Unit 7.04 Programming Exercises

Number 1 MasterMind:

Public Class Form1

Const MAX\_PEGS As Integer = 10

Const MAX\_COLORS As Integer = 9

Dim numPegs As Integer

Dim numColors As Integer

Dim secretCode(-1) As Integer

Dim userGuess(-1) As Integer

Private Sub btnCheckGuess\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnCheckGuess.Click

Dim numPegsCorrect, numColorsCorrect As Integer

Call CorrectPegs(numPegsCorrect)

Call CorrectColors(numColorsCorrect)

Me.lblResults.Text = Me.lblResults.Text & "Pegs correct: " & \_

numPegsCorrect & Space(5) & "Colors correct: " & numColorsCorrect & vbCrLf

If numPegsCorrect = numPegs And numColorsCorrect = numPegs And \_

(numPegsCorrect <> 0 And numColorsCorrect <> 0) Then

MessageBox.Show("You Win!")

End If

End Sub

'Calculates the number of correct pegs chosen by the user.

'

'post: numPegsCorrect contains the number of correct pegs.

'

Sub CorrectPegs(ByRef numPegsCorrect As Integer)

numPegsCorrect = 0

For peg As Integer = 0 To userGuess.Length - 1

If userGuess(peg) = secretCode(peg) Then

numPegsCorrect += 1

End If

Next peg

End Sub

'Calculates the number of correct colors chosen by the user.

'

'post: The number of correct colors returned.

'

Sub CorrectColors(ByRef numColorsCorrect As Integer)

Dim used As Integer = -1

Dim secretCodeCopy(secretCode.Length) As Integer

numColorsCorrect = 0

'secretCode is copied so original is not changed

For peg As Integer = 0 To secretCode.Length - 1

secretCodeCopy(peg) = secretCode(peg)

Next peg

For guessPeg As Integer = 0 To userGuess.Length - 1

For secretPeg As Integer = 0 To secretCodeCopy.Length - 1

If userGuess(guessPeg) = secretCodeCopy(secretPeg) And \_

userGuess(guessPeg) <> used Then

numColorsCorrect += 1

userGuess(guessPeg) = used

secretCodeCopy(secretPeg) = used

End If

Next secretPeg

Next guessPeg

End Sub

Private Sub btnSelectPegs\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSelectPegs.Click

Dim numPegsEntered As String

Call DisableColors()

numPegsEntered = InputBox("Enter the number of pegs:")

If IsNumeric(numPegsEntered) And Val(numPegsEntered) > 0 And Val(numPegsEntered) <= MAX\_PEGS Then

'Assign a value to global variable numPegs

numPegs = Val(numPegsEntered)

ReDim secretCode(numPegs - 1)

ReDim userGuess(numPegs - 1)

'Assign a value to global variable numColors

Call GetNumColors()

'Assign colors to array of pegs

For index As Integer = 0 To (secretCode.Length - 1)

secretCode(index) = RndInt(1, numColors)

Next index

'Enable appropriate number of pegs

Call EnableColors()

Else

MessageBox.Show("Please enter a numeric value between 1 - " & MAX\_PEGS & " for the number of pegs.")

End If

End Sub

'Disables all color radio buttons.

'

'post: All color radio buttons are disabled.

'

Sub DisableColors()

Me.radPeg1Yellow.Enabled = False

Me.radPeg1Yellow.Checked = False

Me.radPeg1Red.Enabled = False

Me.radPeg1Red.Checked = False

Me.radPeg1Blue.Enabled = False

Me.radPeg1Blue.Checked = False

Me.radPeg1Green.Enabled = False

Me.radPeg1Green.Checked = False

Me.radPeg1White.Enabled = False

Me.radPeg1White.Checked = False

Me.radPeg1Purple.Enabled = False

Me.radPeg1Purple.Checked = False

Me.radPeg1Orange.Enabled = False

Me.radPeg1Orange.Checked = False

Me.radPeg1Brown.Enabled = False

Me.radPeg1Brown.Checked = False

Me.radPeg1Black.Enabled = False

Me.radPeg1Black.Checked = False

End Sub

'Prompts user for number of colors from 1 to MAX\_COLORS. If user enters a valid number then

'the numColors is set to that number, otherwise numColors is set to a default of MAX\_COLORS.

