TICKET BOOKER THING

The one for Capitol Theatre

Abstract

Bruh I rilly wanna go to bed.

Luca Palermo

[Email address]

System Definition

I am designing a seat booking system designed for a theatre seating layout. It should be designed such that the purchase of seats in a theatre is made an effortless process. A user should be able to select a ticket for the relevant choice of seat, including consideration of class.

The file from the notification I am reading from is CapitolTheatre-1.txt

Requirements

The program must do the following:

File Manipulation

1. Loads in an external file (CSV) that defines the seating layout
2. Loads in seat booking data from a CSV of your own design that greys out already booked seats.
3. Record the name of seat ownership once booked to an external CSV file

User Interface

1. Before loading seat booking data, provides an option in the system that will allow 'social distancing' to be turned on or off. This will allow seating to be appropriately allocated whilst allowing for only 1 seat per X square meters ie. prevents allocation of adjacent seats besides, in front or behind. This mode should grey out blocked seats and prevent users interacting with them / selecting them.
2. Allows the user to select seats for an individual or a group of specified size
3. Provides an automatically recommended seating arrangement for groups if more than 1 seat is chosen, eg. 1 row together, and then grouped but split across adjacent rows.
4. Design the layout to be suitable for display on a 1920 x 1080 screen
5. Display an appropriate seat map (showing a layout of the venue / aircraft with all seats) for user interaction.
6. Display properties of different seats, eg. Seat location, class, and price

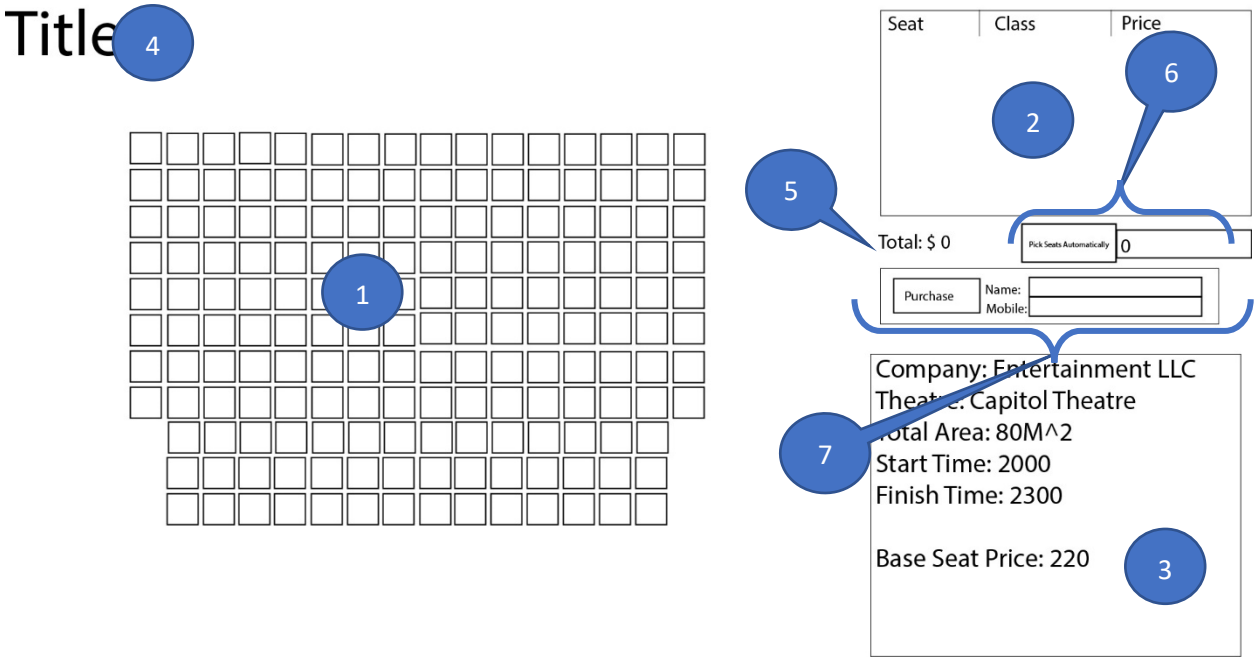
Programming

1. You must create all the string parsing routines yourself, without using inbuilt language substring and search functions.
2. Pricing for seats is based on a base price x fare class multiplier model. You can research and design your own pricing structure, but it should reflect the model defined.

