Maintenance

Contents

[List of procedures 3](#_Toc355130517)

[frmAdmin 3](#_Toc355130518)

[Private Sub btnalgorithm\_Click 3](#_Toc355130519)

[Private Sub btnBack\_Click 3](#_Toc355130520)

[Private Sub btnDay\_Click 3](#_Toc355130521)

[Private Sub btnImport\_Click 3](#_Toc355130522)

[Private Sub btnResent\_Click 5](#_Toc355130523)

[frmAvailability 5](#_Toc355130524)

[Private Sub btnExit\_Click 5](#_Toc355130525)

[Private Sub chklst\_ItemCheck 6](#_Toc355130526)

[Private Sub frmAvailability\_Load 12](#_Toc355130527)

[frmDaySettings 13](#_Toc355130528)

[Private Sub btnBack\_Click 13](#_Toc355130529)

[Private Sub cmbDay\_SelectedIndexChanged 13](#_Toc355130530)

[Private Sub cmbEnd\_SelectedIndexChanged 13](#_Toc355130531)

[Private Sub cmbNdays\_SelectedIndexChanged 14](#_Toc355130532)

[Private Sub cmbStart\_SelectedIndexChanged 14](#_Toc355130533)

[Private Sub frmDaySettings\_Load 15](#_Toc355130534)

[Private Sub rad10min\_CheckedChanged 15](#_Toc355130535)

[Private Sub rad5min\_CheckedChanged 16](#_Toc355130536)

[Public Sub Secondhalf 16](#_Toc355130537)

[frmStart 17](#_Toc355130538)

[Private Sub btnadmin\_Click 17](#_Toc355130539)

[Private Sub btnAvailability\_Click 17](#_Toc355130540)

[Private Sub btnExit\_Click 18](#_Toc355130541)

[Private Sub frmStart\_Load 18](#_Toc355130542)

[mod1 20](#_Toc355130543)

[Public Function Getappointmentrec 20](#_Toc355130544)

[Public Function GetDay 20](#_Toc355130545)

[Public Function Getlesson 20](#_Toc355130546)

[Public Function GetStaff 21](#_Toc355130547)

[Public Function GetStaffAV 21](#_Toc355130548)

[Public Function GetStudAV 22](#_Toc355130549)

[Public Function GetStudent 22](#_Toc355130550)

[Public Sub importLessonsStudent 22](#_Toc355130551)

[Public Sub importLessonStaff 23](#_Toc355130552)

[Public Sub ImportStaff 24](#_Toc355130553)

[Public Sub ImportStudents 25](#_Toc355130554)

[Public Function militarytime 27](#_Toc355130555)

[Public Sub populateStartEndDaySettings 27](#_Toc355130556)

[Public Sub Putappointment 28](#_Toc355130557)

[Public Sub Putday 28](#_Toc355130558)

[Public Sub Putlesson 29](#_Toc355130559)

[Public Sub PutStaff 29](#_Toc355130560)

[Public Sub PutStaffAv 29](#_Toc355130561)

[Public Sub PutStudAv 30](#_Toc355130562)

[Public Sub PutStudent 30](#_Toc355130563)

[Public Sub SendEmails2 30](#_Toc355130564)

[Public Sub sendemailspart1 31](#_Toc355130565)

[Public Sub TheSortingAlogorithm 32](#_Toc355130566)

[List of variables 37](#_Toc355130567)

[Global variables 37](#_Toc355130568)

[Module-level variables 40](#_Toc355130569)

[Data type summary 41](#_Toc355130570)

[Glossary 42](#_Toc355130571)

# List of procedures

## frmAdmin

### Private Sub btnalgorithm\_Click

This is a subroutine that calls “thesortingAlgortithm()”, which is a subroutine that generates the appointments. This is triggered by the user clicking on btnalgorithm.