'

'post: The number of colors has been chosen by the user or set to MAX\_COLORS.

'

Sub GetNumColors()

Dim userColors As String = InputBox("Enter the number of colors:")

If IsNumeric(userColors) And Val(userColors) > 0 And Val(userColors) <= MAX\_COLORS Then

numColors = Val(userColors)

Else

MessageBox.Show("A default number of " & MAX\_COLORS & " has been selected for the number of colors.")

numColors = MAX\_COLORS

End If

End Sub

'Enables as many colors as the user has selected to use in the game

'

'post: numColors peg radio buttons have been enabled.

'

Sub EnableColors()

If numColors >= 1 Then

radPeg1Yellow.Enabled = True

End If

If numColors >= 2 Then

radPeg1Red.Enabled = True

End If

If numColors >= 3 Then

radPeg1Blue.Enabled = True

End If

If numColors >= 4 Then

radPeg1Green.Enabled = True

End If

If numColors >= 5 Then

radPeg1White.Enabled = True

End If

If numColors >= 6 Then

radPeg1Purple.Enabled = True

End If

If numColors >= 7 Then

radPeg1Orange.Enabled = True

End If

If numColors >= 8 Then

radPeg1Brown.Enabled = True

End If

If numColors >= 9 Then

radPeg1Black.Enabled = True

End If

End Sub

'Allows the user to make a guess.

'

'Post: An array of guesses has been created.

'

Private Sub btnMakeGuess\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnMakeGuess.Click

Static pegNum As Integer = 0

Dim guessMade As String = Nothing

If pegNum = 0 Then

MessageBox.Show("Directions: For each peg, select a color and then click the Make Guess button.")

MessageBox.Show("Select color choice for peg #1 and then select the Make Guess button.")

pegNum += 1

Me.grpPeg.Text = "Peg " & pegNum

ElseIf pegNum <= numPegs Then

Call GetGuess(pegNum - 1)

'Prepare for next guess

pegNum += 1

If pegNum <= numPegs Then

MessageBox.Show("Select color choice for peg #" & pegNum & " and then select the Make Guess button.")

Me.grpPeg.Text = "Peg " & pegNum

Else

'Show guess in label

For pegGuessed As Integer = 0 To userGuess.Length - 1

guessMade &= " " & userGuess(pegGuessed)

Next pegGuessed

Me.lblGuesses.Text &= guessMade & vbCrLf

MessageBox.Show("The color choices for all of the pegs have been entered. Select the Check Guess button to check the guesses.")

pegNum = 0

End If

End If

End Sub

'Gets the unique guesses of the MAX\_COLORS peg colors from the user.

'

'post: userGuess has been updated with the color guesses from the user.

'

Sub GetGuess(ByVal pegNum As Integer)

'Get guess of each peg

If Me.radPeg1Yellow.Checked Then

userGuess(pegNum) = 1

ElseIf Me.radPeg1Red.Checked Then

userGuess(pegNum) = 2

ElseIf Me.radPeg1Blue.Checked Then

userGuess(pegNum) = 3

ElseIf Me.radPeg1Green.Checked Then

userGuess(pegNum) = 4

ElseIf Me.radPeg1White.Checked Then

userGuess(pegNum) = 5

ElseIf Me.radPeg1Purple.Checked Then

userGuess(pegNum) = 6

ElseIf Me.radPeg1Orange.Checked Then

userGuess(pegNum) = 7

ElseIf Me.radPeg1Brown.Checked Then

userGuess(pegNum) = 8

ElseIf Me.radPeg1Black.Checked Then

userGuess(pegNum) = 9

End If

End Sub

'Calculates a random number between lowNum and highNum.