Storyboard

Main window

Diagram



1. Grid of buttons generated by the program, each button represents a seat
2. List View that lists selected seats and their class and price
3. Text Box that, lists the information of the event and venue
4. Title
5. Text box that displays the sum of the seat prices
6. Elements that are responsible for picking seats automatically given a number of seats in the box next to the button
7. Used to log name and mobile numbers from the text boxes to attach to a booking after pressing the purchase things.

Design Principles

I have made a clear separation between the text, buttons etc, and the seat grid, to ensure that you expect text and input spaces on the right side and the seats on the left side. As well as that, the grid has been designed to be spaced away from the other UI elements, as well as being the largest UI element to emphasize it's importance.

Social Distancing Popup

Diagram

1. Checkbox to toggle social distancing
2. Number input box to input room area per person
3. Button that saves preferences and loads

Design Principles

Button is made the largest item on the popup to emphasize it's importance. Other than that, no other principles have been used.

Loading seat layout

Create a module or series of modules that read (load) and parse the provided sample text file layouts for a Theatre / Aircraft.

Saving bookings

Store the data collected for seat ownership into an external text file. This is to be loaded and used to indicate the availability of seats on the seat map.

Seat states are saved in a separate text file containing the following:

```
blockMap:

AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA

_________________
XAA_AAAAAAAA_AAX
XAA_AAAABAAA_AAX
XAA_AAAAAAAA_AAX
socialMap:
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA
AAA_AAAAAAAA_AAA

_________________
XAA_AAAAAAAA_AAX
XAA_AAAAAAAA_AAX
XAA_AAAAAAAA_AAX
***zzz***
```

It is two maps formatted as a 2D Char array representing the states of the buttons, the block map representing the seats purchased by customers, and the social map representing the seats that have been blocked automatically by the program due to being adjacent to purchased seats.

