```
Imports System.Math 'Imports the System.Math library so some other functions can be used

Public Class _2_Player

    Dim TotalTime As Integer 'Creates a variable that stores the total amount of time passed per game
    Dim Animation As Boolean = False 'Creates a boolean variable to store whether an animation is in process as otherwise players can take their turns during the process

    Dim Drag = False 'Declares a boolean variable the holds the state of whether the user is dragging the ship or not
    Dim Rotation As Boolean = False 'Declares a boolean to store whether the ship is rotated or not

    Dim SecondPlacement As Boolean = False 'Holds whether its the second players turn to place the ships

    Dim Grid1(9, 9) As Integer 'Stores where the ships are on the first grid
    Dim Grid2(9, 9) As Integer 'Stores where the ships are on the second grid

    Dim TurnNum As Integer = 2 'Determines whose turn it is

    Dim SetupComplete As Boolean = False 'Stores whether setup has been completed

    Dim BorderWidth As Integer = SystemInformation.BorderSize.Width 'Stores the value of the size of the borders of the form so as it works across all versions of windows
    Dim TitlebarHeight As Integer = SystemInformation.CaptionHeight + BorderWidth 'Retrieved from https://ivision.wordpress.com/2007/01/05/title-bar-height-and-form-border-width-of-net-form/ Gets the size of the title bar + the border width so then it doesnt matter what version of windows it is run on

    Dim BoatColl1 As New Microsoft.VisualBasic.Collection() 'Creates essentially an array that will store all of the first players ships
    Dim BoatColl2 As New Microsoft.VisualBasic.Collection() 'Creates essentially an array that will store all of the second players ships

    Dim MaxHits(4) As Integer 'Creates an array of integers that is used to store the maximum amount of hits
    Dim CurrentHits1(4) As Integer 'Creates an array of integers that stores the current amount of hits each of player 1's ships has taken
    Dim CurrentHits2(4) As Integer 'Creates an array of integers that stores the current amount of hits each of player 2's ships has taken

    Dim Player1ShipsSunk As Integer 'Creates an array of integers that store how many of player 1's ships has been sunk
    Dim Player2ShipsSunk As Integer 'Creates an array of integers that store how many of player 2's ships has been sunk

    Private Sub picLargeBoat_Down(ByVal sender As System.Object, ByVal e As System.Windows.Forms.MouseEventArgs) Handles picXLrgBoat2.MouseDown, picXLrgBoat1.MouseDown, picSmlBoat2.MouseDown, picSmlBoat1.MouseDown, picMedBoat4.MouseDown, picMedBoat3.MouseDown, picMedBoat2.MouseDown, picMedBoat1.MouseDown, picLrgBoat2.MouseDown, picLrgBoat1.MouseDown 'Runs this sub whenever these events are triggered, in this case, when the mouse button is first put down

        If SetupComplete = False Then 'Tests if the setup stage is still active

            Drag = True 'Sets drag to be true

        End If

    End Sub

    Private Sub CheckIfOutside(ByVal BoatLocY, ByVal BoatLocX, ByVal Boat) 'Declares a module that requires parameters to run, BoatLocY, BoatLocX and sender. This module wasn't automatically made by double clickling and element on the GUI
```

```
        With Boat 'Perform all actions that alter properties to this if they start with a full stop

            If Rotation = True Then 'Tests if the boat has been rotated

                If BoatLocY < 0 Or BoatLocY > 9 Or BoatLocX < 0 Or BoatLocX + ((.Width) / 24) - 1 > 9 Then 'Tests if the first square of the boat and the last square of the boat are not inside the grid
                    .tag = .tag & "outside" 'If one of the squares was outside the grid, then set that boat's tag to be 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 'Tests if the senders height is greater than it's width, essentially seeing if it is vertical

                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 picXLrgBoat2.MouseUp, picXLrgBoat1.MouseUp, picSmlBoat2.MouseUp, picSmlBoat1.MouseUp, picMedBoat4.MouseUp, picMedBoat3.MouseUp, picMedBoat2.MouseUp, picMedBoat1.MouseUp, picLrgBoat2.MouseUp, picLrgBoat1.MouseUp 'Triggers this sub when the mouse button is released

        With sender

            If SetupComplete = False Then

                Drag = False
```

```