'

'post: a number between lowNum and highNum has been returned.

'

Function RndInt(ByVal lowNum As Integer, ByVal highNum As Integer) As Integer

Randomize()

Return Int((highNum - lowNum + 1) \* Rnd() + lowNum)

End Function

End Class

Number 2:

Pitch Pennies

Public Class Form1

Const PUZZLE As String = "Puzzle"

Const GAME As String = "Game"

Const BALL As String = "Ball"

Const POSTER As String = "Poster"

Const DOLL As String = "Doll"

Const NUM\_SQUARES As Integer = 25 'number of squares on interface

Dim board(NUM\_SQUARES) As String

Dim puzzleCount As Integer

Dim gameCount As Integer

Dim ballCount As Integer

Dim posterCount As Integer

Dim dollCount As Integer

Private Sub btnNewGame\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnNewGame.Click

Me.btnTossPennies.Enabled = True

Call Reset()

Call LoadPrizes()

Call LoadButtons()

End Sub

'Reset game for a new game.

'

'post: The Text property on every button on the interface has been cleared, all

'counter variables for the prizes have been reset to 0, and each element of the

'board array has been initialized with Nothing.

'

Sub Reset()

'Reset prize counters

puzzleCount = 0

gameCount = 0

ballCount = 0

posterCount = 0

dollCount = 0

'Clear text on buttons on interface

Me.btnSquare1.Text = Nothing

Me.btnSquare2.Text = Nothing

Me.btnSquare3.Text = Nothing

Me.btnSquare4.Text = Nothing

Me.btnSquare5.Text = Nothing

Me.btnSquare6.Text = Nothing

Me.btnSquare7.Text = Nothing

Me.btnSquare8.Text = Nothing

Me.btnSquare9.Text = Nothing

Me.btnSquare10.Text = Nothing

Me.btnSquare11.Text = Nothing

Me.btnSquare12.Text = Nothing

Me.btnSquare13.Text = Nothing

Me.btnSquare14.Text = Nothing

Me.btnSquare15.Text = Nothing

Me.btnSquare16.Text = Nothing

Me.btnSquare17.Text = Nothing

Me.btnSquare18.Text = Nothing

Me.btnSquare19.Text = Nothing

Me.btnSquare20.Text = Nothing

Me.btnSquare21.Text = Nothing

Me.btnSquare22.Text = Nothing

Me.btnSquare23.Text = Nothing

Me.btnSquare24.Text = Nothing

Me.btnSquare25.Text = Nothing

Me.lblAnswer.Text = Nothing

'Clear the array

For index As Integer = 0 To board.Length - 1

board(index) = Nothing

Next index

End Sub

'Stores prize names in board array.

'

'post: Three prize names of each prize (Puzzle, Game, Ball, Poster, Doll)

'have been randomly stored in board array.

'

Sub LoadPrizes()

Call LoadPiece(PUZZLE) 'fill 3 squares with Puzzle

Call LoadPiece(GAME) 'fill 3 squares with Game

Call LoadPiece(BALL) 'fill 3 squares with Ball

Call LoadPiece(POSTER) 'fill 3 squares with Poster

Call LoadPiece(DOLL) 'fill 3 squares with Doll

End Sub

'Stores a prize name in board array.

'

'post: Three prizes of prize have been stored in the array

'

Sub LoadPiece(ByVal prize As String)

Dim randomNum As Integer

Dim count As Integer = 0

Do

randomNum = RndInt(0, 24)

If board(randomNum) = Nothing Then

board(randomNum) = prize

count += 1

End If

Loop While count < 3

End Sub

'Display the prizes, stored in board array, on the buttons on the interface.

'

'post: Prize names or Nothing have been displayed on buttons.

