

# IT-PRUEFUNG

Prüfungshilfen für IT Zertifizierungen



<http://www.it-pruefung.com>

Wir bieten Ihnen einen kostenlosen einjährigen Upgrade Service an

**Exam** : **70-483**

**Title** : Programming in C#

**Vendor** : Microsoft

**Version** : DEMO

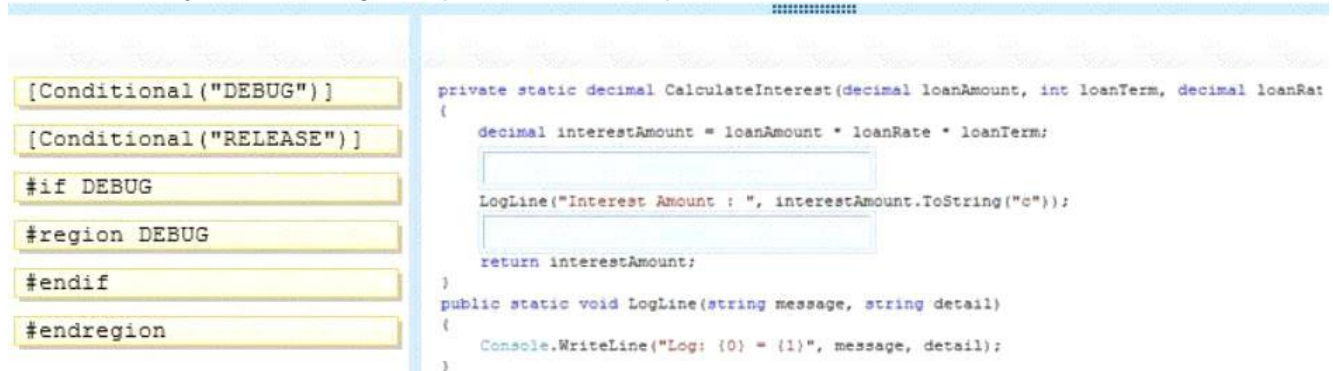
**NO.1** You are testing an application. The application includes methods named CalculateInterest and LogLine. The CalculateInterest() method calculates loan interest. The LogLine() method sends diagnostic messages to a console window.

You have the following requirements:

- \* The CalculateInterest() method must run for all build configurations.
- \* The LogLine() method must be called only for debug builds.

You need to ensure that the methods run correctly.

How should you complete the relevant code? (To answer, drag the appropriate code segments to the correct locations in the answer area. Each code segment may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.)



The code snippet shows the CalculateInterest method and the LogLine method. The LogLine method is currently commented out. The drag-and-drop interface shows several code segments that can be placed into the code snippet.

Available code segments:

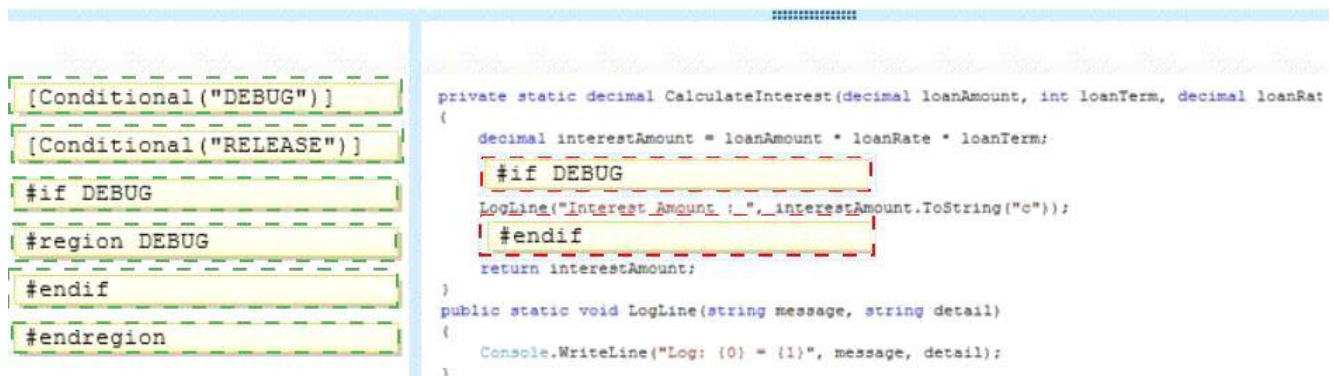
- [Conditional("DEBUG")]
- [Conditional("RELEASE")]
- #if DEBUG
- #region DEBUG
- #endif
- #endregion