            .Left = (.Left - (.Left Mod 24)) + 13 'moves the left hand side of the object to a multiple of 24 + 13 so it 'locks on' to a grid square
            .Top = (.Top - (.Top Mod 24)) + 13 'moves the left hand side of the object to a multiple of 24 + 13 so it 'locks on' to a grid square

        End If

        .BackColor = Color.Aqua 'changes the colour of the boat to be aqua so then it looks like the ship isn't just a rectangle, purely for graphic effect

    End With

End Sub

Private Sub picLargeBoat_Move(ByVal sender As System.Object, ByVal e As System.Windows.Forms.MouseEventArgs) Handles picXLrgBoat2.MouseMove, picXLrgBoat1.MouseMove, picSmlBoat2.MouseMove, picSmlBoat1.MouseMove, picMedBoat4.MouseMove, picMedBoat3.MouseMove, picMedBoat2.MouseMove, picMedBoat1.MouseMove, picLrgBoat2.MouseMove, picLrgBoat1.MouseMove 'Triggers this sub when the mouse is moved over the objects

    If SetupComplete = False Then

        If Drag = True Then

            sender.Left = (MousePosition.X - Me.Location.X - 13) 'Moves the object to be on the mouse to make it 'drag' on the X axis
            sender.Top = (MousePosition.Y - Me.Location.Y - TitlebarHeight - 13) 'Moves the object to be on the mouse to make it 'drag' on the Y axis

        End If

    End If

End Sub

Private Sub picLargeBoat_Rotate(ByVal sender As System.Object, ByVal e As System.Windows.Forms.MouseEventArgs) Handles picXLrgBoat2.MouseWheel, picXLrgBoat1.MouseWheel, picSmlBoat2.MouseWheel, picSmlBoat1.MouseWheel, picMedBoat4.MouseWheel, picMedBoat3.MouseWheel, picMedBoat2.MouseWheel, picMedBoat1.MouseWheel, picLrgBoat2.MouseWheel, picLrgBoat1.MouseWheel 'Triggers this sub when the mouse wheel is used (either up or down) whilst hovering over a boat

    With sender

        If SetupComplete = False Then

            CheckIfRotated(sender) 'Calls a previously created module with the parameter sender, in this case checking if sender is rotated

            Dim Width As Integer = .Size.Width 'Creates a local variable (local to this sub) with the value of the width of the sender
            Dim Height As Integer = .Size.Height 'Creates a local variable (local to this sub) with the value of the height of the sender

            .Size = New System.Drawing.Size(Height, Width) 'Idea from https://msdn.microsoft.com/en-us/library/system.windows.forms.picturebox.sizemode(v=vs.110).aspx Recreates the size of the object to be swapped, so height becomes width and vice versa

            If Rotation = False Then

                Rotation = True
                .Image = My.Resources.ResourceManager.GetObject(CStr(.Size.Width / 24) & " long ship Rotation") 'Makes the image in sender the same resource
```

```
    in resources that has the name CStr(.Size.Width / 24) & " long ship Rotation" so if a 5 long ship was rotated it gets the size of the object which has been
    made specifically that size. Essentially all so the image can be changed dynamically

                Else

                    Rotation = False
                    .Image = My.Resources.ResourceManager.GetObject(CStr(.Size.Height / 24) & " long ship") 'Gets the non-rotated version of the image

                End If

            End If

        End With

    End Sub

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

        frmBattleship.Show() 'Shows/opens the form frmBattleShip
        Me.Close() 'Closes the current form

    End Sub

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

        Dim Cont As Boolean = True 'Declares a local variable that is used to see whether the game should proceed to the playing phase

        Dim BoatLocX As Integer 'Declares a local variable that stores the X position of where the next boat square will be entered into the array
        Dim BoatLocY As Integer 'Declares a local variable that stores the Y position of where the next boat square will be entered into the array

        If SecondPlacement = False Then

            For i = 1 To BoatColl1.Count 'Runs a loop for how many values there are inclusive between 1 and the number of boats in the collection while declaring
     a local variable that changes with the amount of times it has looped

                BoatColl1(i).tag = "" 'Makes the tag of the object in the collection nothing

            Next

            For i = 1 To BoatColl1.Count

                For k = 1 To BoatColl1.Count

