remove(stringArray[stringArray.Length - 1].Length - 1, 1);

foreach (string val in stringArray)

WriteLine(val);

d. sValue = sValue.PadLeft(sValue.Length + 3, '\*');

sValue = sValue.PadRight(sValue.Length + 3, '\*');

e. sValue = sValue.Replace("first", "best");

27. a. 10

b. 11

c. 34

d. 3026

e. 2

28. a. foreach(int ar in cArray)

{

WriteLine(ar);

}

b.

for(int r = 0; r<cArray.GetLength(0); r ++)

{

for(int c = 0; c < cArray.GetLength(1); c ++)

cArray[r,c] += 5;

}

c. foreach(string ar in dArray)

{

WriteLine(ar);

}

d. No, foreach is used for read only access.

e. for(int p = 0; p < dArray.GetLength(0); p ++)

for(int r = 0; r < dArray.GetLength(1); r ++)

for(int c = 0; c < dArray.GetLength(2); c ++)

dArray[p,r,c] = “0”;

29. Create array declarations for the following problem specifications.

a. string [ , ] stateName = new string[5,3] {{"Miami", "Jacksonville", "Orlando"},

{"Knoxville", "Nashville", "Chattanooga"},

{"Louisville", "Lexington", "Bowling Green"},

{"Cincinnati", "Dayton", "Cleveland",}

{"Atlanta", “Macon”, “Savannah”}};

b. string [, ] name = new string [10, 2];

c. bool [, ] examKey = new bool[5, 15];

30. The Dictionary class has better performance than a HashTable for value types such as integers. Elements of HashTable are of type Object and require boxing and unboxing for storage and retrieval of value types.