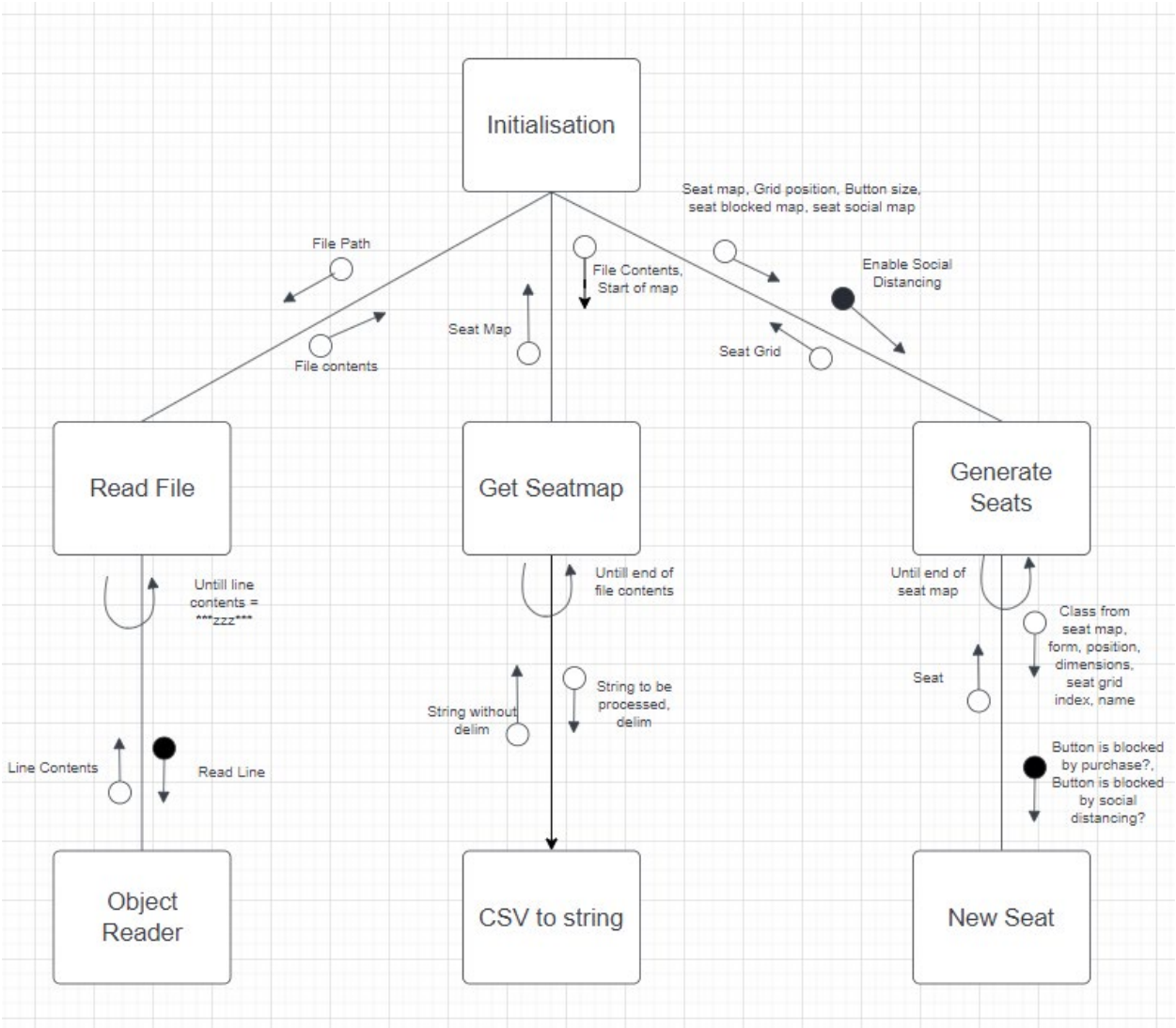
Social distancing protocol

Create the option to turn on ‘social distancing’ that will disable particular seats according to the 1 client per X square meter rule. This will also result in appropriate seat recommendations for new bookings.

There are two maps because if social distancing is disabled, the social map does not apply to the UI, whilst still being updated as new seats are purchased, this allows for people to purchase seats in bulk next to each other, so that family members can sit together, as social distancing does not apply. Also, if the number of seats attempting to be bought exceeds the remaining number of meters squared, then the program will not allow the user to purchase the seats.

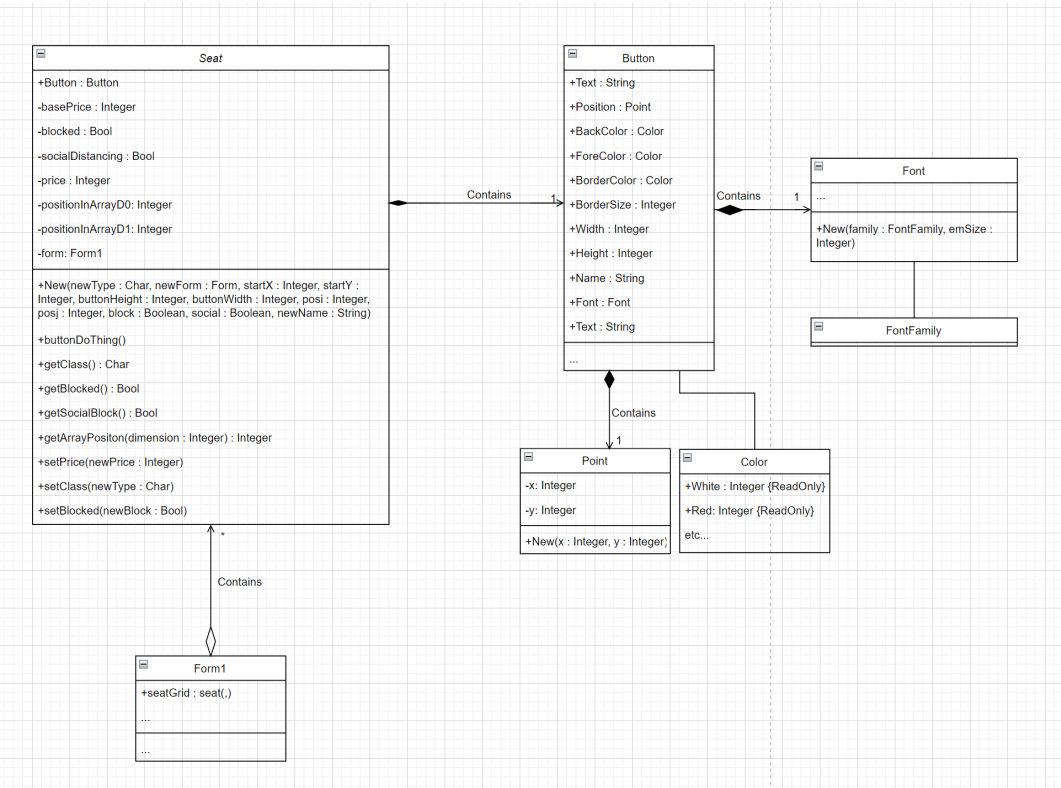
Structure Chart

Create a Structure chart for your file opening and reading routines.