Code snippet:

```
private static decimal CalculateInterest(decimal loanAmount, int loanTerm, decimal loanRate)
{
    decimal interestAmount = loanAmount * loanRate * loanTerm;
    // LogLine("Interest Amount : ", interestAmount.ToString("c"));
    return interestAmount;
}

public static void LogLine(string message, string detail)
{
    Console.WriteLine("Log: {0} = {1}", message, detail);
}
```

**Answer:**



The code snippet shows the CalculateInterest method and the LogLine method. The LogLine method is now uncommented and wrapped in #if DEBUG and #endif directives. The drag-and-drop interface shows the segments that were placed into the code snippet.

Available code segments:

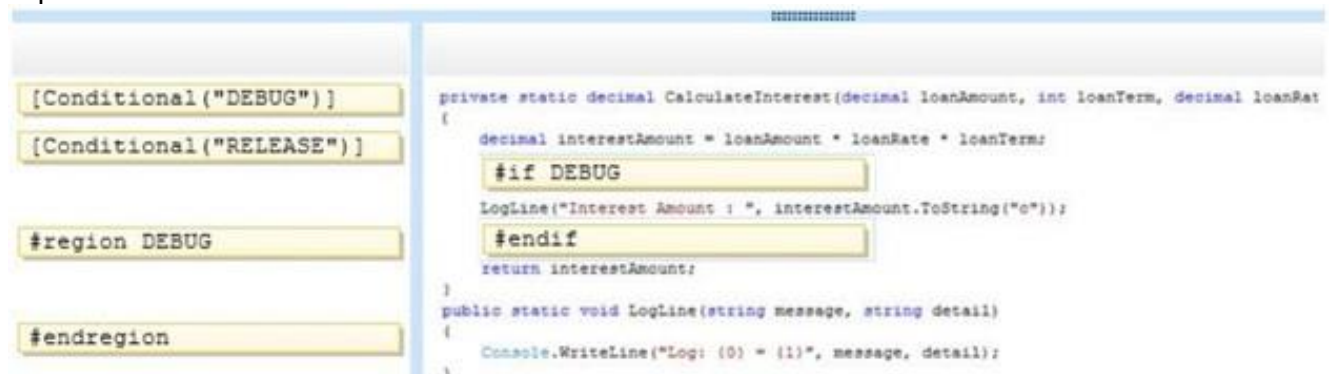
- [Conditional("DEBUG")]
- [Conditional("RELEASE")]
- #if DEBUG
- #region DEBUG
- #endif
- #endregion

Code snippet:

```
private static decimal CalculateInterest(decimal loanAmount, int loanTerm, decimal loanRate)
{
    decimal interestAmount = loanAmount * loanRate * loanTerm;
    #if DEBUG
    LogLine("Interest Amount : ", interestAmount.ToString("c"));
    #endif
    return interestAmount;
}

public static void LogLine(string message, string detail)
{
    Console.WriteLine("Log: {0} = {1}", message, detail);
}
```

**Explanation**



The code snippet shows the CalculateInterest method and the LogLine method. The LogLine method is now uncommented and wrapped in #if DEBUG and #endif directives. The drag-and-drop interface shows the segments that were placed into the code snippet.

Available code segments:

- [Conditional("DEBUG")]
- [Conditional("RELEASE")]
- #if DEBUG
- #region DEBUG
- #endif
- #endregion

Code snippet:

```
private static decimal CalculateInterest(decimal loanAmount, int loanTerm, decimal loanRate)
{
    decimal interestAmount = loanAmount * loanRate * loanTerm;
    #if DEBUG
    LogLine("Interest Amount : ", interestAmount.ToString("c"));
    #endif
    return interestAmount;
}

public static void LogLine(string message, string detail)
{
    Console.WriteLine("Log: {0} = {1}", message, detail);
}
```

When the C# compiler encounters a directive, followed eventually by an #endif directive, it will compile the code between the directives only if the specified symbol is defined. Unlike C and C++, you cannot assign a numeric value to a symbol; the #if statement in C# is Boolean and only tests whether the symbol has been defined or not. For example,

```
#define DEBUG
#if DEBUG
```

```
Console.WriteLine("Debug version");  
#endif
```

Reference: <http://stackoverflow.com/questions/2104099/c-sharp-if-then-directives-for-debug-vs-release>

**NO.2** You are developing an application that will transmit large amounts of data between a client computer and a server. You need to ensure the validity of the data by using a cryptographic hashing algorithm. Which algorithm should you use?

