```
Imports System.Math

Public Class _1_Player

    Dim TotalTime As Integer
    Dim Animation As Boolean = False

    Dim Drag = False
    Dim Rotation As Boolean = False

    Dim Grid1(9, 9) As Integer
    Dim Grid2(9, 9) As Integer

    Dim TurnNum As Integer = 2

    Dim SetupComplete As Boolean = False

    Dim BorderWidth As Integer = SystemInformation.BorderSize.Width
    Dim TitlebarHeight As Integer = SystemInformation.CaptionHeight + BorderWidth

    Dim RecentHit As Boolean = False 'Creates a varible to see whether the computer has recently hit a ship

    Dim MemPosX As Integer 'Creates a variable to store where the computer got it's initial hit
    Dim MemPosY As Integer

    Dim PosX As Integer 'Creates a varible to store where the computer is 'looking' to fire next
    Dim PosY As Integer

    Dim Side As Integer = -1 'Creates a variable that stores which side of the inital hit square a computer should shoot at
    Dim FollowSide As Boolean = False 'Creates a variable to see whether the computer should pick another side

    Dim ReUseSides(3) As Integer 'Creates an array to store the sides that have already been fired at
    Dim ReUseSidesCount As Integer = 0 'Creates a variable to count how many sides have been fired at

    Dim BoatColl As New Microsoft.VisualBasic.Collection()

    Dim MaxHits(4) As Integer
    Dim CurrentHits1(4) As Integer
    Dim CurrentHits2(4) As Integer

    Dim Player1ShipsSunk As Integer = 0
    Dim Player2ShipsSunk As Integer = 0

    Dim ComputerTypeShip(9, 9) As String 'Creates a 2 dimensional array that stores what kind of ship is where on the grid

    Dim ChangedSides As Boolean = False 'Creats a variable to test whether the ship changed the side it was following

    Private Sub picLargeBoat_Down(ByVal sender As System.Object, ByVal e As System.Windows.Forms.MouseEventArgs) Handles picXLrgBoat1.MouseDown, picSmlBoat1. ↙
    MouseDown, picMedBoat2.MouseDown, picMedBoat1.MouseDown, picLrgBoat1.MouseDown
```

```
        If SetupComplete = False Then

            Drag = True

        End If

    End Sub

    Private Sub CheckIfOutside(ByVal BoatLocY, ByVal BoatLocX, ByVal Boat)

        With Boat

            If Rotation = True Then

                If BoatLocY < 0 Or BoatLocY > 9 Or BoatLocX < 0 Or BoatLocX + ((.Width) / 24) - 1 > 9 Then
                    .tag = .tag & "outside"
                End If

            Else

                If BoatLocY < 0 Or BoatLocY + ((.Height) / 24) - 1 > 9 Or BoatLocX < 0 Or BoatLocX > 9 Then
                    .tag = .tag & "outside"
                End If

            End If

        End With

    End Sub

    Private Sub CheckIfRotated(ByVal sender)

        With sender

            If .size.height > .size.width Then

                Rotation = False

            Else

                Rotation = True

            End If

        End With

    End Sub
    Private Sub picLargeBoat_Up(ByVal sender As System.Object, ByVal e As System.Windows.Forms.MouseEventArgs) Handles picXLrgBoat1.MouseUp, picSmlBoat1.MouseUp, ↙
     picMedBoat2.MouseUp, picMedBoat1.MouseUp, picLrgBoat1.MouseUp

        With sender

            If SetupComplete = False Then

                Drag = False

                .Left = (.Left - (.Left Mod 24)) + 13
                .Top = (.Top - (.Top Mod 24)) + 13

            End If

            .BackColor = Color.Aqua

        End With

    End Sub

    Private Sub picLargeBoat_Move(ByVal sender As System.Object, ByVal e As System.Windows.Forms.MouseEventArgs) Handles picXLrgBoat1.MouseMove, picSmlBoat1.    ↙
    MouseMove, picMedBoat2.MouseMove, picMedBoat1.MouseMove, picLrgBoat1.MouseMove

        If SetupComplete = False Then

            If Drag = True Then

                sender.Left = (MousePosition.X - Me.Location.X - 13)
                sender.Top = (MousePosition.Y - Me.Location.Y - TitlebarHeight - 13)

            End If

        End If

    End Sub

    Private Sub picLargeBoat_Rotate(ByVal sender As System.Object, ByVal e As System.Windows.Forms.MouseEventArgs) Handles picXLrgBoat1.MouseWheel, picSmlBoat1. ↙
    MouseWheel, picMedBoat2.MouseWheel, picMedBoat1.MouseWheel, picLrgBoat1.MouseWheel

        If SetupComplete = False Then

            With sender

                CheckIfRotated(sender)

                Dim Width As Integer = .Size.Width
