Solution :Practice Exam

Lesson 1: Introduction to Programming

1. You need to gain a better understanding of the solution before writing the program. You decide to develop an algorithm that lists all necessary steps to perform an operation in the correct order. Any technique that you use should minimize complexity and ambiguity. Which of the following techniques should you use?
a) flowchart
b) decision table
c) C# program
d) A paragraph in English

Answer: a
Difficulty: Medium
Section Reference: Introducing Algorithms
A flowchart is a graphical representation of an algorithm that lists, in the correct order, all the necessary steps to perform the operation. A flowchart is simple to create and understand and is not ambiguous.

2. Which of the following languages is not considered a high-level programming language?
a) C#
b) Visual Basic
c) Common Intermediate Language
d) C++

Answer: c
Difficulty: Easy
Section Reference: Introducing C#
C#, Visual Basic, and C++ are all high-level programming languages. The Common Intermediate Language is low-level programming language used by the .NET Framework language compilers to create an executable file.

3. You are writing code for a business application by using C#. You write the following statement to declare an array:

```
int[] numbers = { 1, 2, 3, 4, 5 };
```

Now, you need to access the second item in this array (the number 2). Which of the following expression should you use?
a) `numbers[0]`
b) `numbers[1]`
c) `numbers[2]`
d) `numbers[3]`

Answer: b
Difficulty: Medium
Section Reference: Understanding Arrays

Any array item can be directly accessed by using an index. In the .NET Framework, array indexes are zero-based, meaning that to access the first element of an array, you use the index 1; to access the second element, you use the index 2; and so on. In the given case, as you need to access the second element, you use the expression `numbers[2]`.

4. You are developing a C# program. You write the following code:

```
int x = 10;
int y = ++x;
int z = y++;
```

What will be the value of the variable `z` after all the above statements are executed?
a) 10
b) 11
c) 12
d) 13

Answer: b
Difficulty: Hard
Section Reference: Understanding Operators
The way unary increment and decrement operators work when used as part of an assignment can affect the results. In particular, when the unary increment and decrement operators are used as prefixes, the current value of the identifier is returned before the increment or decrement. On the other hand, when used as a suffix, the value of the identifier is returned after the increment or decrement is complete.

When the first statement is executed, the value of `x` is 10. When the second statement is executed, the value of `x` and `y` are both 11. When the final statement is executed, the current value of `y` (11) is assigned to `z` before `y` is incremented by 1.

5. You are writing a method named `PrintReport` that doesn't return a value to the calling code. Which keyword should you use in your method declaration to indicate this fact?
a) `void`
b) `private`
c) `int`
d) `string`

Answer: a
Difficulty: Easy
Section Reference: Understanding Methods
When a method doesn't return a value back to the calling code, it is indicated by using the `void` keyword in the method declaration.

6. You need to provide complex multi-way branching in your C# program. You need to make sure that your code is easy to read and understand. Which of the following C# statements should you use?
a) `case`

b) `break`
c) `if-else`
d) `switch`

Answer: d
Difficulty: Medium
Section Reference: Understanding Decision Structures
The `switch` statement allows multi-way branching. In many cases, using a `switch` statement can simplify a complex combination of `if-else` statements.

7. You are writing a C# program that iterates through a collection such as arrays and lists. You need to make sure that you process each item in the collection once. You also need to ensure that your code is easy to read and debug. Which of the following C# statements provide the best solution for this requirement?
a) `while`
b) `for`
c) `foreach`
d) `do-while`

Answer: c
Difficulty: Easy
Section Reference: Understanding Repetition Structures
The `foreach` statement—an enhanced version of the `for` statement—is useful for iterating through collections such as arrays and lists. Using `foreach` statements eliminates the need for maintaining an index to the current item in the list. This improves code readability, makes debugging easier, and minimizes errors.

