Potential Exam Questions

Prerelease Questions

**Example of a static method**

Public Shared Function NewRandom(ByVal lowerbound As Integer, ByVal upperbound As Integer)

**Example of composite aggregation**

Dim numbers As New NumberMachine

**Example of Inheritance**

Inherits BingoCard

**State an advantage of using named constants for constant values**

* Improves readability of code
* Reduces likelihood of errors

**There is a variable called – in -. There is also a variable called – in --. Explain why these 2 different variables can have the same identifier.**

* The scope of the 2 variables is different
* Both are local variables in different subroutine

**Describe what the selection structure inside repetition structure of AssignRowPlaces Subroutine does**

This selection structure checks if there are any duplicate numbers. Then, match is sent to true and so numberCount reduces by one so that another number can be added

**State the name of an identifier for a user-defined subroutine that has 2 parameters which are both integers**

FullHouse

**State the name of an identifier for a user defined subroutine whose action is to produce output to the screen**

DisplayCard

**Look at the repetition structure in the --. Describe the circumstances under which this structure in the skeleton program will stop repeating**

**Look at the --. Why has a for loop been chosen for the repetition structure?**

* There are known number of iterations

**The for loop repeats --- times. Why has a named constant been used instead of the numeric value?**

* Makes program code easier to understand
* Makes it easier to update program

**Describe the difference between the data stored in a binary file and in a text file?**

In a text file, all the data is stored as strings. In a binary file, data is stored using a variety of different data types.

**State reasons why subroutine should, ideally, not use global variables**

* Easier reuse of routines in other programs
* Help us make the program code more understandable
* Ensures the routine is self-contained

**Explain how data is shared between the separate subroutines**

* Parameters are passed
* Values are returned from a subroutine
* Constants are available to all subroutines

**Explain the difference between a protected attribute and a private attribute.**

* A protected attribute can be accessed within its class and by derived class instances / subclasses
* A private attribute can only be accessed within its class

**Explain what is meant by a structured programming approach**

* Decomposition of a problem (use of top - down approach )
* Use of block structures
* Structured programming makes use of control structures
* Avoidance of use of go to statements

**Why is it good practice to use local variables?**

* Makes subroutine self contained
* Releases storage when subroutine terminates
* Able to test subroutine independently
* Local variables cannot be accidently altered by a subroutine call from the subroutine

Programming Questions

**Aligning the bingo card**

Public Sub Displaycard() 'prints the card

For x = 0 To 2

For y = 0 To 8

If y > 0 Then

If numbers(x, y) > 9 Then

Console.Write(numbers(x, y) & ",")

Else

Console.Write(" " & numbers(x, y) & ",")

End If

Else

Console.Write(numbers(x, y) & ",")

End If

Next y

Console.WriteLine()

Next x

End Sub

**Loop through all numbers**

In NumberMachine class

Private numberOrder(89) As Integer

Private back As Integer = 89

For x = 1 To 90

numberOrder(x - 1) = x ' sets numberOrder from 1 to 99

Next x

For x = 1 To 1000

num1 = repo.NewRandom(0, 89) ' random number assigned to num1

num2 = repo.NewRandom(0, 89) ' random number assigned to num2

' switches numberOrder(num1) with numberOrder(num2)

temp = numberOrder(num1) ' integer in numberOrder with position of num1 is assigned to temp

numberOrder(num1) = numberOrder(num2) ' integer in numberOrder with position of num2 is assigned to numberOrder(num1)

**Check for corners in the Bingo Bonus game**

Private Function Corners(ByVal calledNumbers As Integer(), ByVal tail As Integer) As Integer

Dim check As Integer

Dim x As Integer

Dim done As Boolean = False

Do

If numbers(0, x) = 0 Then

x += 1

Else

For z = 0 To tail ' 0 to amount of numbers called

If numbers(0, x) = calledNumbers(z) Then

check += 1

End If

Next

done = True

End If

Loop Until done = True

x = 0

done = False

Do

If numbers(2, x) = 0 Then

x += 1

Else

For z = 0 To tail ' 0 to amount of numbers called

If numbers(2, x) = calledNumbers(z) Then

check += 1

End If

done = True

Next

End If

Loop Until done = True

x = 8

done = False

Do

If numbers(0, x) = 0 Then

x -= 1

Else

For z = 0 To tail ' 0 to amount of numbers called

If numbers(0, x) = calledNumbers(z) Then

check += 1

End If

done = True

Next

End If

Loop Until done = True

x = 8

done = False

Do

If numbers(0, x) = 0 Then

x -= 1

Else

For z = 0 To tail ' 0 to amount of numbers called

If numbers(0, x) = calledNumbers(z) Then

check += 1

End If

done = True

Next

End If

Loop Until done = True

Return check

End Function

**Check for horizontal in Bingo Bonus game**

Private Function Horizontal(ByVal calledNumbers As Integer(), ByVal tail As Integer) As Boolean

Dim matched As Integer

For y = 0 To 2

For x = 0 To 8

If numbers(y, x) <> 0 Then

For z = 0 To tail ' 0 to amount of numbers called

If numbers(y, x) = calledNumbers(z) Then

matched += 1 ' checks if numbers matched with calledNUmbers

End If

Next

End If

Next

If matched = 5 Then

Return True

Else

matched = 0

End If

Next

Return False

End Function

**Allow duck and other words to be called instead of numbers**

Private Function Caller() As Integer

Caller = numbers.nextBall

Console.WriteLine("and the next ball is.....")

If Caller = 11 Then

Console.WriteLine("legs 11")

ElseIf Caller = 2 Then

Console.WriteLine("two little ducks")

Else

Console.WriteLine(Caller)

End If

End Function

**Validate data entry**

Try Catch Everything