                    If i <> k AndAlso BoatColl1(i).Bounds.intersectsWith(BoatColl1(k).Bounds) = True Then 'Tests if the object BoatColl(i) essentially overlaps
    BoatColl(k) and makes sure they aren't the same object
                        BoatColl1(k).tag = BoatColl1(k).tag & "ontop" 'Makes the tag of the one that is on top "ontop"
                        i = BoatColl1.Count 'Acts as a way to exit the for loop that encompasses this area so then it doesn't trigger more than it needs to
                        Exit For 'Leaves the current for loop
                    End If
```

```
            Next

        Next

        For i = 1 To BoatColl1.Count

            BoatLocX = (BoatColl1(i).Location.X - picGrid100.Location.X) / 24 'Sets BoatLocX to be the X grid value of where the ship was placed on the screen so the top left would be 0 rather than it's screen value
            BoatLocY = (BoatColl1(i).Location.Y - picGrid100.Location.Y) / 24 'Sets BoatLocX to be the Y grid value of where the ship was placed on the screen so the top left would be 0 rather than it's screen value

            CheckIfRotated(BoatColl1(i))

            CheckIfOutside(BoatLocY, BoatLocX, BoatColl1(i))

        Next

        For i = 1 To BoatColl1.Count()

            If BoatColl1(i).tag.contains("outside") = True Then 'Tests if the current object has "outside" in it's tag

                Cont = False
                BoatColl1(i).BackColor = Color.Red 'Gives the user feedback as to which ship has the issue
                BoatColl1(i).BringToFront() 'Makes the ship easier to see and able to be clicked on depending on what ship it is that is underneath
                MsgBox("Please put your boats inside the grid")
                Exit For

            ElseIf BoatColl1(i).tag.contains("ontop") = True Then

                Cont = False
                BoatColl1(i).BackColor = Color.Red
                BoatColl1(i).BringToFront()
                MsgBox("Please don't your boats on top of one another")
                Exit For

            End If
        Next

        If Cont = True Then 'Tests if any errors might have occured in placement

            picXLrgBoat1.Visible = False 'Makes the object invisible so the other player cannot see where they have been placed
            picLrgBoat1.Visible = False
            picMedBoat1.Visible = False
            picMedBoat2.Visible = False
            picSmlBoat1.Visible = False

            MsgBox("Switch to player 2 now")
```

```
                SecondPlacement = True

                picXLrgBoat2.Visible = True 'Allows the second player to be able to see their own ships and gives them the ability to be placed
                picLrgBoat2.Visible = True
                picMedBoat3.Visible = True
                picMedBoat4.Visible = True
                picSmlBoat2.Visible = True

            End If

            Cont = False

        Else

            For i = 1 To BoatColl2.Count

                BoatColl2(i).tag = ""

            Next

            For i = 1 To BoatColl2.Count

                For k = 1 To BoatColl2.Count

                    If i <> k AndAlso BoatColl2(i).Bounds.intersectsWith(BoatColl2(k).Bounds) = True Then
                        BoatColl2(k).tag = BoatColl2(k).tag & "ontop"
                        i = BoatColl2.Count
                        Exit For
                    End If

                Next

            Next

            For i = 1 To BoatColl2.Count

                BoatLocX = (BoatColl2(i).Location.X - picGrid200.Location.X) / 24
                BoatLocY = (BoatColl2(i).Location.Y - picGrid200.Location.Y) / 24

                CheckIfRotated(BoatColl2(i))

                CheckIfOutside(BoatLocY, BoatLocX, BoatColl2(i))

            Next

            For i = 1 To BoatColl2.Count()

                If BoatColl2(i).tag.contains("outside") = True Then
```

```
                    Cont = False
                    MsgBox("Please put your boats inside the grid")
                    BoatColl2(i).BackColor = Color.Red
                    BoatColl2(i).BringToFront()
                    Exit For

                ElseIf BoatColl2(i).tag.contains("ontop") = True Then

                    Cont = False
                    MsgBox("Please don't your boats on top of one another")
                    BoatColl2(i).BackColor = Color.Red