8. You are developing a C# program that needs to perform 5 iterations. You write the following code:

```
01:  int count = 0;
02:  while (count <= 5)
03:  {
04:      Console.WriteLine("The value of count = {0}", count);
05:      count++;
06:  }
```

When you run the program, you notice that the loop does not iterate five times. What should you do to make sure that the loop is executed exactly five times?
a) Change the code in line 01 to
```
int count = 1;
```
b) Change the code in line 02 to:
```
while (count == 5)
```
c) Change the code in line 02 to
```
while (count >= 5)
```
d) Change the code in line 05 to
```
++count;
```

Answer: a
Difficulty: Medium
Section Reference: Understanding the while loop
When the value of count starts at 1, the `while` loop executes once for each value of count 1, 2, 3, 4, and 5. When the `while` condition is changed to `(count == 5)` or to `(count >= 5)`, the loop will execute 0 times because the initial value of count is 0. Having `++count` as a standalone statement is same as `count++` and will not cause the results to vary.

9. You are developing a C# program. You write the following code line:

```
int x = 6 + 4 * 4 / 2 - 1;
```

What will be the value of the variable `x` after this statement is executed?
a) 19
b) 13
c) 20
d) 14

Answer: b
Difficulty: Medium
Section Reference: Understanding Operators
To evaluate this expression, you have to take into account operator precedence. The operators * and / have a higher precedence than + and –. You can also write this expression as

6 + ((4 * 4) / 2) – 1

This simplifies to, 6 + (8) – 1, resulting in 13.

10. You are writing a C# program that needs to manipulate very large integer values that may exceed 12 digits. The values can be positive or negative. Which data type should you use to store a variable like this?
a) `int`
b) `float`
c) `double`
d) `long`

Answer: d
Difficulty: Easy
Section Reference: Understanding Data Types
The `long` data type takes double the memory size of the `int` data type and can store integer values that exceed 12 digits. The `int` data type is relatively smaller. The `float` and `double` data types are more suited for storing floating-point numbers.

11. You have written a C# method that opens a database connection by using the `SqlConnect` object. The method retrieves some information from the database and then closes the connection.

You need to make sure that your code fails gracefully when there is a database error. To handle this situation, you wrap the database code in a `try-catch-finally` block. You use two `catch` blocks—one to catch the exceptions of type `SqlException` and the second to catch the exception of type `Exception`. Which of the following places should be the best choice for closing the `SqlConnection` object?
a) Inside the `try` block, before the first `catch` block
b) Inside the `catch` block that catches `SqlException` objects
c) Inside the `catch` block that catches `Exception` objects
d) Inside the `finally` block

Answer: d
Difficulty: Medium
Section Reference: Understanding Exception Handling
You need to make sure that the `SqlConnection` object is closed properly whether an exception occurred. The `finally` block is always executed and therefore is the best place to place such a code. The other answers are incorrect because code in these blocks can execute sometime but is not guaranteed to execute in every situation.

12. You are assisting your colleague in solving a compiler error that his code is throwing. Following is the problematic portion of his code:

```
try
{
    bool success = ApplyPicardoRotation(100, 0);
    // additional code lines here
}
catch(DivideByZeroException dbze)
{
    //exception handling code
}
catch(NotFiniteNumberException nfne)
{
    //exception handling code
}
catch(ArithmeticException ae)
{
    //exception handling code
}
catch(OverflowException oe)
{
    //exception handling code
}
```

To remove the compilation error, which of the following ways should you suggest to rearrange the code?
a)

```
try
{
    bool success = ApplyPicardoRotation(100, 0);
    // additional code lines here
}
catch(DivideByZeroException dbze)
```

```
{
    //exception handling code
}
catch(ArithmeticException ae)
{
    //exception handling code
}
catch(OverflowException oe)
{
    //exception handling code
}
```

b)
```
try
{
    bool success = ApplyPicardoRotation(100, 0);
    // additional code lines here
}
catch(DivideByZeroException dbze)
{
    //exception handling code
}
catch(Exception e)
{
    //exception handling code
}
catch(OverflowException oe)
{
    //exception handling code
}
```