- A. RNGCryptoServiceProvider
- B. HMACSHA512
- C. Rfc2898DeriveBytes
- D. ECDsa

**Answer:** B

Explanation

The HMACSHA512 class computes a Hash-based Message Authentication Code (HMAC) using the SHA512 hash function.

Reference:

[https://msdn.microsoft.com/en-us/library/system.security.cryptography.hmacsha512\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.security.cryptography.hmacsha512(v=vs.110).aspx)

**NO.3** An application serializes and deserializes XML from streams. The XML streams are in the following format:

```
<Name xmlns="http://www.contoso.com/2012/06">  
  <LastName>Jones</LastName>  
  <FirstName>David</FirstName>  
</Name>
```

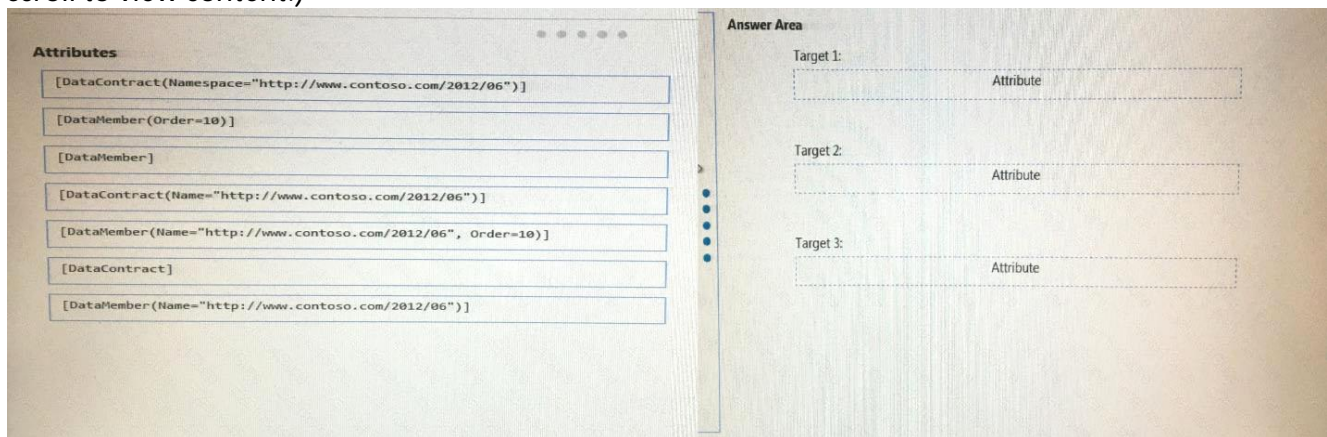
The application reads the XML streams by using a DataContractSerializer object that is declared by the following code segment:

```
Target 1  
class Name  
{  
    Target 2  
    public string FirstName { get; set; }  
    Target 3  
    public string LastName { get; set; }  
}
```