                Dim Height As Integer = .Size.Height

                .Size = New System.Drawing.Size(Height, Width) 'https://msdn.microsoft.com/en-us/library/system.windows.forms.picturebox.sizemode(v=vs.110).aspx
                If Rotation = False Then

                    Rotation = True
                    .Image = My.Resources.ResourceManager.GetObject(CStr(.Size.Width / 24) & " long ship Rotation") 'http://stackoverflow.com/questions/15282874/ ↙
    make-a-button-click-change-a-picture-in-the-picture-box-visual-basic

                Else

                    Rotation = False
                    .Image = My.Resources.ResourceManager.GetObject(CStr(.Size.Height / 24) & " long ship")

                End If

            End With

        End If

    End Sub

    Private Sub btnReset_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnReset.Click

        frmBattleship.Show()
        Me.Close()

    End Sub

    Private Function RandomNumberComputerPlacement(ByVal x As Integer, ByVal y As Integer) As Integer 'Creates a sub like thing that can be used as a number ↙
    value and/or returns a value

        Return x + Math.Floor(Rnd() * y)

    End Function

    Private Sub btnConfirm_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnConfirm.Click

        Dim Cont As Boolean = True

        For i = 1 To BoatColl.Count

            BoatColl(i).tag = ""

        Next

        For i = 1 To BoatColl.Count

            For k = 1 To BoatColl.Count

                If i <> k AndAlso BoatColl(i).Bounds.intersectsWith(BoatColl(k).Bounds) = True Then
```

```
                    BoatColl(k).tag = BoatColl(k).tag & "ontop"
                    i = BoatColl.Count
                    Exit For
                End If

            Next

        Next

        Dim BoatLocX As Integer
        Dim BoatLocY As Integer

        For i = 1 To BoatColl.Count

            BoatLocX = (BoatColl(i).Location.X - picGrid100.Location.X) / 24
            BoatLocY = (BoatColl(i).Location.Y - picGrid100.Location.Y) / 24

            CheckIfRotated(BoatColl(i))

            CheckIfOutside(BoatLocY, BoatLocX, BoatColl(i))

        Next

        For i = 1 To BoatColl.Count()

            If BoatColl(i).tag.contains("outside") = True Then

                Cont = False
                BoatColl(i).BackColor = Color.Red
                BoatColl(i).BringToFront()
                MsgBox("Please put your boats inside the grid")
                Exit For

            ElseIf BoatColl(i).tag.contains("ontop") = True Then

                Cont = False
                BoatColl(i).BackColor = Color.Red
                BoatColl(i).BringToFront()
                MsgBox("Please don't your boats on top of one another")
                Exit For

            End If
        Next

        If Cont = True Then

            'Entering into the array
            For i = 1 To BoatColl.Count
```

```
                BoatLocX = (BoatColl(i).Location.X - picGrid100.Location.X) / 24
                BoatLocY = (BoatColl(i).Location.Y - picGrid100.Location.Y) / 24

                CheckIfRotated(BoatColl(i))

                If Rotation = True Then

                    For k = 0 To ((BoatColl(i).Size.Width) / 24) - 1
                        Grid1(BoatLocY, BoatLocX + k) = 1
                    Next

                Else

                    For k = 0 To ((BoatColl(i).Size.Height) / 24) - 1
                        Grid1(BoatLocY + k, BoatLocX) = 1
                    Next

                End If

            Next

            btnConfirm.Visible = False
            SetupComplete = True
            lblGameTime.Visible = True
            tmrGameTime.Enabled = True

            Dim BoatNum = 2 'Defines a variable to the first 'ship number' so then a human can make tracking what ship it is up to easier
            Dim PlaceX As Integer = 0 'Defines a varible that will track which point in the array next needs to be set to have a ship in it
            Dim PlaceY As Integer = 0
            Dim RandomRotate As Integer = 0 'Defines a variable to hold the rotation decider

            For i = 1 To 5

                RandomRotate = Math.Floor(Rnd() * 2) 'Rnd will create a random number between 0 and 1, by timesing it by two, you get when the number is floored ↙
    (the decimal part of the number removed), either a 0 or 1