c)
```
try
{
    bool success = ApplyPicardoRotation(100, 0);
    // additional code lines here
}
catch(DivideByZeroException dbze)
{
    //exception handling code
}
catch(NotFiniteNumberException nfne)
{
    //exception handling code
}
catch(OverflowException oe)
{
    //exception handling code
}
catch(ArithmeticException ae)
{
    //exception handling code
}
```

d)
```
try
{
    bool success = ApplyPicardoRotation(100, 0);
```

```
        // additional code lines here
    }
    catch(DivideByZeroException dbze)
    {
        //exception handling code
    }
    catch(NotFiniteNumberException nfne)
    {
        //exception handling code
    }
    catch(Exception e)
    {
        //exception handling code
    }
    catch(ArithmeticException ae)
    {
        //exception handling code
    }
```

Answer: c
Difficulty: Medium
Section Reference: Understanding Exception Handling
The correct answer arranges the `catch` statements from specific exceptions to the general exceptions. If you place the code to catch a general exception before the specific exception, the `catch` block for that specific statement will never get executed. The C# compiler detects this and flags this situation as error. The exceptions of type `Exception` are most general and hence should be placed in the last `catch` block. Next, the exception of type `ArithmeticException` is more general than `DivideByZeroException`, `OverflowException`, and `NotFiniteNumberException` and should be placed after these specific exceptions.

13. You are developing a C# program. You write a recursive method to calculate the factorial of a number. Which of the following code segment should you use to generate correct results?

a)
```
public static int Factorial(int n)
{
    if (n == 0)
    {
        return 1;
    }
    else
    {
        return n * Factorial(n - 1);
    }
}
```

b)
```
public static int Factorial(int n)
{
    if (n == 0)
    {
        return 1;
    }
    else
    {
        return (n – 1) * Factorial(n);
```

```
        }
    }
```

c)
```
    public static int Factorial(int n)
    {
        if (n == 0)
        {
            return n;
        }
        else
        {
            return Factorial(n - 1);
        }
    }
```

d)
```
    public static int Factorial(int n)
    {
            return n * Factorial(n - 1);
    }
```

Answer: a
Difficulty: Medium
Section Reference: Understanding Recursion
Answer a specifies the correctly formed base and the recursive case. Answer b is incorrect because the expression `(n – 1) * Factorial(n)` is not progressing towards the base case. Answer c is incorrect because the recursive case is not using the multiplication to get to the final value. Answer d is incorrect because the base case is missing and the method will never terminate.

14. You are developing a C# program. You write the following code:

```
01:  int count = 0;
02:  while (count < 5)
03:  {
04:      if (count == 3)
05:         break;
06:      count++;
07:  }
```

How many times will the control enter the `while` loop?
a) 5
b) 4
c) 3
d) 2

Answer: b
Difficulty: Medium
Section Reference: Understanding Repetition Structures

You enter the loop each time when the value of count is 0, 1, 2, and 3. So the correct answer is 4. When the value of count reaches 3, the `break` statement is executed to terminate the loop and transfer the control outside the loop.

15. You are developing a C# program. You write the following code:

```
int i = 6;
do
{
    if (i == 3)
       break;
    Console.WriteLine("The value of i = {0}", i);
    i++;
    }
while (i <= 5);
```

How many times will the control enter the `while` loop?
a) 0
b) 1
c) 2
d) 3

Answer: b
Difficulty: Medium
Section Reference: Understanding Repetition Structures
The control will enter the loop only once. At the end of the first iteration, the condition will fail (because the number 6 is greater than the number 5) and the loop will terminate.

16. You are writing a C# program and need to select an appropriate repetition structure for your requirement. You need to make sure that the test for the termination condition is performed at the bottom of the loop rather than at the top. Which repletion structure should you use?
a) The `while` statement
b) The `for` statement
c) The `foreach` statement
d) The `do-while` statement

Answer: d
Difficulty: Easy
Section Reference: Understanding Repetition Structures
The `do-while` statement performs the test for the termination condition at the bottom of the loop. All other repletion structure performs the test at the top of the loop.