It is used to generate appointments

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

'calls the sorting algortithm which is stored in mod1

Call TheSortingAlogorithm()

End Sub

### Private Sub btnBack\_Click

This is a subroutine that is triggered by clicking on btnBack and it closes the current form and opens frmStart

This is used to leave the form and go back to the prior one.

Private Sub btnBack\_Click(sender As System.Object, e As System.EventArgs) Handles

btnBack.Click

'opens form start and close the admin form

frmStart.Show()

Me.Close()

End Sub

### Private Sub btnDay\_Click

This is a subroutine that is triggered by clicking on btnDay and it opens frmdaysettings

This is used to go to the day settings form

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

'opnes frm daysettings

frmDaySettings.Show()

End Sub

### Private Sub btnImport\_Click

This is a subroutine that is triggered by clicking btnImport and it tries to call the 4 different import subroutines stored in mod1. If any fail due to problems with the csv then it will report an error message and end the subroutine. If it fails due to an error with the data then it will be handled by the having passed back stopimport as true at which point the subroutine will end.

This is used to import student staff and lesson information from csv’s to dat

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

'tries to inport staff. if it failes then due to safeguards in Importstaff() it must be that there is an error

'with the csv so it reports that and quits the subroutine

Try

Call ImportStaff()

Catch

MsgBox("system cannot find staff.csv", , "Error")

Exit Sub

End Try

'if there was no problems with the staff csv but there were isues with the data then an error message will have been sent

'and the varialbe stopimport will be set to true meaning that the subroutine should stop.

If stopimport = True Then

stopimport = False

Exit Sub

End If

'tries to inport students. if it failes then due to safeguards in Importstudents() it must be that there is an error

'with the csv so it reports that and quits the subroutine

Try

Call ImportStudents()

Catch

MsgBox("system cannot find student.csv", , "Error")

Exit Sub

End Try

'if there was no problems with the student csv but there were isues with the data then an error message will have been sent

'and the varialbe stopimport will be set to true meaning that the subroutine should stop.

If stopimport = True Then

stopimport = False

Exit Sub

End If

'tries to inport student lesson information. if it failes then due to safeguards in ImportLessonsStudent() it must be that there is an error

'with the csv so it reports that and quits the subroutine

Try

Call importLessonsStudent()

Catch

MsgBox("system cannot find studentclasses.csv", , "Error")

Exit Sub

End Try

'if there was no problems with the studentclass csv but there were isues with the data then an error message will have been sent

'and the varialbe stopimport will be set to true meaning that the subroutine should stop.

If stopimport = True Then

stopimport = False

Exit Sub

End If

'tries to inport staff lesson information. if it failes then due to safeguards in ImportLessonStaff() it must be that there is an error

'with the csv so it reports that and quits the subroutine

Try

Call importLessonStaff()

Catch

MsgBox("system cannot find classslots.csv", , "Error")

Exit Sub

End Try

'if there was no problems with the classslots csv but there were isues with the data then an error message will have been sent

'and the varialbe stopimport will be set to true meaning that the subroutine should stop.

If stopimport = True Then

stopimport = False

Exit Sub

End If

End Sub

### Private Sub btnResent\_Click

This is a subroutine that is triggerd by the clicking of btnresent which is a typo and should have been btnreset. It using a loop from 0 to the number of records stored in each dat file overwrites every field and record with nothing.

This is used to reset the dat files between parents evenings

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

' for each dat file it goes through and writes over every record wiht empty space

'overwrites staff.dat

For counter As Integer = 1 To Nstaff

Staff = Nothing

PutStaff(Staff, counter)

Next

'overwrites staffav.dat

For counter As Integer = 1 To NStaffAv

StaffAv = Nothing

PutStaffAv(StaffAv, counter)

Next

'overwrites student.dat

For counter As Integer = 1 To Nstudents

student = Nothing

PutStudent(student, counter)

Next

'overwrites studentav.dat

For counter As Integer = 1 To NStudAv

StudAv = Nothing

PutStudAv(StudAv, counter)