                If RandomRotate = 0 Then 'Determines whether a ship will be vertical or horizontal
                    Rotation = True
                Else
                    Rotation = False
                End If

                Select Case i 'See's which ship is currently trying to be placed

                    Case 1

                        BoatNum = 2

                        If Rotation = True Then
                            PlaceX = RandomNumberComputerPlacement(7, 2) 'Calls the randomnumbercomputerplacement sub to place a ship as little as the 7th column ↙
     and randomly for the last 2, this ensures ships will never overlap in placement although it is slightly less random
                            PlaceY = RandomNumberComputerPlacement(8, 2)

                        Else

                            PlaceX = RandomNumberComputerPlacement(8, 2)
                            PlaceY = RandomNumberComputerPlacement(7, 2)

                        End If

                    Case 2
                        BoatNum = 3

                        If Rotation = True Then

                            PlaceX = 0
                            PlaceY = RandomNumberComputerPlacement(6, 4)

                        Else

                            PlaceX = RandomNumberComputerPlacement(0, 3)
                            PlaceY = RandomNumberComputerPlacement(6, 2)

                        End If

                    Case 3
                        BoatNum = 3

                        If Rotation = True Then

                            PlaceX = 3
                            PlaceY = RandomNumberComputerPlacement(6, 4)

                        Else

                            PlaceX = RandomNumberComputerPlacement(3, 3)
                            PlaceY = RandomNumberComputerPlacement(6, 2)

                        End If

                    Case 4
                        BoatNum = 4

                        If Rotation = True Then

                            PlaceX = 6
                            PlaceY = RandomNumberComputerPlacement(0, 7)
                        Else

                            PlaceX = RandomNumberComputerPlacement(6, 3)
                            PlaceY = RandomNumberComputerPlacement(0, 2)

                        End If

                    Case 5
                        BoatNum = 5

                        If Rotation = True Then

                            PlaceX = RandomNumberComputerPlacement(0, 2)
                            PlaceY = RandomNumberComputerPlacement(0, 6)

                        Else

                            PlaceX = RandomNumberComputerPlacement(0, 6)
                            PlaceY = RandomNumberComputerPlacement(0, 2)

                        End If

                End Select

                If Rotation = True Then

                    For k = 0 To (BoatNum - 1) 'Tests whether the ship was rotated

                        Select Case i

                            Case 1
                                ComputerTypeShip(PlaceY, PlaceX + k) = "s" 'Enters the type of ship into a separate array to keep track of which ships have been ↙
    hit a certain amount of times, such as with current hits in player 2
                            Case 2
                                ComputerTypeShip(PlaceY, PlaceX + k) = "m1"
                            Case 3
                                ComputerTypeShip(PlaceY, PlaceX + k) = "m2"
                            Case 4
                                ComputerTypeShip(PlaceY, PlaceX + k) = "l"
                            Case 5
                                ComputerTypeShip(PlaceY, PlaceX + k) = "x"

                        End Select

                        Grid2(PlaceY, PlaceX + k) = 1

                    Next
                Else

                    For k = 0 To (BoatNum - 1)

                        Select Case i

                            Case 1
                                ComputerTypeShip(PlaceY + k, PlaceX) = "s" 'Adjusted placement for the other orentation
                            Case 2
                                ComputerTypeShip(PlaceY + k, PlaceX) = "m1"
                            Case 3
                                ComputerTypeShip(PlaceY + k, PlaceX) = "m2"
                            Case 4