Data Structure

Justify your choice of data structure for your seat information.



My programming heavily relies on arrays, they can be scaled to any application where needed, and when used in two dimensions, can represent the 2d matrix of seats, making it trivial to find adjacent seats, and add new ones.

What I wanted to do was to have each button physically contain the relevant data to its corresponding seat, class, price etc. What I decided to do to achieve this is to make a seat class that contains the button, with everything gathering the data from the button rather than an array. This is an efficient alternative to the less primitive method of making an array of buttons corresponding to an array of records containing the relevant data, this system would keep the two arrays in sync, but have them not actually related in any way. This would complicate programming down the line. Besides, I much prefer Object Oriented Programing as a method of programing anyway.

Algorithm

Represent your File Opening, and parsing routine in pseudocode

Storage of bought seats will be logged within the inbuilt

```
Public CLASS Form1
    DECLARE seatGrid As seat(,) 'Makes seatgrid global to all subroutines in form1'
    'These variables are dedicated to saving files to ram, these ones specifically save the
filepaths'
    DECLARE filePath As STRING = "C:\Users\Luca\OneDrive - Barker College\Documents\Barker
Files\Year 11\SDD\Assessment 2\Text-Files\"
    DECLARE baseFile As STRING = "CapitolTheatre-1.txt"
    DECLARE statusFile As STRING = "SeatStatus.txt"
    DECLARE fullFileBase As STRING() 'these variables contain the actual files'
    DECLARE fullFileStatusBlock As CHAR(,) 'Store the states of the seats based on IF they are
purchased and IF they are blocked by social distancing'
    DECLARE fullFileStatusSocial As CHAR(,)
    DECLARE enableSocialDistancing As BOOLEAN = TRUE
    Private SUB Form1_Visible(sender As Object, e As EventArgs) Handles Me.Shown 'on start'
        fullFileBase = readFile(filePath + baseFile) 'gets the whole file'
        DECLARE testSeatMap(,) As CHAR = getSeatMap(10, fullFileBase) 'gets the seat map'
        DECLARE tempFileStatus As STRING() = readFile(filePath + statusFile) 'gets the seat
status map'
        DECLARE maxMapLength As Integer = tempFileStatus(1).Length 'gets the max length of a
row (the top row)'
        DECLARE tempFileStatusBlock((tempFileStatus.Length - 2) / 2 - 2, maxMapLength - 1) As
CHAR 'makes the temporary arrays of the seat states'
        DECLARE tempFileStatusSocial((tempFileStatus.Length - 2) / 2 - 2, maxMapLength - 1) As
CHAR
        FOR i = 0 To tempFileStatusBlock.GetLength(0) - 1 'puts the states in the block array'
            DECLARE tempchar As CHAR() = tempFileStatus(i + 1).ToCharArray()
            FOR j = 0 To maxMapLength - 1
                tempFileStatusBlock(i, j) = tempchar(j)
            NEXT
        NEXT
        FOR i = 0 To tempFileStatusSocial.GetLength(0) - 1 'puts the states in the social
distance array'
            DECLARE tempchar As CHAR() = tempFileStatus(i + 1 +
tempFileStatusBlock.GetLength(0) + 1).ToCharArray()
            FOR j = 0 To maxMapLength - 1
                tempFileStatusSocial(i, j) = tempchar(j)
            NEXT
        NEXT
        fullFileStatusBlock = tempFileStatusBlock 'puts the temporary arrays in the global
variables'
        fullFileStatusSocial = tempFileStatusSocial
        generateSeats(testSeatMap.GetLength(0), testSeatMap.GetLength(1) - 1, 100, 100, 50, 50,
testSeatMap, fullFileStatusBlock, fullFileStatusSocial) 'generate the button grid'
    ENDSUB
    Public SUB generateSeats(newCollum As Integer, newRow As Integer, startX As Integer, startY
As Integer, buttonHeight As Integer, buttonWidth As Integer, seatMap As CHAR(,), blockMap As
CHAR(,), SocialMap As CHAR(,))
        DECLARE constX As Integer = startX 'saving original xy values before alteration'
        DECLARE consty As Integer = startY
        DECLARE buttonTemplate(newCollum + 1, newRow + 1) As seat 'makes a template of the
seats in an array'
        DECLARE tempBlock As BOOLEAN = FALSE 'starter BOOLEAN FOR blocks and social distancing'
        DECLARE tempSocial As BOOLEAN = FALSE
        DECLARE nameCounterLetter As Integer = 0
        DECLARE nameCounterNumber As Integer = 1
        DECLARE tempName As STRING
        seatGrid = buttonTemplate 'makes the array global'
        FOR i = 1 To newCollum STEP 1'FOR every collum'