Next

'overwrites day.dat

For counter As Integer = 1 To NDay

Day = Nothing

Putday(Day, counter)

Next

'overwrites lesson.dat

For counter As Integer = 1 To Nlesson

Lesson = Nothing

Putlesson(Lesson, counter)

Next

'overwrites appointment.dat

For counter As Integer = 1 To NAppointment

Appointment = Nothing

Putappointment(Appointment, counter)

Next

End Sub

## frmAvailability

### Private Sub btnExit\_Click

This subroutine is triggered by clicking on btnexit and it closes the current form and opens frmstart

This is used to leave the availability form and go back to the last form

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

'closes the availability form and opens the start form

frmStart.Show()

Me.Close()

End Sub

### Private Sub chklst\_ItemCheck

This subroutine is triggered by the user checking the checkbox on the side of an item in the list. It checks weather the user is a student or a staff member and whether it has been checked or not. It then works out the time as a value from 0 to 277. It then finds the record and sets corrects the block and availability.

This is used to handle the changing of availbablitiy of blocks by students and teachers

Private Sub chklst\_ItemCheck(ByVal sender As Object, ByVal e As System.Windows.Forms.ItemCheckEventArgs) Handles chklstavailability.ItemCheck

'handles the chanaging of availablitity for a blcok

Dim parts() As String = Split(chklstavailability.SelectedItem, " ")

'checks if the availability is turned on or off

If chklstavailability.GetItemChecked(chklstavailability.SelectedIndex) = False Then

'availability turned on

If usertype = 1 Then

'student

If Appointmentlength = 5 Then

'appointment length 5

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 5)

'finds relevant stud av records that are in the first half hour of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 1 to their block number

StudAv.available = True

StudAv.Block += 1

PutStudAv(StudAv, StudAv.studAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 6 To ((parts(2) \ 100) + (((parts(2)) - (((parts(3)) \ 100) \* 100)) / 5) + 11)

'finds relevant stud av records that are in the second half hour of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 2 to their block number

StudAv.available = True

StudAv.Block += 2

PutStudAv(StudAv, StudAv.studAVNO)

Next

Else

'appointment length 10

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 4) Step 2

'finds relevant stud av records that are in the first third of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 1 to their block number

StudAv.available = True

StudAv.Block += 1

PutStudAv(StudAv, StudAv.studAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 6 To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 10) Step 2

'finds relevant stud av records that are in the second third of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 2 to their block number

StudAv.available = True

StudAv.Block += 2

PutStudAv(StudAv, StudAv.studAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 12 To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 16) Step 2

'finds relevant stud av records that are in the last third of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 4 to their block number

StudAv.available = True

StudAv.Block += 4

PutStudAv(StudAv, StudAv.studAVNO)

Next

End If

Else

'staff

'appointment length 5

If Appointmentlength = 5 Then

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 5)

'finds relevant stud av records that are in the first half hour of the block

For counter1 As Integer = 1 To NStaffAv

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 1 to their block number

StaffAv.Available = True

StaffAv.Block += 1

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

For counter As Integer = ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 6) To (((parts(2) \ 100) + (((parts(2)) - (((parts(3)) \ 100) \* 100)) / 5) + 11))

'finds relevant stud av records that are in the second half hour of the block

For counter1 As Integer = 1 To NStaffAv

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 2 to their block number

StaffAv.Available = True

StaffAv.Block += 2

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

Else

'appointment length 10

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 4) Step 2

'finds relevant stud av records that are in the first half hour of the block

For counter1 As Integer = 1 To NStaffAv

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 1 to their block number

StaffAv.Available = True

StaffAv.Block += 1

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 6 To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 10) Step 2

'finds relevant stud av records that are in the second half hour of the block

For counter1 As Integer = 1 To NStaffAv

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 2 to their block number

StaffAv.Available = True

StaffAv.Block += 2

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 12 To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 16) Step 2