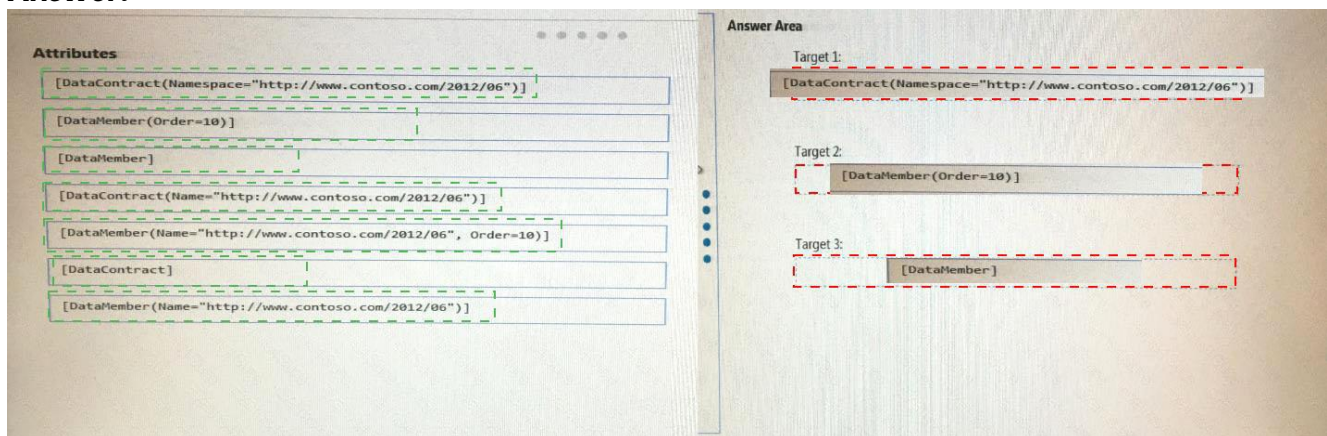
Which attributes should you include in Target 1 if Target 2 and Target 3 to complete the code? (To answer, drag the appropriate attributes to the correct targets in the answer area. Each attribute may



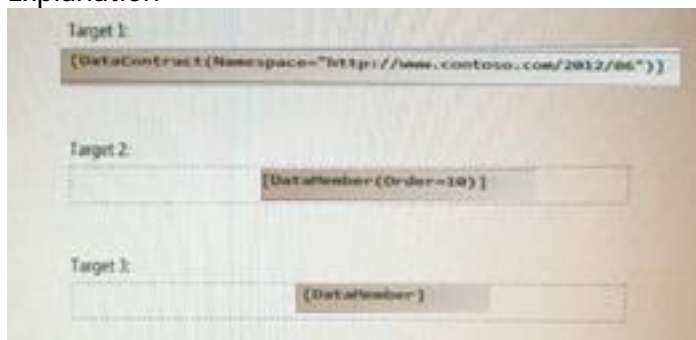
be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.)



**Answer:**



**Explanation**



**NO.4** You are developing an application. The application calls a method that returns an array of integers named `employeeIds`. You define an integer variable named `employeeIdToRemove` and assign a value to it. You declare an array named `filteredEmployeeIds`.

You have the following requirements:

- \* Remove duplicate integers from the `employeeIds` array.
- \* Sort the array in order from the highest value to the lowest value.
- \* Remove the integer value stored in the `employeeIdToRemove` variable from the `employeeIds` array.

You need to create a LINQ query to meet the requirements.

Which code segment should you use?

- ☐ A. `int[] filteredEmployeeIds = employeeIds.Where(value => value != employeeIdToRemove).OrderBy(x => x).ToArray();`
- ☐ B. `int[] filteredEmployeeIds = employeeIds.Where(value => value != employeeIdToRemove).OrderByDescending(x => x).ToArray();`
- ☐ C. `int[] filteredEmployeeIds = employeeIds.Distinct().Where(value => value != employeeIdToRemove).OrderByDescending(x => x).ToArray();`
- ☐ D. `int[] filteredEmployeeIds = employeeIds.Distinct().OrderByDescending(x => x).ToArray();`

A. Option A

B. Option D

C. Option B

D. Option C

**Answer:** D

Explanation

The Distinct keyword avoids duplicates, and OrderByDescending provides the proper ordering from highest to lowest.

**NO.5** You develop an application that displays information from log files when errors occur. The application will prompt the user to create an error report that sends details about the error and the session to the administrator.

When a user opens a log file by using the application, the application throws an exception and closes. The application must preserve the original stack trace information when an exception occurs during this process.

You need to implement the method that reads the log files.