            FOR j = 1 To newRow STEP 1'For every row'
                IF blockMap(i - 1, j - 1) = "B" THEN 'is it taken in the block map? IF so make
the button reflect that'
                    tempBlock = TRUE
                ELSE
                    tempBlock = FALSE
```

```
                ENDIF
                IF SocialMap(i - 1, j - 1) = "B" And enableSocialDistancing THEN 'is it taken
in the social distancing map? IF so make the button reflect that IF social distancing is
enabled'
                    tempSocial = TRUE
                ELSE
                    tempSocial = FALSE
                ENDIF
                tempName = Chr(Asc("A") + nameCounterLetter) + Str(nameCounterNumber)
                seatGrid(i, j) = New seat(seatMap(i - 1, j - 1), Me, startX, startY,
buttonHeight, buttonWidth, i, j, tempBlock, tempSocial, tempName)  'declare a new button'
                IF seatMap(i - 1, j - 1) <> "_" And seatMap(i - 1, j - 1) <> "X" THEN
                    nameCounterNumber = nameCounterNumber + 1
                ENDIF
                startX = startX + buttonWidth + 5 'increment the position of the button'
            NEXT
            nameCounterNumber = 0
            IF seatMap(i - 1, 0) <> "_" And seatMap(i - 1, 0) <> "X" THEN
                nameCounterLetter = nameCounterLetter + 1
            ENDIF
            startY = startY + buttonHeight + 5
            startX = constX
        NEXT
    ENDSUB
    Public FUNCTION readFile(filePath As STRING) As STRING()

        IF filepath exists THEN 'ensures IF file exists'
            OPENFILE filePath for INPUT
            DECLARE counter As INTEGER = 0
            DECLARE textLine As STRING
            TRY
                READ textLine from filePath
                WHILE textLine <> "***zzz***"
                    counter = counter + 1
                    READ textLine from filePath
                ENDWHILE
                Go back to top of file
                DECLARE outputTemp(counter - 1) As STRING
                counter = 0
                READ textLine from filePath
                READ textLine from filePath
                WHILE textLine <> "***zzz***"
                    outputTemp(counter) = textLine
                    counter = counter + 1
                    READ textLine from filePath
                ENDWHILE
                CLOSEFILE filePath
                RETURN outputTemp
            CATCH ex As Exception
                PRINT "it brokey"
                PRINT ex.Message
            ENDTRY
        ELSE
            PRINT "no file onoo"
        ENDIF
    ENDFUNCTION
    Public FUNCTION getSeatMap(startingFileIndex As INTEGER, file As STRING()) As CHAR(,)
        DECLARE outputx As INTEGER = file.Length - (10 + 1)
        DECLARE desiredSize As INTEGER = CSV_to_String(file(startingFileIndex), ",").Length
        DECLARE output(outputx, desiredSize) As CHAR
        FOR i = startingFileIndex TO file.Length - 1 STEP 1
            DECLARE adjustString As STRING = CSV_to_String(file(i), ",")
            WHILE desiredSize <> adjustString.Length 'should the NEXT line not be the size of
the biggest lint'
                IF desiredSize - adjustString.Length = 1 THEN 'IF the difference is 1 THEN'
                    IF adjustString.Length Mod 2 = 0 THEN 'IF the STRING length is even split
the STRING in half and pad the middle'
                        DECLARE tempcharnested As CHAR() = adjustString.ToCharArray
                        DECLARE front As STRING = ""
                        DECLARE back As STRING = ""
                        DECLARE substringLength As INTEGER = adjustString.Length / 2
```

```
                        FOR k = 0 TO adjustString / 2 - 1 STEP 1
                            front = front + tempcharnested(k)
                        NEXT
                        FOR k = adjustString / 2 TO adjustString - 1 STEP 1
                            back = back + tempcharnested(k)
                        NEXT
                        adjustString = front + "X" + back
                    ELSE 'otherwise just pad the end'
                        adjustString = adjustString + "X"
                    ENDIF
                ELSEIF desiredSize - adjustString.Length > 1 THEN 'IF the difference is more
than 1 pad front and end'
                    adjustString = "X" + adjustString + "X"
                ENDIF
            ENDWHILE
            DECLARE tempchar As CHAR() = adjustString.ToCharArray
            FOR j = 0 TO tempchar.Length - 1 STEP 1