'finds relevant stud av records that are in the third half hour of the block

For counter1 As Integer = 1 To NStaffAv

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets them to available and addes 4 to their block number

StaffAv.Available = True

StaffAv.Block += 4

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

End If

End If

Else

'box is unchecked

If usertype = 1 Then

'student

If Appointmentlength = 5 Then

'appointment lenght 5

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 5)

'finds relevant stud av records that are in the first half hour of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 1

StudAv.Block -= 1

If StudAv.Block = 0 Then

StudAv.available = False

End If

PutStudAv(StudAv, StudAv.studAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 6 To ((parts(2) \ 100) + (((parts(2)) - (((parts(3)) \ 100) \* 100)) / 5) + 11)

'finds relevant stud av records that are in the second half hour of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 2

StudAv.Block -= 2

If StudAv.Block = 0 Then

StudAv.available = False

End If

PutStudAv(StudAv, StudAv.studAVNO)

Next

Else

'appointment length 10

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 4) Step 2

'finds relevant stud av records that are in the first half hour of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 1

StudAv.Block -= 1

If StudAv.Block = 0 Then

StudAv.available = False

End If

PutStudAv(StudAv, StudAv.studAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 6 To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 10) Step 2

'finds relevant stud av records that are in the second half hour of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 2

StudAv.Block -= 2

If StudAv.Block = 0 Then

StudAv.available = False

End If

PutStudAv(StudAv, StudAv.studAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 12 To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 16) Step 2

'finds relevant stud av records that are in the third half hour of the block

For counter1 As Integer = 1 To NStudAv

GetStudAV(counter1)

If StudAv.StudNo = student.StudNO And StudAv.Appointment = counter And StudAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 4

StudAv.Block -= 4

If StudAv.Block = 0 Then

StudAv.available = False

End If

PutStudAv(StudAv, StudAv.studAVNO)

Next

End If

Else

'staff

If Appointmentlength = 5 Then

'appointment length 5

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 5)

'finds relevant stud av records that are in the first half hour of the block

For counter1 As Integer = 1 To Nstaff

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 1

StaffAv.Block -= 1

If StaffAv.Block = 0 Then

StaffAv.Available = False

End If

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 6 To ((parts(2) \ 100) + (((parts(2)) - (((parts(3)) \ 100) \* 100)) / 5) + 11)

'finds relevant stud av records that are in the second half hour of the block

For counter1 As Integer = 1 To NStaffAv

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 2

StaffAv.Block -= 2

If StaffAv.Block = 0 Then

StaffAv.Available = False

End If

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

Else

'appointment length 10

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 4) Step 2

'finds relevant stud av records that are in the first half hour of the block

For counter1 As Integer = 1 To NStaffAv

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 1

StaffAv.Block -= 1

If StaffAv.Block = 0 Then

StaffAv.Available = False

End If

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 6 To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 10) Step 2

'finds relevant stud av records that are in the second half hour of the block

For counter1 As Integer = 1 To NStaffAv

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 2

StaffAv.Block -= 2

If StaffAv.Block = 0 Then

StaffAv.Available = False

End If

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

For counter As Integer = (parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 12 To ((parts(2) \ 100) + (((parts(2)) - (((parts(2)) \ 100) \* 100)) / 5) + 16) Step 2

'finds relevant stud av records that are in the first half hour of the block

For counter1 As Integer = 1 To NStaffAv

GetStaffAV(counter1)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Appointment = counter And StaffAv.DayNO = parts(1) Then

Exit For

End If

Next

'sets there availability to false if block number is 0 after decreasing it by 4

StaffAv.Block -= 4

If StaffAv.Block = 0 Then

StaffAv.Available = False

End If

PutStaffAv(StaffAv, StaffAv.staffAVNO)

Next

End If

End If

End If

End Sub

### Private Sub frmAvailability\_Load

This is a subroutine that is triggered by frmavailability loading. It populates a label that greets the user and adds a personal touch. It then adds an item for each block to the checked list box.