How should you complete the relevant code? (To answer, drag the appropriate code segments to the correct locations in the answer area. Each code segment may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.)

```
using (StringReader sr = new StringReader("log.txt"))  
using (StreamReader sr = new StreamReader("log.txt"))  
throw new FileNotFoundException();  
throw;
```

```
{  
    try  
    {  
        string line;  
        while ((line = sr.ReadLine()) != null)  
        {  
            Console.WriteLine(line);  
        }  
    }  
    catch (FileNotFoundException e)  
    {  
        Console.Write(e.ToString());  
    }  
}
```

**Answer:**

```
using (StringReader sr = new StringReader("log.txt"))  
using (StreamReader sr = new StreamReader("log.txt"))  
throw new FileNotFoundException();  
throw;
```

```
using (StreamReader sr = new StreamReader("log.txt"))
```

```
{  
    try  
    {  
        string line;  
        while ((line = sr.ReadLine()) != null)  
        {  
            Console.WriteLine(line);  
        }  
    }  
    catch (FileNotFoundException e)  
    {  
        Console.Write(e.ToString());  
        throw;  
    }  
}
```

Explanation



```
using (StringReader sr = new StringReader("log.txt"))
```

```
throw new FileNotFoundException();
```

```
.....
```

```
using (StreamReader sr = new StreamReader("log.txt"))
```

```
{
    try
    {
        string line;
        while ((line = sr.ReadLine()) != null)
        {
            Console.WriteLine(line);
        }
    }
    catch (FileNotFoundException e)
    {
        Console.Write(e.ToString());
        throw;
    }
}
```

StreamReader - Implements a TextReader that reads characters from a byte stream in a particular encoding.

Reference: [http://msdn.microsoft.com/en-us/library/system.io.streamreader\(v=vs.110\).aspx](http://msdn.microsoft.com/en-us/library/system.io.streamreader(v=vs.110).aspx) Once an exception is thrown, part of the information it carries is the stack trace. The stack trace is a list of the method call hierarchy that starts with the method that throws the exception and ends with the method that catches the exception. If an exception is re-thrown by specifying the exception in the throw statement, the stack trace is restarted at the current method and the list of method calls between the original method that threw the exception and the current method is lost. To keep the original stack trace information with the exception, use the throw statement without specifying the exception.

Reference: [http://msdn.microsoft.com/en-us/library/ms182363\(v=vs.110\).aspx](http://msdn.microsoft.com/en-us/library/ms182363(v=vs.110).aspx) Incorrect:

StringReader - Implements a TextReader that reads from a string.

Reference: [http://msdn.microsoft.com/en-us/library/system.io.stringreader\(v=vs.110\).aspx](http://msdn.microsoft.com/en-us/library/system.io.stringreader(v=vs.110).aspx)

**NO.6** You are developing a C# application. The application includes a class named Rate. The following code segment implements the Rate class:

```
public class Rate
{
    public string Category { get; set; }
    public DateTime Date { get; set; }
    public decimal Value { get; set; }
}
```

You define a collection of rates named rateCollection by using the following code segment:

```
Collection<Rate> rateCollection = new Collection<Rate>();
```

The application receives an XML file that contains rate information in the following format:

```
<?xml version="1.0" encoding="utf-8" ?>
<RateSheet>
    <rate category="buyout" date="2012-03-22">
        <value>0.0375</value>
    </rate>
    <rate category="fixed" date="2012-03-23">
        <value>0.0475</value>
    </rate>
</RateSheet>
```

You need to parse the XML file and populate the rateCollection collection with Rate objects.

How should you complete the relevant code? (To answer, drag the appropriate code segments to the correct locations in the answer area. Each code segment may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.)

<pre>while(reader.ReadToFollowing while(reader.ReadToFollowing("rate")) reader.MoveToElement(); reader.MoveToFirstAttribute(); reader.MoveToContent(); reader.MoveToNextAttribute(); reader.ReadToFollowing("value");</pre>	<pre>using (XmlReader reader = XmlReader.Create(new StringReader(rateXML))) {     Rate rate = new Rate();     rate.Category = reader.Value;     DateTime rateDate;     if (DateTime.TryParse(reader.Value, out rateDate))     {         rate.Date = rateDate;     }     decimal value;     if (decimal.TryParse(reader.ReadElementContentAsString(), out value))     {         rate.Value = value;     }     rateCollection.Add(rate); }</pre>
---	--