                                ComputerTypeShip(PlaceY + k, PlaceX) = "l"
                            Case 5
                                ComputerTypeShip(PlaceY + k, PlaceX) = "x"

                        End Select

                        Grid2(PlaceY + k, PlaceX) = 1

                    Next

                End If

            Next

            NextTurnCPU() 'Calls on the sub NextTurnCpu

        End If

        Cont = False

    End Sub

    Private Sub picGrid2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles picGrid299.Click, picGrid298.Click, picGrid297.Click,  ↙
    picGrid296.Click, picGrid295.Click, picGrid294.Click, picGrid293.Click, picGrid292.Click, picGrid291.Click, picGrid290.Click, picGrid289.Click, picGrid288. ↙
    Click, picGrid287.Click, picGrid286.Click, picGrid285.Click, picGrid284.Click, picGrid283.Click, picGrid282.Click, picGrid281.Click, picGrid280.Click,  ↙
    picGrid279.Click, picGrid278.Click, picGrid277.Click, picGrid276.Click, picGrid275.Click, picGrid274.Click, picGrid273.Click, picGrid272.Click, picGrid271. ↙
    Click, picGrid270.Click, picGrid269.Click, picGrid268.Click, picGrid267.Click, picGrid266.Click, picGrid265.Click, picGrid264.Click, picGrid263.Click,  ↙
    picGrid262.Click, picGrid261.Click, picGrid260.Click, picGrid259.Click, picGrid258.Click, picGrid257.Click, picGrid256.Click, picGrid255.Click, picGrid254. ↙
    Click, picGrid253.Click, picGrid252.Click, picGrid251.Click, picGrid250.Click, picGrid249.Click, picGrid248.Click, picGrid247.Click, picGrid246.Click,  ↙
    picGrid245.Click, picGrid244.Click, picGrid243.Click, picGrid242.Click, picGrid241.Click, picGrid240.Click, picGrid239.Click, picGrid238.Click, picGrid237. ↙
    Click, picGrid236.Click, picGrid235.Click, picGrid234.Click, picGrid233.Click, picGrid232.Click, picGrid231.Click, picGrid230.Click, picGrid229.Click,  ↙
    picGrid228.Click, picGrid227.Click, picGrid226.Click, picGrid225.Click, picGrid224.Click, picGrid223.Click, picGrid222.Click, picGrid221.Click, picGrid220. ↙
    Click, picGrid219.Click, picGrid218.Click, picGrid217.Click, picGrid216.Click, picGrid215.Click, picGrid214.Click, picGrid213.Click, picGrid212.Click,  ↙
    picGrid211.Click, picGrid210.Click, picGrid209.Click, picGrid208.Click, picGrid207.Click, picGrid206.Click, picGrid205.Click, picGrid204.Click, picGrid203. ↙
    Click, picGrid202.Click, picGrid201.Click, picGrid200.Click
```

```
        If Animation = False Then

            Dim Won As Boolean = False 'Creates a variable to hold whether the user has won as the time it takes to close allows the computer to have another ↙
    turn

            With sender

                Dim ClickPosX As Integer = (.Location.X - picGrid200.Location.X) / 24

                Dim ClickPosY As Integer = (.Location.Y - picGrid100.Location.Y) / 24

                If .tag <> "fired on" Then

                    Shot(.Location.X, .Location.Y)

                    If Grid2(ClickPosY, ClickPosX) = 1 Then

                        Explode(sender)
                        .BackColor = Color.Red
                        MsgBox("Hit")
                        .BringToFront()

                        Dim BoatNumHit As Integer

                        Select Case ComputerTypeShip(ClickPosY, ClickPosX) 'Sees what type of ship was hit

                            'determines which ship should have a hit added to it
                            Case "s"
                                BoatNumHit = 4
                                CurrentHits2(BoatNumHit) += 1
                            Case "m1"
                                BoatNumHit = 3
                                CurrentHits2(BoatNumHit) += 1
                            Case "m2"
                                BoatNumHit = 2
                                CurrentHits2(BoatNumHit) += 1
                            Case "l"
                                BoatNumHit = 1
                                CurrentHits2(BoatNumHit) += 1
                            Case "x"
                                BoatNumHit = 0
                                CurrentHits2(BoatNumHit) += 1

                        End Select

                        If CurrentHits2(BoatNumHit) = MaxHits(BoatNumHit) Then

                            MsgBox(MaxHits(BoatNumHit) & " size ship sunk!")