This is used to greet the user and set up the form

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

'hides the start form

frmStart.Hide()

'checks weather the user is a student and files in the name correctly

If usertype = 1 Then

lblname.Text = "welcome " + student.Forename.Trim + " " + student.Surname.Trim

ElseIf usertype = 2 Then

lblname.Text = "welcome " + Staff.Forename.Trim + " " + Staff.Surname.Trim

End If

'populates the checked list box containing appointment blocks

If Appointmentlength = 5 Then

'appointmetn length is 5 minuets

For counter As Integer = 1 To NDay

'start and finish times for each day are retrived

Day = GetDay(counter)

For counter2 As Integer = 0 To (Day.finish - Day.Start) \ 6 - 2

'for each appintment an item is added contianinng the appointment number changed into a 24 hour time

chklstavailability.Items.Add("Day " + counter.ToString + " " + militarytime(Day.Start + (counter2 \* 6)) + " to " + militarytime(Day.Start + 12 + (counter2 \* 6)))

Next

Next

Else

'appointment lent is 10 minuetes

For counter As Integer = 1 To NDay

'start and finish times for each day are retrived

Day = GetDay(counter)

For counter2 As Integer = 0 To (Day.finish - Day.Start) \ 6 - 4

'for each appintment an item is added contianinng the appointment number changed into a 24 hour time

chklstavailability.Items.Add("Day " + counter.ToString + " " + militarytime(Day.Start + (counter2 \* 6)) + " to " + militarytime(Day.Start + 18 + (counter2 \* 6)))

Next

Next

End If

End Sub

## frmDaySettings

### Private Sub btnBack\_Click

this is a subroutine that is triggered by clicking on btnback. it closes the currnet form

this is used to exit the form.

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

'closes the form

Me.Close()

End Sub

### Private Sub cmbDay\_SelectedIndexChanged

This subroutine is triggerd by selecting a value in cmbday. It then calls populatestartenddaysettings.

This is used to trigger the effects of changing the day selected

Private Sub cmbDay\_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbDay.SelectedIndexChanged

'calls the subrootine that will populate cmbstart and cmbend

Call populateStartEndDaySettings()

End Sub

### Private Sub cmbEnd\_SelectedIndexChanged

This subroutine is triggerd by selecting a value in cmbend. It sets the time as the new end time for that day then clears the tiem cmb boxes and calls populatestartenddaysettings to repopulate them but updated. It then selects the current start and end times

This is used to inact any changes made to the end time of a day.

Private Sub cmbEnd\_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbEnd.SelectedIndexChanged

Dim hours As Integer = (cmbEnd.SelectedItem) \ 100

Dim minuets As Integer = ((cmbEnd.SelectedItem) - (((cmbEnd.SelectedItem) \ 100) \* 100)) / 5

'eliminates accidental loops

If change1 = True Then

change1 = False

Exit Sub

End If

'sets the end time for the day selected

Day.finish = (hours \* 12) + minuets

Day.DayNO = cmbDay.SelectedItem

Putday(Day, Day.DayNO)

'clears cmbend and cmbstart

cmbEnd.Items.Clear()

cmbStart.Items.Clear()

'calls the subroutine to populate cmbstart and cmbend

Call populateStartEndDaySettings()

change = True

'selscts the start time in cmbstart

cmbStart.SelectedIndex = Day.Start / 6

change1 = True

'selects the end time in cmbend

If Appointmentlength = 5 Then

cmbEnd.SelectedIndex = (Day.finish - Day.Start) \ 6 - 2

Else

cmbEnd.SelectedIndex = (Day.finish - Day.Start) \ 6 - 3

End If

End Sub

### Private Sub cmbNdays\_SelectedIndexChanged

This subroutine is triggered by changing the selected item in cmbNdays. It clears cmbstart end and day, then it resets the day.dat file then it calls secondhalf to populate the day altering comboboxes