17. You are writing a C# program. You write the following method:

```
public static void TestSwitch(int op1, int op2, char opr)
{
    int result;
    switch (opr)
    {
```

```
            case '+':
                result = op1 + op2;
            case '-':
                result = op1 - op2;
            case '*':
                result = op1 * op2;
            case '/':
                result = op1 / op2;
            default:
                Console.WriteLine("Unknown Operator");
                return;
        }
        Console.WriteLine("Result: {0}", result);
        return;
    }
```

However, when you compile this code, you get the following error message:

Control cannot fall through from one case label to another

How should you modify the code to make sure that it compiles successfully?

a) After each case, add the following code line:
`break;`

b) After each case, add the following code line:
`continue;`

c) After each case, add the following code line:
`goto default;`

d) After each case, add the following code line:
`return;`

Answer: a
Difficulty: Easy
Section Reference: Understanding Decision Structures
In this code example, no `break` statement occurs after each case. The `break` statement terminates the `switch` statement and transfers control to the next statement outside the `switch` block. The `continue` statement is not the right answer because no enclosing loop is included here. The `goto` and `return` statements are not correct because they will change the program's intended output.

18. You are developing an algorithm for a retail Web site. You need to calculate discounts on certain items based on the quantity purchased. You develop the following decision table to calculate the discount:

Quantity < 10	Y	N	N	N
Quantity < 50	Y	Y	N	N
Quantity < 100	Y	Y	Y	N
Discount	5%	10%	15%	20%

If a customer buys 50 units of an item, what discount will be applicable to the purchase?

a) 5 percent
b) 10 percent
c) 15 percent
d) 20 percent

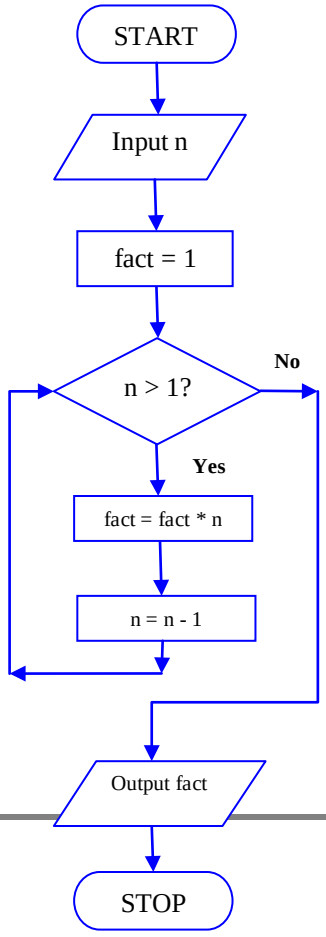
Answer: c
Difficulty: Medium
Section Reference: Introducing Decision Tables
When a customer buys 50, the conditions Quantity < 10 and Quantity < 50 are both false but the condition Quantity < 100 is true. So, when you look at the column N, N, and Y, the corresponding value of the discount is 15%.

19. You are developing an algorithm before you write the C# program. You need to perform some calculations on a number. You develop the following flowchart for the calculation:

If the input value of n is 5, what is the output value of the variable fact according to this flowchart?

a) 720
b) 120
c) 24
d) 6

Answer: b



Difficulty: Easy
Section Reference: Introducing Flowcharts
Here, you enter the loop once each time when the value of n is 5, 4, 3, and 2. So the final result will be 5 * 4 * 3 * 2 = 120.

20. You are writing a C# program that needs to iterate a fixed number of times. You need to make sure that your code is easy to understand and maintain even when the loop body contains complex code. Which of the following C# statements provide the best solution for this requirement?
a) `while`
b) `for`
c) `foreach`
d) `do-while`

Answer: b
Difficulty: Easy
Section Reference: Understanding Repetition Structures
The `for` loop is ideal for creating iterations that must execute a specified number of times. The `for` loop combines the three elements of iteration—the initialization expression, the termination condition expression, and the counting expression—into a more readable code by placing them outside the loop body. This improves code readability, makes debugging easier and minimizes errors.