                output(i - startingFileIndex, j) = tempchar(j)
            NEXT
        NEXT
        RETURN output
    ENDFUNCTION
ENDCLASS
Public CLASS seat
    Public button As NEW Button
    Private basePrice As INTEGER = 4000
    Private type As CHAR
    Private Blocked_for_social_distancing As BOOLEAN
    Private price As INTEGER
    Public SUB New(newType As CHAR, newForm As Form, startX As INTEGER, startY As INTEGER,
buttonHeight As INTEGER, buttonWidth As INTEGER, posi As INTEGER, posj As INTEGER, block As
BOOLEAN, social As BOOLEAN, newName As STRING) 'sets type at the start'
        DECLARE seatClass As CHAR = newType
        DECLARE socialDistancing As BOOLEAN = block
        DECLARE positionInArrayD0 As INTEGER = posi - 1
        DECLARE positionInArrayD1 As INTEGER = posj - 1
        DECLARE form = newForm
        DECLARE blocked As BOOLEAN = block
        socialDistancing = social

        WITH button
            IF seatClass = "a" THEN 'sets visual style and price of seats by type'
                .BackColor = Color.Aqua
                .ForeColor = Color.Blue
                This.price = basePrice * 1.5
            ELSEIF seatClass = "b" THEN
                .BackColor = Color.Green
                .ForeColor = Color.White
                This.price = basePrice * 1
            ELSEIF seatClass = "c" THEN
                .BackColor = Color.Yellow
                .ForeColor = Color.DarkGoldenrod
                This.price = basePrice * 0.8
            ELSEIF seatClass = "d" THEN
                .BackColor = Color.Orange
                .ForeColor = Color.White
                This.price = basePrice * 0.6
            ELSEIF seatClass = "e" THEN
                .BackColor = Color.Red
                .ForeColor = Color.DarkRed
                This.price = basePrice * 0.4
            ELSEIF seatClass = "_" THEN
                .BackColor = Color.Gray
                .ForeColor = Color.Black
                This.price = basePrice * 1000
                .Enabled = FALSE
            ELSEIF seatClass = "X" THEN
                'prevents end'
            ELSE 'notifies IF character it finds is incompatable'
                MsgBox("Error: Unrecognised Seat Type: " + seatClass)
                Application.Exit()
```

```
            ENDIF 'changes appearance of button and establishes price'
            IF socialDistancing THEN 'changes appearance and makes button disabled when blocked
or purchaced'
                .Enabled = FALSE
                .BackColor = Color.DarkGray
                .Text = "S"
            ELSEIF blocked THEN
                .Enabled = FALSE
                .BackColor = Color.LightGray
                .Text = "N"
            ELSE
                .Text = CHAR.ToUpper(seatClass)
            ENDIF
            .FlatStyle = FlatStyle.Flat 'add boarder'
            .FlatAppearance.BorderColor = Color.White
            .FlatAppearance.BorderSize = 2
            .Width = buttonHeight 'add dimensions'
            .Height = buttonWidth
            .Location = New Point(startX, startY) 'put button in location'
            .Name = newName
            .Font = NEW Font(.Font.FontFamily, 9) 'resize font'
            IF seatClass = "_" THEN
                .Text = "_"
            ELSE
                .Text = .Name
            ENDIF
            IF seatClass <> "X" THEN
                form.Controls.Add(button)
            ENDIF

            AddHandler .Click, AddressOf buttonDoThing 'add its handle'
            'generate the button'
        ENDWITH
    ENDSUB
    Public SUB buttonDoThing(Sender As Object, e As EventArgs) 'subroutine that'
        form.seatSelection(This, positionInArrayD0, positionInArrayD1)
    ENDSUB
    'Accessors'
    Public FUNCTION getClass() As CHAR
        RETURN type
    ENDFUNCTION
    Public FUNCTION getBlocked() As BOOLEAN
        RETURN Blocked_for_social_distancing
    ENDFUNCTION
    Public FUNCTION getPrice() As INTEGER
        RETURN price
    ENDFUNCTION
    Public FUNCTION getSocialBlock() As INTEGER
        RETURN socialDistancing
    ENDFUNCTION
    'Mutators'
    Public SUB setClass(newType As CHAR)
        type = newType
    ENDSUB
    Public SUB setBlocked(newBlock As BOOLEAN)
        Blocked_for_social_distancing = newBlock
    ENDSUB
    Public SUB setPrice(newPrice As INTEGER) 'probably not going TO be used'
        price = newPrice
    ENDSUB
    Public SUB setSocialBlock(newBlock As BOOLEAN)
        socialDistancing = newBlock
    ENDSUB
ENDCLASS
```