It is used to populate generate the day records and incase it changed again it resets all the day information

Private Sub cmbNdays\_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbNdays.SelectedIndexChanged

'records the number of dayz

NDay = cmbNdays.SelectedItem

'clears the secondary comboboxes

cmbStart.Items.Clear()

cmbEnd.Items.Clear()

cmbDay.Items.Clear()

'resets the day settings

For counter As Integer = 1 To NDay

Day.DayNO = counter

Day.finish = 288

Day.Start = 0

Putday(Day, Day.DayNO)

Next

'calls the subroutine second half which decided weather to complete the other repacutions of changing the number of days

Call secondhalf()

End Sub

### Private Sub cmbStart\_SelectedIndexChanged

This subroutine is triggerd by selecting a value in cmbstart. It sets the time as the new start time for that day then clears the tiem cmb boxes and calls populatestartenddaysettings to repopulate them but updated. It then selects the current start and end times

This is used to inact any changes made to the start time of a day.

Private Sub cmbStart\_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbStart.SelectedIndexChanged

Dim hours As Integer = (cmbStart.SelectedItem) \ 100

Dim minuets As Integer = ((cmbStart.SelectedItem) - (((cmbStart.SelectedItem) \ 100) \* 100)) / 5

'eliminates any accidental loops

If change = True Then

change = False

Exit Sub

End If

'sets the start time for the day selected

Day.Start = (hours \* 12) + minuets

Day.DayNO = cmbDay.SelectedItem

Putday(Day, Day.DayNO)

'clears the cmbstart andc cmbend

cmbEnd.Items.Clear()

cmbStart.Items.Clear()

'calls the subroutine to populate cmbstart and cmbend

Call populateStartEndDaySettings()

change = True

'selects the start time in cmbstart

cmbStart.SelectedIndex = Day.Start / 6

change1 = True

'selects the end time in cmbend

If Appointmentlength = 5 Then

cmbEnd.SelectedIndex = (Day.finish - Day.Start) \ 6 - 2

Else

cmbEnd.SelectedIndex = (Day.finish - Day.Start) \ 6 - 3

End If

End Sub

### Private Sub frmDaySettings\_Load

This subroutine is triggerd by frmdaysettings loading. I then populates cmbndays with 1 to 255 days.

It is used to set up the basic parts of frmday settings

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

'populates the dropdown combobox with numbers 1 to 255

For counter As Integer = 1 To 255

cmbNdays.Items.Add(counter)

Next

End Sub

### Private Sub rad10min\_CheckedChanged

This subroutine is triggered by checking the 10 min radio button. Incase it has been done once day information has been input it resets the day records and calls second half to start populatingthe day settings combo boxes that decide times.

It is used to populate generate the day records and incase it changed again it resets all the day information

Private Sub rad10min\_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles rad10min.CheckedChanged

'records the new value of appintment length and checks if its ready to start hte second half

Appointmentlength = 10

'clears the secondary comboboxes

cmbStart.Items.Clear()

cmbEnd.Items.Clear()

cmbDay.Items.Clear()

'resets the day settings

For counter As Integer = 1 To NDay

Day.DayNO = counter

Day.finish = 288

Day.Start = 0

Putday(Day, Day.DayNO)

Next

'calls the subroutine second half which decided weather to complete the other repacutions of changing the appointment length

Call secondhalf()

End Sub

### Private Sub rad5min\_CheckedChanged

This subroutine is triggered by checking the 5 min radio button. Incase it has been done once day information has been input it resets the day records and calls second half to start populatingthe day settings combo boxes that decide times.