                    BoatColl2(i).BringToFront()
                    Exit For

                End If
            Next

            If Cont = True Then

                For i = 1 To BoatColl1.Count

                    BoatLocX = (BoatColl1(i).Location.X - picGrid100.Location.X) / 24
                    BoatLocY = (BoatColl1(i).Location.Y - picGrid100.Location.Y) / 24

                    CheckIfRotated(BoatColl1(i))

                    If Rotation = True Then

                        For k = 0 To ((BoatColl1(i).Size.Width) / 24) - 1 'Runs this loop for as many size as the current ship is so for the 5 size ship it will ↙
    run 5 times
                            Grid1(BoatLocY, BoatLocX + k) = 1 'Makes the position in the array Grid1 BoatLocY, BoatLocX "contain a ship". The + k is added so     ↙
    then it will follow along the correct rotation as they get entered into the array
                        Next

                    Else

                        For k = 0 To ((BoatColl1(i).Size.Height) / 24) - 1
                            Grid1(BoatLocY + k, BoatLocX) = 1 '+ k on the other co-ordinate as it's rotated the other way
                        Next

                    End If

                Next

                For i = 1 To BoatColl2.Count

                    BoatLocX = (BoatColl2(i).Location.X - picGrid200.Location.X) / 24
                    BoatLocY = (BoatColl2(i).Location.Y - picGrid200.Location.Y) / 24
```

```
                    CheckIfRotated(BoatColl2(i))

                    If Rotation = True Then

                        For k = 0 To ((BoatColl2(i).Size.Width) / 24) - 1
                            Grid2(BoatLocY, BoatLocX + k) = 1
                        Next

                    Else

                        For k = 0 To ((BoatColl2(i).Size.Height) / 24) - 1
                            Grid2(BoatLocY + k, BoatLocX) = 1
                        Next

                    End If

                Next

                NextTurn2P() 'Runs the module and makes the game start
                btnConfirm.Visible = False 'Makes sure that the button confirm ship placement can no longer be clicked
                SetupComplete = True
                lblGameTime.Visible = True
                tmrGameTime.Enabled = True

            End If

        End If

    End Sub

    Private Sub NextTurn2P()

        If TurnNum = 2 Or TurnNum = 3 Then 'Tests if it's either the first time it triggers or just a normal time it triggers

            picXLrgBoat2.Visible = False 'Makes who's turn it was ships invisible before the switch of seats
            picLrgBoat2.Visible = False
            picMedBoat3.Visible = False
            picMedBoat4.Visible = False
            picSmlBoat2.Visible = False

            TurnNum = 1 'Changes turn so that other things work
            MsgBox("Switch to player 1") 'Tells the user who's turn it is

            picXLrgBoat1.Visible = True 'Makes the person who's turn it is now able to see their own ships
            picLrgBoat1.Visible = True
            picMedBoat1.Visible = True
            picMedBoat2.Visible = True
            picSmlBoat1.Visible = True
```

```
        Else

            picXLrgBoat1.Visible = False
            picLrgBoat1.Visible = False
            picMedBoat1.Visible = False
            picMedBoat2.Visible = False
            picSmlBoat1.Visible = False

            TurnNum = 2
            MsgBox("Switch to player 2")

            picXLrgBoat2.Visible = True
            picLrgBoat2.Visible = True
            picMedBoat3.Visible = True
            picMedBoat4.Visible = True
            picSmlBoat2.Visible = True

        End If

    End Sub

    'Runs this sub whenever one of the spaces on the first grid is clicked
    Private Sub picGrid1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles picGrid199.Click, picGrid198.Click, picGrid197.Click,
    picGrid196.Click, picGrid195.Click, picGrid194.Click, picGrid193.Click, picGrid192.Click, picGrid191.Click, picGrid190.Click, picGrid189.Click, picGrid188.
    Click, picGrid187.Click, picGrid186.Click, picGrid185.Click, picGrid184.Click, picGrid183.Click, picGrid182.Click, picGrid181.Click, picGrid180.Click,
    picGrid179.Click, picGrid178.Click, picGrid177.Click, picGrid176.Click, picGrid175.Click, picGrid174.Click, picGrid173.Click, picGrid172.Click, picGrid171.