**Answer:**

```

while(reader.ReadToFollowing("rate"))
{
    reader.MoveToElement();
    reader.MoveToFirstAttribute();
    reader.MoveToContent();
    reader.MoveToNextAttribute();
    reader.ReadToFollowing("value");
}

using (XmlReader reader = XmlReader.Create(new StringReader(rateXML)))
{
    while(reader.ReadToFollowing("rate"))
    {
        Rate rate = new Rate();
        reader.MoveToFirstAttribute();
        rate.Category = reader.Value;
        reader.MoveToNextAttribute();
        DateTime rateDate;
        if (DateTime.TryParse(reader.Value, out rateDate))
        {
            rate.Date = rateDate;
        }
        reader.ReadToFollowing("value");
        decimal value;
        if (decimal.TryParse(reader.ReadElementContentAsString(), out value))
        {
            rate.Value = value;
        }
        rateCollection.Add(rate);
    }
}

```

### Explanation

```

using (XmlReader reader = XmlReader.Create(new StringReader(rateXML)))
{
    while(reader.ReadToFollowing("rate"))
    {
        Rate rate = new Rate();
        reader.MoveToFirstAttribute();
        rate.Category = reader.Value;
        reader.MoveToNextAttribute();
        DateTime rateDate;
        if (DateTime.TryParse(reader.Value, out rateDate))
        {
            rate.Date = rateDate;
        }
        reader.ReadToFollowing("value");
        decimal value;
        if (decimal.TryParse(reader.ReadElementContentAsString(), out value))
        {
            rate.Value = value;
        }
        rateCollection.Add(rate);
    }
}

```

### Explanation

\* Target 1: The element name is rate not Ratesheet.

The Xmlreader readToFollowing reads until the named element is found.

\* Target 2:

The following example gets the value of the first attribute.

reader.ReadToFollowing("book");

reader.MoveToFirstAttribute();

string genre = reader.Value;

Console.WriteLine("The genre value: " + genre);

\* Target 3, Target 4:

The following example displays all attributes on the current node.

C#VB

if (reader.HasAttributes) {

Console.WriteLine("Attributes of <" + reader.Name + ">");

```

while (reader.MoveToNextAttribute()) {
    Console.WriteLine("{0}={1}", reader.Name, reader.Value);
}
// Move the reader back to the element node.
reader.MoveToElement();
}

```

The `XmlReader.MoveToElement` method moves to the element that contains the current attribute node.

Reference: `XmlReader` Methods

[https://msdn.microsoft.com/en-us/library/System.Xml.XmlReader\\_methods\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/System.Xml.XmlReader_methods(v=vs.110).aspx)

**NO.7** You are developing an application that includes the following code segment. (Line numbers are included for reference only.)

```

01 class Animal
02 {
03     public string Color { get; set; }
04     public string Name { get; set; }
05 }
06 private static IEnumerable<Animal> GetAnimals(string sqlConnectionString)
07 {
08     var animals = new List<Animal>();
09     SqlConnection sqlConnection = new SqlConnection(sqlConnectionString);
10     using (sqlConnection)
11     {
12         SqlCommand sqlCommand = new SqlCommand("SELECT Name, ColorName FROM Animals", sqlConnection);
13
14         using (SqlDataReader sqlDataReader = sqlCommand.ExecuteReader())
15         {
16
17             {
18                 var animal = new Animal();
19                 animal.Name = (string)sqlDataReader["Name"];
20                 animal.Color = (string)sqlDataReader["ColorName"];
21                 animals.Add(animal);
22             }
23         }
24     }
25     return customers;
26 }

```

The `GetAnimals()` method must meet the following requirements:

- \* Connect to a Microsoft SQL Server database.
- \* Create `Animal` objects and populate them with data from the database.
- \* Return a sequence of populated `Animal` objects.

You need to meet the requirements.

Which two actions should you perform? (Each correct answer presents part of the solution. Choose two.)

- A.** Insert the following code segment at line 13:  
`sqlConnection.Open();`
- B.** Insert the following code segment at line 16:  
`while(sqlDataReader.Read())`
- C.** Insert the following code segment at line 13:  
`sqlConnection.BeginTransaction();`
- D.** Insert the following code segment at line 16:  
`while(sqlDataReader.GetValues())`
- E.** Insert the following code segment at line 16:  
`while(sqlDataReader.NextResult())`



**Answer:** A,B

Explanation

B: SqlConnection.Open - Opens a database connection with the property settings specified by the ConnectionString.