Errors

Identify and describe the different types of errors encountered during the development

of your software solution.

Seats inheriting a button does not work, so the seat class contains a value

Originally, I wanted to create a class that inherited the button class, such that it was a button that had additional attributes. Written as follows:

Buttons do not inherit seat methods

```
Public CLASS seat
    Inherits Button
    ...
ENDCLASS
```

However, this is not how inheritance works, so it ended up causing a logic error without a runtime error, so failing that, I instead made it such that the seat contained a button, achieving the same outcome:

```
Public CLASS seat
    Public button As NEW Button
    ...
ENDCLASS
```

Initially I did not adhere to encapsulation

I had made a habit of making the seat's button public, and having the form access the button's attributes directly

```
nameTest.Text = sender.button.Name
typeTest.Text = "type: " + sender.getClass()
priceTest.Text = "price: " + Str(sender.getPrice())
If sender.button.FlatAppearance.BorderColor = Color.Red Then 'marks seats as
selected
     sender. button.FlatAppearance.BorderColor = Color.White
```

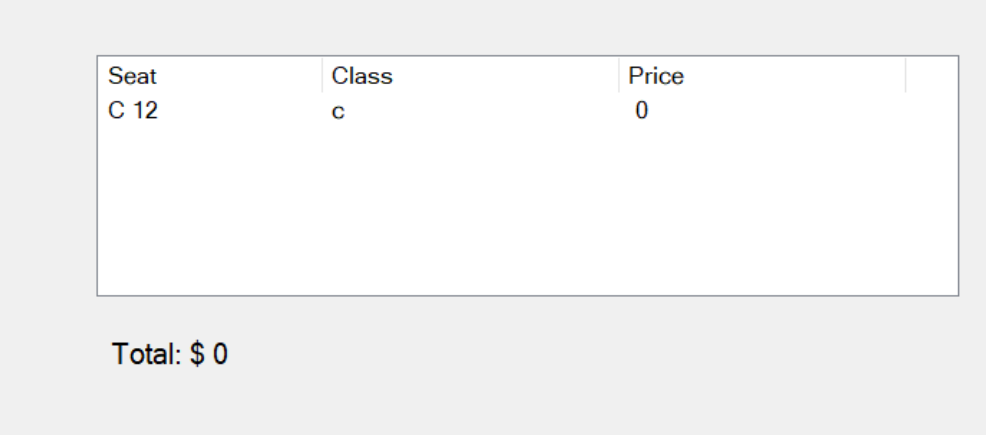
This does not adhere to encapsulation, and will make debugging difficult in the long run, so I transitioned to better encapsulation

```
nameTest.Text = sender.getName()
typeTest.Text = "type: " + sender.getClass()
priceTest.Text = "price: " + Str(sender.getPrice())
If sender.getBorder = Color.Red Then 'marks seats as selected
    sender.setBorder(Color.White)
```

Program will not write to basePrice

I wanted the program to grab the base seat price from the file, but it would not write to the seats, so all the seats cost nothing.

```
Erase tempCharRemove
tempCharRemove = "Base Seat Price: ".ToCharArray
Erase tempCharLine
tempCharLine = fullFileBase(6).ToCharArray
tempString = ""
For i = tempCharRemove.Length To tempCharLine.Length - 1 ...
basePrice = Convert.ToInt32(tempString)
```