It is used to populate generate the day records and incase it changed again it resets all the day information

Private Sub rad5min\_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles rad5min.CheckedChanged

'checks if the the 5 min radio button is checked and if it is sets the appointmetn length to 5 mins

'otherwise it sets the appointment length to 10 mins

If rad5min.Checked = True Then

Appointmentlength = 5

Else

Appointmentlength = 10

End If

'clears the secondary comboboxes

cmbStart.Items.Clear()

cmbEnd.Items.Clear()

cmbDay.Items.Clear()

'resets the day settings

For counter As Integer = 1 To NDay

Day.DayNO = counter

Day.finish = 288

Day.Start = 0

Putday(Day, Day.DayNO)

Next

'calls the subroutine second half which decided weather to complete the other repacutions of changing the appointment length

Call secondhalf()

End Sub

### Public Sub Secondhalf

This subroutine is triggered by changing the appointmentlength or the number of days. It checks if there is an appointment length set and if there is a number days set. If there is it genereates the day records and makes the cmbday , cmbstart and cmbend combo boxes visible and populates cmbdy.

It used to populate the day file and allow for the editing of the time for days.

Public Sub secondhalf()

'sub to check if the appointment length and day numbers have been chosen. if so it populates the

'day.dat file

If NDay <> -1 And Appointmentlength <> -1 Then

For counter As Integer = 1 To NDay

Day.DayNO = counter

Day.Start = 0

Day.finish = 288

Putday(Day, counter)

Next

Else

Exit Sub

End If

cmbDay.Visible = True

cmbStart.Visible = True

cmbEnd.Visible = True

'populates the day selector combo box with each of the days

cmbDay.Items.Clear()

For counter = 1 To NDay

cmbDay.Items.Add(counter)

Next

End Sub

## frmStart

### Private Sub btnadmin\_Click

This subroutine is triggered by clicking btnadmin. It closes the current form and opens up frm admin

It is used to go to the admin form

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

'opens up the admin form and closes the start form

frmAdmin.Show()

Me.Hide()

End Sub

### Private Sub btnAvailability\_Click

This subroutine is triggered by clicking btnavailability. It checks If the users user type Is 1 or 2 if so it opesn up from availability.

It is used to go to the availability form

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

'though the button should only be vissible if hte user is in the system if first checks if they are tne sends them to the availablility form

If usertype = 1 Or usertype = 2 Then

frmAvailability.Show()

End If

End Sub

### Private Sub btnExit\_Click

This subroutine is triggered by clicking btnexit. It closes the current form.

It is used to close the start form and close the program.

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

'closes the form

Me.Close()

End Sub

### Private Sub frmStart\_Load

This subroutine is triggered by the loading of frmstart. It retrives the username and sets that to user. It then works out the number of each type of record.it checks the username against the staff and student records to find a match. If the user is a student their user type is set to 1 if it is staff it is set to 2. If there is no staff.dat file then if the user is in the format of a staff member they are sent to the admin form. If they did not match anything then they are told exactly that and advised to see an admin if it is wrong

It is used to set up the start form and work out who is logging on and give them the access they are permited.

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

'splits up the user data to seperate out the users username

Dim parts() As String = Split(My.User.Name, "\")

user = parts(1)

'finds the length of each dat file and if they dont exist sets it to -1

Try

Nstaff = FileLen("staff.dat") / Len(Staff)

Catch

Nstaff = -1

End Try

Try

NStaffAv = FileLen("staffav.dat") / Len(StaffAv)

Catch

NStaffAv = -1

End Try

Try

Nstudents = FileLen("student.dat") / Len(student)

Catch

Nstudents = -1

End Try

Try

NStudAv = FileLen("studav.dat") / Len(StudAv)

Catch

NStudAv = -1

End Try

Try

NAppointment = FileLen("appointments.dat") / Len(Appointment)

Catch

NAppointment = -1

End Try

Try

NDay = FileLen("day.dat") / Len(Day)

Catch

NDay = -1

End Try

Try

Nlesson = FileLen("lesson.dat") / Len(Lesson)

Catch

Nlesson = -1

End Try

'checks if the user is a student or a staff member

If Len(user) = 6 Then

'user is in the form of a students so the list of students is checked

For counter As Integer = 1 To Nstudents

student = GetStudent(counter)