                            Player2ShipsSunk = Player2ShipsSunk + 1
                            CurrentHits2(BoatNumHit) += 1

                        End If

                        If Player2ShipsSunk = 5 Then

                            Won = True

                            MsgBox("Congratulations! You have won!")
                            frmBattleship.Show()
                            Me.Close()

                        End If

                    Else

                        .BackColor = Color.White
                        MsgBox("Miss")

                    End If


                    If Won = False Then 'Makes sure that the computer cannot have another go before the form closes as it was

                        NextTurnCPU()

                        .tag = "fired on"

                    End If

                Else

                    MsgBox("You cannot fire at the same spot twice")

                End If

            End With

        End If

    End Sub

    Private Sub NextTurnCPU()

        TurnNum = 2
        MsgBox("Computer Turn") 'Tells the user what is happening

        If RecentHit = True Then 'Checks whether the computer has hit a ship recently
```

```
            FollowHit() 'Runs the followhit module

        Else

            Do

                PosX = Math.Floor(Rnd() * 10) 'Picks a random spot on the grid to fire at
                PosY = Math.Floor(Rnd() * 10)

            Loop While Me.Controls("picGrid1" & PosY.ToString & PosX.ToString).Tag = "fired on" 'A post test repetition loop so then it randoms it's fire at ↙
    least once

            ComputerFire() 'Runs the computer fire module

        End If

    End Sub

    Private Sub ComputerFire()

        With Me.Controls("picGrid1" & PosY.ToString & PosX.ToString) 'http://www.dreamincode.net/forums/topic/320630-using-a-variable-in-the-name-of-an-object/

            Dim FirePosX As Integer = (24 * PosX) + picGrid100.Location.X + 3 'Convert a grid position into coordinates that can be tested against the positions ↙
    of the ships on the players side of the board
            Dim FirePosY As Integer = (24 * PosY) + picGrid100.Location.Y + 3

            Shot(FirePosX - 3, FirePosY - 3)

            If Grid1(PosY, PosX) = 1 Then 'If the computer gets a hit

                If RecentHit = False Then 'If the computer hasn't hit recently

                    RecentHit = True 'Makes it so then the computer realises it's hit recently
                    MemPosX = PosX 'Sets a spot so then the computer can go back to that space when it runs out of sides to test
                    MemPosY = PosY

                    For i = 0 To 3

                        ReUseSides(i) = -1 'Clears the ReUseSides array so then it can be used again

                    Next

                    ReUseSidesCount = 0 'Resets the count so it is ready to put bad sides into the array

                    For i = 0 To 3

                        Side = i

                        CheckSpot() 'Checks if there are bad spots around where the computer initially fired so then it doesn't have to find them randomly as ↙
    that had issues

                    Next

                Else

                    FollowSide = True 'If the computer has hit recently, then it says that it should follow this side

                End If

                .BringToFront()
                Explode(Me.Controls("picGrid1" & PosY.ToString & PosX.ToString))
                .BackColor = Color.Red
                MsgBox("Hit")

                Dim BoatNumHit As Integer 'Creates a variable that stores which boat was hit

                For i = 1 To 5

                    If (FirePosX > BoatColl(i).Left And FirePosX < BoatColl(i).Right And FirePosY > BoatColl(i).Top And FirePosY < BoatColl(i).Bottom) Then 'Is ↙
    supposed to test which boat was hit

                        BoatNumHit = i - 1 'Sets BoatNumHit to be it's number that it is known in in the arrays
                        Exit For

                    End If

                Next

                CurrentHits1(BoatNumHit) += 1 'Increments that current hit on the ship

                If CurrentHits1(BoatNumHit) = MaxHits(BoatNumHit) Then 'Tests if that hit would sink the ship

                    MsgBox(MaxHits(BoatNumHit) & " size ship sunk by computer!") 'Gives the user feedback as to what ship was sunk
                    Player1ShipsSunk = Player1ShipsSunk + 1
                    CurrentHits1(BoatNumHit) += 1
                    RecentHit = False

                End If