    Click, picGrid170.Click, picGrid169.Click, picGrid168.Click, picGrid167.Click, picGrid166.Click, picGrid165.Click, picGrid164.Click, picGrid163.Click,
    picGrid162.Click, picGrid161.Click, picGrid160.Click, picGrid159.Click, picGrid158.Click, picGrid157.Click, picGrid156.Click, picGrid155.Click, picGrid154.
    Click, picGrid153.Click, picGrid152.Click, picGrid151.Click, picGrid150.Click, picGrid149.Click, picGrid148.Click, picGrid147.Click, picGrid146.Click,
    picGrid145.Click, picGrid144.Click, picGrid143.Click, picGrid142.Click, picGrid141.Click, picGrid140.Click, picGrid139.Click, picGrid138.Click, picGrid137.
    Click, picGrid136.Click, picGrid135.Click, picGrid134.Click, picGrid133.Click, picGrid132.Click, picGrid131.Click, picGrid130.Click, picGrid129.Click,
    picGrid128.Click, picGrid127.Click, picGrid126.Click, picGrid125.Click, picGrid124.Click, picGrid123.Click, picGrid122.Click, picGrid121.Click, picGrid120.
    Click, picGrid119.Click, picGrid118.Click, picGrid117.Click, picGrid116.Click, picGrid115.Click, picGrid114.Click, picGrid113.Click, picGrid112.Click,
    picGrid111.Click, picGrid110.Click, picGrid109.Click, picGrid108.Click, picGrid107.Click, picGrid106.Click, picGrid105.Click, picGrid104.Click, picGrid103.
    Click, picGrid102.Click, picGrid101.Click, picGrid100.Click

        If Animation = False Then 'Checks whether the an animation is in process

            If TurnNum = 2 Then 'Makes sure that it's not player 1 trying to fire at themself

                With sender

                    Dim ClickPosX As Integer = (.Location.X - picGrid100.Location.X) / 24 'Makes ClickPosX the X grid value version of where was clicked

                    Dim ClickPosY As Integer = (.Location.Y - picGrid100.Location.Y) / 24 'Makes ClickPosY the Y grid value version of where was clicked

                    If .tag <> "fired on" Then 'Tests if the player has already tried to shoot there

                        Shot(.Location.X, .Location.Y) 'Calls the shot sub with the parameters of the location of the sender
```

```
                    If Grid1(ClickPosY, ClickPosX) = 1 Then 'Tests if it was a hit

                        Explode(sender) 'Runs the explode module with the paramter sender
                        .BackColor = Color.Red 'shows the user that they got a hit in traditional battleship colours
                        MsgBox("Hit")
                        .BringToFront() 'Provides feedback that it was a hit

                        Dim MouseX As Integer = (MousePosition.X - Me.Location.X) 'Sets MouseX to be the position of the mouse in respect to the form as otherwise you get the mouse position on the screen
                        Dim MouseY As Integer = (MousePosition.Y - Me.Location.Y - TitlebarHeight)

                        Dim BoatNumHit As Integer 'Makes a local variable to store which ship was hit

                        For i = 1 To 5

                            If (MouseX > BoatColl1(i).Left And MouseX < BoatColl1(i).Right And MouseY > BoatColl1(i).Top And MouseY < BoatColl1(i).Bottom) Then 'Tests which ship the mouseposition with respect to the form was in when the hit happened

                                BoatNumHit = i - 1 'Sets boatnumhit to be the ships respective spot for use in a real array
                                Exit For

                            End If

                        Next

                        CurrentHits1(BoatNumHit) += 1 'Makes the current hit count of that particular ship + 1

                        If CurrentHits1(BoatNumHit) = MaxHits(BoatNumHit) Then 'Tests if that hit has sunk the ship

                            MsgBox(MaxHits(BoatNumHit) & " size ship sunk!") 'Tells the player what ship they just sunk
                            Player1ShipsSunk = Player1ShipsSunk + 1 'Adds to the count of how many ships have been sunk
                            CurrentHits1(BoatNumHit) += 1 'Makes the current hits no longer = maxhits so then it doesn't keep on reading out that the ship has been sunk, a bit of a cheat around making new arrays and variables

                        End If

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

                            MsgBox("Congratulations! Player 2 has won!") 'Provides feedback showing who won
                            frmBattleship.Show()
                            Me.Close()

                        End If

                    Else

                        .BackColor = Color.White 'Feedback to show a miss
                        MsgBox("Miss")
```

```
                    End If

                    NextTurn2P()