'

Sub LoadButtons()

Me.btnSquare1.Text = board(0)

Me.btnSquare2.Text = board(1)

Me.btnSquare3.Text = board(2)

Me.btnSquare4.Text = board(3)

Me.btnSquare5.Text = board(4)

Me.btnSquare6.Text = board(5)

Me.btnSquare7.Text = board(6)

Me.btnSquare8.Text = board(7)

Me.btnSquare9.Text = board(8)

Me.btnSquare10.Text = board(9)

Me.btnSquare11.Text = board(10)

Me.btnSquare12.Text = board(11)

Me.btnSquare13.Text = board(12)

Me.btnSquare14.Text = board(13)

Me.btnSquare15.Text = board(14)

Me.btnSquare16.Text = board(15)

Me.btnSquare17.Text = board(16)

Me.btnSquare18.Text = board(17)

Me.btnSquare19.Text = board(18)

Me.btnSquare20.Text = board(19)

Me.btnSquare21.Text = board(20)

Me.btnSquare22.Text = board(21)

Me.btnSquare23.Text = board(22)

Me.btnSquare24.Text = board(23)

Me.btnSquare25.Text = board(24)

End Sub

'Pitches ten pennies onto a 5 x 5 board of 25 buttons and counts which squares are

'landed on.

'

'post: Ten pennies have been pitched onto the 5 x 5 board and displayed with an x on the

'buttons and counter variables for each of the prizes have been updated if a penny is

'pitched on that square.

'

Private Sub btnTossPennies\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnTossPennies.Click

Dim toss As Integer

Me.btnTossPennies.Enabled = False

For index As Integer = 1 To 10

toss = RndInt(0, 24) 'get which square penny pitched onto

board(toss) &= " x " 'mark interface with x for a penny

Next index

Call DeterminePrizes()

End Sub

'Displays the names of the prizes won or No prizes.

'

'post: The names of the prizes won have been displayed on the interface or No prizes

'has been displayed.

'

Sub DeterminePrizes()

Dim marking As String

Dim firstTwoChars As String

Dim lastChar As String

Call LoadButtons()

For index As Integer = 0 To 24

marking = board(index) 'get name of prize or Nothing on square

If marking <> Nothing Then

lastChar = marking.Substring(marking.Length - 2, 1)

If lastChar = "x" Then

firstTwoChars = marking.Substring(0, 2) 'get first two characters of text, if any

Select Case firstTwoChars 'update counters

Case "Pu"

puzzleCount += 1

Case "Ga"

gameCount += 1

Case "Ba"

ballCount += 1

Case "Po"

posterCount += 1

Case "Do"

dollCount += 1

End Select

End If

End If

Next index

If puzzleCount < 3 And gameCount < 3 And ballCount < 3 And \_

posterCount < 3 And dollCount < 3 Then

Me.lblAnswer.Text = "No prizes."

Else

Me.lblAnswer.Text = "Prizes won: "

If puzzleCount >= 3 Then

Me.lblAnswer.Text &= "puzzle "

End If

If gameCount >= 3 Then

Me.lblAnswer.Text &= "game "

End If

If ballCount >= 3 Then

Me.lblAnswer.Text &= "ball "

End If

If posterCount >= 3 Then

Me.lblAnswer.Text &= "poster "

End If

If dollCount >= 3 Then

Me.lblAnswer.Text &= "doll "

End If

End If

End Sub

'Calculates a random number between lowNum and highNum.

'

'post: a number between lowNum and highNum has been returned.