                If Player1ShipsSunk = 5 Then 'I all the player 1 ships have been sunk

                    MsgBox("Bad Luck! Computer has won!") 'Shows the user that the computer has won
                    frmBattleship.Show()
                    Me.Close()

                End If

            Else
```

```
                If RecentHit = True Then 'If the computer has recently hit

                    ChangeSide() 'Calls the ChangeSides sub

                End If

                If ReUseSidesCount = 4 Then 'If the computer is on it's last possible side it can follow

                    RecentHit = False 'Makes it so then the computer goes back to randomly firing

                End If

                .BackColor = Color.White
                MsgBox("Miss")

            End If

            .BringToFront()
            .Tag = "fired on"

            TurnNum = 1
            MsgBox("Your turn")

        End With

    End Sub

    Private Sub FollowHit()

Line1:  'https://msdn.microsoft.com/en-us/library/69whc95c.aspx 'Provides an 'anchor point' so then the program knows, when the Goto command is used, where to go ↙
     to

        If FollowSide = False Then 'If the computer isn't currently trying to following a side

            PickSide() 'Picks a random side that isnt in the ReUseSides array

        End If

        ChangedSides = False

        JumpHit() 'Calls the jump hit sub which tries to skip over already hit spaces

        CheckSpot() 'Calls the CheckSpot sub which makes sure that the spot it is about to fire on is legal

        If ChangedSides = True Then 'If the computer changed sides during the CheckSpot check

            If ReUseSidesCount <> 4 Then 'If it isn't the last side the computer has determined can't be fired on
                GoTo Line1 'Goes back to the line "Line1"
            End If

        Else

            ComputerFire() 'Calls the computer fire function

        End If

    End Sub

    Private Sub PickSide()

        While Side = ReUseSides(0) Or Side = ReUseSides(1) Or Side = ReUseSides(2) Or Side = ReUseSides(3) 'Continue looping until the side randomly picked is ↙
    not in the ReUseSides array

            Side = Math.Floor(Rnd() * 4) 'Randomly pick a number between 0 and 3 inclusive

        End While

    End Sub

    Private Sub CheckSpot()

        OutsideGridChangeSides() 'Checks if the spot the computer is trying to fire on is outside of the grid

        If (Me.Controls("picGrid1" & PosY.ToString & PosX.ToString).Tag = "fired on" And Grid1(PosY, PosX) = 0) Then 'Tests if the computer has already fired ↙
    there and it was a miss

            ChangeSide() 'Calls the ChangeSide sub

        End If

    End Sub

    Private Sub OutsideGridChangeSides()

        'Depending on which side is selected to be fired on and where the computer is trying to fire depends on whether the computer needs to try and change ↙
    sides. Eg if it's trying to fire to the left but it's already at the left most point of the grid it needs to change sides
        Select Case Side

            Case 0
                If PosY > 8 Then
                    ChangeSide()
                End If
            Case 1
                If PosX > 8 Then
                    ChangeSide()
                End If
            Case 2
```

```
                If PosY < 1 Then
                    ChangeSide()
                End If
            Case 3
                If PosX < 1 Then
                    ChangeSide()
                End If
        End Select

    End Sub

    Private Sub ChangeSide()

        PosX = MemPosX 'Sets where the computer is firing from, back to the original place it hit
        PosY = MemPosY
        ReUseSides(ReUseSidesCount) = Side 'Adds the bad side into the ReUseSides array so then it doesn't get repicked
        ReUseSidesCount += 1 'Adds to the count of where it puts the side into the array
        FollowSide = False 'Stops following that side

        ChangedSides = True 'Because the computer changed sides, it sets changed sides to true

        If ReUseSidesCount = 4 Then 'If the computer has tried all sides it goes back to firing randomly

            RecentHit = False

        End If

    End Sub

    Private Sub JumpHit()

        Do

            Select Case Side
                'Continues to move in the direction of the side
                Case 0
                    PosY += 1
                Case 1
                    PosX += 1
                Case 2
                    PosY -= 1
                Case 3