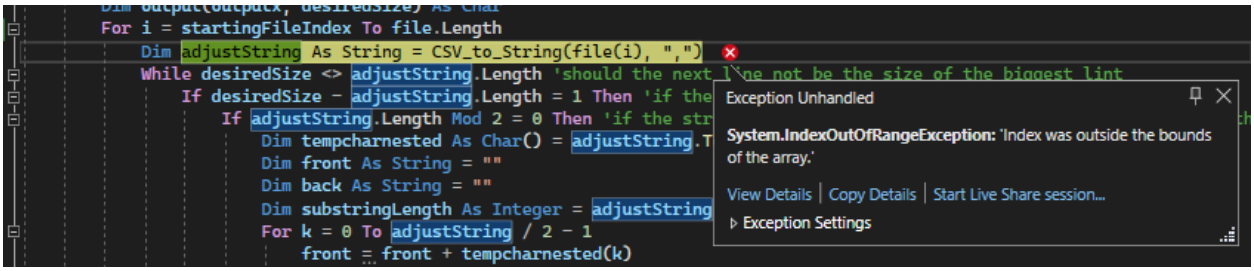
I then found that it was because I had made the seats before the base price was found, so the seats inherited a base price of 0 and afterwards base price was set, I can just move the seat generation to the front of the subroutine to fix it.

Get seat map For loop crashing

When I ran the sub routine to get the seat map, the program crashed. Here is the code:

```
Public Function getSeatMap(startingFileIndex As Integer, file As String()) As Char(,)
    Dim outputx As Integer = file.Length - (10 + 1)
    Dim desiredSize As Integer = CSV_to_String(file(startingFileIndex), ",").Length
    Dim output(outputx, desiredSize) As Char
    For i = startingFileIndex To file.Length
        Dim adjustString As String = CSV_to_String(file(i), ",")
        While desiredSize <> adjustString.Length 'should the next line not be the size of the bigges
            If desiredSize - adjustString.Length = 1 Then 'if the difference is 1 then
                If adjustString.Length Mod 2 = 0 Then 'if the string length is even split the string
                    Dim tempcharnested As Char() = adjustString.ToCharArray
                    Dim front As String = ""
                    Dim back As String = ""
                    Dim substringLength As Integer = adjustString.Length / 2
                    For k = 0 To adjustString / 2 - 1
                        front = front + tempcharnested(k)
                    Next
                    For k = adjustString / 2 To adjustString - 1
                        back = back + tempcharnested(k)
                    Next
                    adjustString = front + "X" + back
                Else 'otherwise just pad the end
                    adjustString = adjustString + "X"
                End If
            ElseIf desiredSize - adjustString.Length > 1 Then 'if the difference is more than 1 pad
                adjustString = "X" + adjustString + "X"
            End If
        End While
        Dim tempchar As Char() = adjustString.ToCharArray
        For j = 0 To tempchar.Length - 1 'to stop vbnull char from entering the string
            output(i - startingFileIndex, j) = tempchar(j)
        Next
    Next
    Return output
```

And here is the error:



It indicated that a logic error had caused a runtime error. Specifically, because I exceeded the bounds of the array. This was because the last index in an array is one less than its length, I just must remove 1 from the end of the for loop.

```
Dim output(outputx, desiredSize) As Char
For i = startingFileIndex To file.Length - 1
    Dim adjustString As String = CSV_to_String(
```

Not Operator not working

I was having trouble with checking if the purchase list was not empty, and the name input was not empty, but I was encountering a syntax error:

```
    End If 'checks if input is number and 10 chars long
    If purchaseList.Count! = 0 And nameInput.Text!= "" And mobileCheck Then
        If occupiedSeatNumber < venueCapacity Or enableSocialDistancing = False Then
            Dim tempSeatArray As seat() = purchaseList.ToArray
```

With a little bit of research, I had found that I had typed down the syntax for the not operator from python (being !=) as opposed to the Visual basic syntan (being <>). So, I replaced the syntax and the code worked.

```
    End If 'checks if input is number and 10 chars long
    If purchaseList.Count <> 0 And nameInput.Text <> "" And mobileCheck Then
        If occupiedSeatNumber < venueCapacity Or enableSocialDistancing = False Then
            Dim tempSeatArray As seat() = purchaseList.ToArray
```

Manual

The program has few requirements to be set up, but it's important to drag the SeatBookerLogFiles folder into the documents folder in your user folder.

To reset the states of the seats, you just need to turn all the Bs into As in SeatStatus.txt before you run the program