                    .tag = "fired on" 'Makes the square that was click have the tag fired on so it cannot be fired on twice

                Else

                    MsgBox("You cannot fire at the same spot twice") 'Tells the user their error of input

                End If

            End With

        End If

    End If

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

        If TurnNum = 1 Then

            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 MouseX As Integer = (MousePosition.X - Me.Location.X)
                            Dim MouseY As Integer = (MousePosition.Y - Me.Location.Y - TitlebarHeight)

                            Dim BoatNumHit As Integer

                            For i = 1 To 5

                                If (MouseX > BoatColl2(i).Left And MouseX < BoatColl2(i).Right And MouseY < BoatColl2(i).Bottom And MouseY > BoatColl2(i).Top)
    Then 'Still doesnt work

                                    BoatNumHit = i - 1
                                    Exit For

                                End If

                            Next

                            CurrentHits2(BoatNumHit) += 1

                            If CurrentHits2(BoatNumHit) = MaxHits(BoatNumHit) Then

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

                            End If

                            If Player2ShipsSunk = 5 Then

                                MsgBox("Congratulations! Player 1 has won!")
                                frmBattleship.Show()
                                Me.Close()

                            End If

                        Else

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

                        End If

                        NextTurn2P()
```

```
                        .tag = "fired on"

                    Else

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

                    End If

                End With

            End If

        End If

    End Sub

    Private Sub _2_Player_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

        BoatColl1.Add(picXLrgBoat1) 'Adds an object to the collection/array
        BoatColl1.Add(picLrgBoat1)
        BoatColl1.Add(picMedBoat1)
        BoatColl1.Add(picMedBoat2)
        BoatColl1.Add(picSmlBoat1)

        BoatColl2.Add(picXLrgBoat2)
        BoatColl2.Add(picLrgBoat2)
        BoatColl2.Add(picMedBoat3)
        BoatColl2.Add(picMedBoat4)
        BoatColl2.Add(picSmlBoat2)

        For i = 0 To 4

            MaxHits(i) = BoatColl1(i + 1).Size.Height / 24 'sets the maximum hits it takes to sink a ship
            CurrentHits1(i) = 0 'sets the current hits for the ships
            CurrentHits2(i) = 0

        Next

        Randomize() 'https://msdn.microsoft.com/en-us/library/8zedbtdt.aspx 'Randomises a seed from where VB draws it's randomness otherwise the computer would ↙
    fire in the same random pattern everytime

    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

    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" 'Sets the text of the label to whatever the current time elapsed is in seconds

        TotalTime += 1 'Adds one to the time

    End Sub

    Private Sub Explode(ByVal GridSpot As Object)

        'Creates a small animation to show a hit

        Animation = True 'Makes sure the program knows not to allow turns to happen during this time

        With GridSpot

            For i = 4 To 1 Step -1 'Goes backwards from 4 to 1

                .Image = My.Resources.ResourceManager.GetObject("Explosion" & i) 'Changes image to one in the resource file
                Wait(100) 'Calls the wait sub for 100 milliseconds

            Next

            For i = 1 To 4

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

            Next

            .Image = Nothing 'Gets rid of any image thats in the box

        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) 'https://msdn.microsoft.com/en-au/library/6y3efyhx(v=vs.90).aspx Plays the sound from the resources "Cannon" in the background so then other code can still run while this is playing

            .BringToFront() 'Makes sure the shot doesn't go underneath the red/white of the squares when they've been shot at

            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 'Makes essentially a small timer that means you dont need the timer object
        sw.Start() 'Starts the time
        Do While sw.ElapsedMilliseconds < time 'While the time on the stop watch is less that the desired time
            Application.DoEvents() 'Means that other things can still happen while this is running
        Loop
        sw.Stop() 'Stops the time
    End Sub

End Class
```