Reference: <http://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqlconnection.open.aspx>

D: SqlDataReader.Read - Advances the SqlDataReader to the next record. Reference:

<http://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqldatareader.read.aspx>

**NO.8** You are developing an application that uses the Microsoft ADO.NET Entity Framework to retrieve order information from a Microsoft SQL Server database. The application includes the following code. (Line numbers are included for reference only.)

```

01 public DateTime? OrderDate;
02 IQueryable<Order> LookupOrdersForYear(int year)
03 {
04     using (var context = new NorthwindEntities())
05     {
06         var orders =
07             from order in context.Orders
08
09             select order;
10         return orders.ToList().AsQueryable();
11     }
12 }

```

The application must meet the following requirements:

- \* Return only orders that have an OrderDate value other than null.
- \* Return only orders that were placed in the year specified in the year parameter.

You need to ensure that the application meets the requirements. Which code segment should you insert at line

08?

- A. `where order.OrderDate.Value.Year == year`
- B. `where order.OrderDate.HasValue && order.OrderDate.Value.Year == year`
- C. `where order.OrderDate.Value != null && order.OrderDate.Value.Year >= year`
- D. `where order.OrderDate.Value == null && order.OrderDate.Value.Year == year`

- A. Option C
- B. Option D
- C. Option A
- D. Option B

**Answer:** D

**NO.9** You are developing an application that includes a class named Order. The application will store a collection of Order objects.

The collection must meet the following requirements:

- \* Internally store a key and a value for each collection item.
- \* Provide objects to iterators in ascending order based on the key.
- \* Ensure that items are accessible by zero-based index or by key.

You need to use a collection type that meets the requirements.

Which collection type should you use?

- A. SortedList
- B. HashTable
- C. Array
- D. LinkedList
- E. Queue

**Answer:** A

Explanation

SortedList<TKey, TValue> - Represents a collection of key/value pairs that are sorted by key based on the associated IComparer<T> implementation.

<http://msdn.microsoft.com/en-us/library/>

**NO.10** You are developing an application.

The application contains the following code segment (line numbers are included for reference only):

```
01 ArrayList array1 = new ArrayList();
02 int var1 = 10;
03 int var2;
04 array1.Add(var1);
05 var2 = array1[0];
```

When you run the code, you receive the following error message: "Cannot implicitly convert type 'object' to

'int'. An explicit conversion exists (are you missing a cast?)."

You need to ensure that the code can be compiled.

Which code should you use to replace line 05?

- A. var2 = (int) array1 [0];
- B. var2 = ((List<int>)array1) [0];
- C. var2 = array1[0] is int;
- D. var2 = array1[0].Equals(typeof(int));

**Answer:** A

**NO.11** You are creating a class library that will be used in a web application.

You need to ensure that the class library assembly is strongly named.

What should you do?

- A. Use the gacutil.exe command-line tool.
- B. Use the EdmGen.exe command-line tool.
- C. Use assembly attributes.
- D. Set the configuration mode to Release when building the application.

**Answer: C****NO.12** You are building a data access layer in an application that contains the following code:

```

public static Object GetTypeDefault(DbType dbDataType)
{
    switch (dbDataType)
    {
        case DbType.Boolean:
            return false;
        case DbType.DateTime:
            return DateTime.MinValue;
        case DbType.Decimal:
            return 0m;
        case DbType.Int32:
            return 0;
        case DbType.String:
            return String.Empty;
        default:
            return null;
    }
}

```

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

	Yes	No
If dbDataType is DateTime, today's date is returned.	<input type="radio"/>	<input type="radio"/>
If dbDatatype is Int64, Null is returned.	<input type="radio"/>	<input type="radio"/>
If dbDatatype is Double, 0 is returned.	<input type="radio"/>	<input type="radio"/>

**Answer:**

	Yes	No
If dbDataType is DateTime, today's date is returned.	<input type="radio"/>	<input checked="" type="checkbox"/>
If dbDatatype is Int64, Null is returned.	<input type="radio"/>	<input type="checkbox"/>
If dbDatatype is Double, 0 is returned.	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Explanation

	Yes	No
If dbDataType is DateTime, today's date is returned.	<input type="radio"/>	<input checked="" type="radio"/>
If dbDatatype is Int64, Null is returned.	<input checked="" type="radio"/>	<input type="radio"/>
If dbDatatype is Double, 0 is returned.	<input type="radio"/>	<input checked="" type="radio"/>