                    PosX -= 1

            End Select

        Loop While (Me.Controls("picGrid1" & PosY.ToString & PosX.ToString).Tag = "fired on" And Grid1(PosY, PosX) = 1) 'Post test repition loop where as long as ↙
      the computer has hit a square it tries to 'jump' over it
    End Sub

    Private Sub frmBattleship_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load 'Runs this sub when the form loads

        BoatColl.Add(picXLrgBoat1) 'Adds the boats to the collection array thing
        BoatColl.Add(picLrgBoat1)
        BoatColl.Add(picMedBoat1)
        BoatColl.Add(picMedBoat2)
        BoatColl.Add(picSmlBoat1)

        For i = 0 To 4 'Resets the arrays of the arrays corresponding to boats and their hits

            MaxHits(i) = BoatColl(i + 1).Size.Height / 24
            CurrentHits1(i) = 0
            CurrentHits2(i) = 0

        Next

        Randomize() 'https://msdn.microsoft.com/en-us/library/8zedbtdt.aspx As randomness works off a seed in vb, the seed needs to be randomised each time to ↙
    make it random

    End Sub

    Private Sub btnHelp_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnHelp.Click

        Help.Show() 'Shows the help form when the help button is clicked

    End Sub

    Private Sub tmrGameTime_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles tmrGameTime.Tick

        lblGameTime.Text = "Game Time: " & TotalTime & " seconds"

        TotalTime += 1

    End Sub

    Private Sub Explode(ByVal GridSpot As Object)

        Animation = True

        With GridSpot

            For i = 4 To 1 Step -1

                .Image = My.Resources.ResourceManager.GetObject("Explosion" & i)
                Wait(100)

            Next
            For i = 1 To 4

                .Image = My.Resources.ResourceManager.GetObject("Explosion" & i)
                Wait(100)

            Next

            .Image = Nothing

        End With

        Animation = False

    End Sub

    Private Sub Shot(ByVal LocX As Integer, ByVal LocY As Integer)

        With picShot

            My.Computer.Audio.Play(My.Resources.Cannon, AudioPlayMode.Background)

            .BringToFront()

            Animation = True 'Tells the program it's doing an animation
            .Visible = True 'Allows the shot to be seen

            Dim StartX As Integer 'Defines integers for the start position of the shot
            Dim StartY As Integer

            Dim DistX As Integer 'Defines integers to store the amount of distance the shot must travel
            Dim DistY As Integer

            Dim Gradient As Decimal 'Holds the gradient of the line between the start point and the dynamic point given by parameters

            If TurnNum = 1 Then 'Checks which turn it is to change the starting position of the shot accordingly

                StartX = 254 'Sets the starting position of the shot
                StartY = 133

                DistX = LocX - StartX 'Gets the distance between the points so then the gradient formula can be used
                DistY = LocY - StartY

                .Left = StartX 'Positions the shot at the start position
                .Top = StartY

                Gradient = 5 * (DistY / DistX) 'Gets 5 times the gradient value so then it can be reasonably quick and still look decent when the animation is ↙
    going
```

```
                While .Left < LocX 'Keeps on going till it's past it's point

                    .Left += 5 'Moves the shot towards the point given and because gradient is how far up it goes per 1 across it is also timsed by 5
                    .Top += Gradient
                    Wait(40) 'Gives the impression that it's moving

                End While

            ElseIf TurnNum = 2 Then

                StartX = 315
                StartY = 133

                DistX = LocX - StartX
                DistY = LocY - StartY

                .Left = StartX
                .Top = StartY

                Gradient = 5 * (DistY / DistX)

                While .Left > LocX

                    .Left -= 5
                    .Top -= Gradient
                    Wait(40)

                End While

            End If

            Animation = False 'Tells the program that the animation has stopped
            .Visible = False 'Makes the shot invisible again

        End With

    End Sub

    Private Sub Wait(ByVal time)

        Dim sw As New Stopwatch
        sw.Start()
        Do While sw.ElapsedMilliseconds < time
            Application.DoEvents()
        Loop
        sw.Stop()
    End Sub

End Class
```