'

Function RndInt(ByVal lowNum As Integer, ByVal highNum As Integer) As Integer

Randomize()

Return Int((highNum - lowNum + 1) \* Rnd() + lowNum)

End Function

End Class

Number 3

Public Class Form1

Private Sub btnPlayGolf\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnPlayGolf.Click

Const NUM\_GOLFERS As Integer = 4

Const NUM\_HOLES As Integer = 9

Dim scoreCard(NUM\_GOLFERS - 1, NUM\_HOLES - 1) As Integer

Call Reset()

'Load Scorecard with scores 1 - 9

For col As Integer = 0 To scoreCard.GetLength(1) - 1

For row As Integer = 0 To scoreCard.GetLength(0) - 1

scoreCard(row, col) = RndInt(1, 9)

Next row

Next col

Call DisplayScorecard(scoreCard)

Call CalculateWinners(scoreCard)

End Sub

'Clears the Scorecard list box and winner display label.

'

'post: List box and label on form have been cleared.

'

Sub Reset()

Me.lstList.Items.Clear()

Me.lblPlayer1Winnings.Text = Nothing

Me.lblPlayer2Winnings.Text = Nothing

Me.lblPlayer3Winnings.Text = Nothing

Me.lblPlayer4Winnings.Text = Nothing

End Sub

'Displays the golf scores in the list box from the scoreCard two-dimensional

'array.

'

'post: The list box now displays each player and the corresponding scores for

'each hole of golf, stored in the scoreCard two-dimensional array.

'

Sub DisplayScorecard(ByRef scoreCard(,) As Integer)

Dim rowScore As String 'each row displaying player and holes

Me.lstList.Items.Add(vbTab & "Hole 1" & vbTab & "Hole 2" & vbTab & \_

"Hole 3" & vbTab & "Hole 4" & vbTab & "Hole 5" & vbTab & "Hole 6" & \_

vbTab & "Hole 7" & vbTab & "Hole 8" & vbTab & "Hole 9")

For row As Integer = 0 To scoreCard.GetLength(0) - 1

rowScore = Nothing

For col As Integer = 0 To scoreCard.GetLength(1) - 1

rowScore &= scoreCard(row, col) & vbTab

Next col

Me.lstList.Items.Add("Player " & row + 1 & vbTab & rowScore)

Next row

End Sub

'Calculates and displays the winner of each hole by comparing the scores of

'all the players.

'

'pre: scoreCard contains scores.

'post: The holes each player won (determined by having the lowest number of

'strokes on that hole) have been displayed in a label.

'

Sub CalculateWinners(ByRef scoreCard(,) As Integer)

Dim winnerRow1 As String = Nothing 'the concatenation of all player 1 wins

Dim winnerRow2 As String = Nothing 'the concatenation of all player 2 wins

Dim winnerRow3 As String = Nothing 'the concatenation of all player 3 wins

Dim winnerRow4 As String = Nothing 'the concatenation of all player 4 wins

Dim lowScore As Integer

Dim winnerRow As Integer

For col As Integer = 0 To scoreCard.GetLength(1) - 1

lowScore = 9 'initialize to highest stroke value

For row As Integer = 0 To scoreCard.GetLength(0) - 1 'compare all of the scores

If lowScore > scoreCard(row, col) Then

lowScore = scoreCard(row, col) 'set low score

winnerRow = row 'set player # of low score

End If

Next row

If winnerRow = 0 Then 'update winner of hole

winnerRow1 &= " Hole " & col + 1

ElseIf winnerRow = 1 Then

winnerRow2 &= " Hole " & col + 1

ElseIf winnerRow = 2 Then

winnerRow3 &= " Hole " & col + 1

Else

winnerRow4 &= " Hole " & col + 1

End If

Next col

'Display how many holes each player won overall

Me.lblPlayer1Winnings.Text = winnerRow1

Me.lblPlayer2Winnings.Text = winnerRow2

Me.lblPlayer3Winnings.Text = winnerRow3

Me.lblPlayer4Winnings.Text = winnerRow4

End Sub

'Calculates a random number between lowNum and highNum.

'

'post: a number between lowNum and highNum has been returned.

'

Function RndInt(ByVal lowNum As Integer, ByVal highNum As Integer) As Integer

Randomize()

Return Int((highNum - lowNum + 1) \* Rnd() + lowNum)

End Function

End Class