'checks if the student has been found

If user = student.StudID Then

'records that the user is a student

usertype = 1

Exit For

End If

Next

Else

'user is not in the form of a student so the list of staff is checked

For counter As Integer = 1 To Nstaff

Staff = GetStaff(counter)

'checks if the user has been found

If user.ToUpper = Staff.staffID.ToUpper Then

'records that the user is a member of staff

usertype = 2

'checks if the user is an admin

If Staff.admin = True Then

'makes the admin button visible so that the admin form may be accessed

btnAvailability.Text = "Your Availability"

btnadmin.Visible = True

End If

Exit For

End If

Next

End If

'checks if the system doesnt have any staff. if so then there arnt any admins and it checks if the user is in the format of a staff member

'if so then it sends the user to the admin file to set up the system

If FileLen("staff.dat") = 0 And IsNumeric(user) = False And Len(user) = 3 Then

frmAdmin.Show()

Me.Close()

End If

'if the usesr has been given a usertype of 0 then he is not in the system so access to the other forms is blocked of by making the buttons

'invisible and it then sends an error message sayting they arent in the system and advising them to check with an admisistrator if they

'feel it is wrong

If usertype = 0 Then

btnAvailability.Visible = False

MsgBox("Your username is not recognised by the system. If this is an error please contact the it technicians.", , "ERROR")

End If

End Sub

## mod1

### Public Function Getappointmentrec

This function is triggered by being called. It has the parameter recNo which is an integer and stores the record number that the user is looking for. The function getappoinmentrec is in the structure of an appointments rec. it opens the appioment.dat file and goes along a the record number’s worth of record lengths and reads in the record then.

This is used to retrieve an appointment record by record number

'retreives a appointment record

Public Function Getappointmentrec(ByVal RecNo As Integer) As AppointmentsRec

'function for getting data from Appointments dat file

Dim Filenum As Integer = FreeFile()

Getappointmentrec = Nothing

'opens the appointments dat file

FileOpen(Filenum, "Appointments.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Appointment))

'gets data

FileGet(Filenum, Getappointmentrec, RecNo)

'closes file

FileClose(Filenum)

End Function

### Public Function GetDay

This function is triggered by being called. It has the parameter recNo which is an integer and stores the record number that the user is looking for. The function getday is in the structure of a day rec. it opens the day.dat file and goes along the record number’s worth of record lengths and reads in the record then.

This is used to retrieve a day record by record number

'retrives a day record

Public Function GetDay(ByVal RecNo As Integer) As DayRec

'function for getting data from day dat file

Dim Filenum As Integer = FreeFile()

GetDay = Nothing

'opens the day dat file

FileOpen(Filenum, "Day.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Day))

'gets data

FileGet(Filenum, GetDay, RecNo)

'closes file

FileClose(Filenum)

End Function

### Public Function Getlesson

This function is triggered by being called. It has the parameter recNo which is an integer and stores the record number that the user is looking for. The function getlesson is in the structure of an lesson rec. it opens the lesson.dat file and goes along the record number’s worth of record lengths and reads in the record then.

This is used to retrieve a lesson record by record number

'retrives a lesson record

Public Function Getlesson(ByVal RecNo As Integer) As LessonRec

'function for getting data from lesson dat file

Dim Filenum As Integer = FreeFile()

Getlesson = Nothing

'opens the lesson dat file

FileOpen(Filenum, "Lesson.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Lesson))

'gets data

FileGet(Filenum, Getlesson, RecNo)

'closes file

FileClose(Filenum)

End Function

### Public Function GetStaff

This function is triggered by being called. It has the parameter recNo which is an integer and stores the record number that the user is looking for. The function getstaff is in the structure of an staff rec. it opens the staff.dat file and goes along the record number’s worth of record lengths and reads in the record then.

This is used to retrieve a staff record by record number

'retrives a staff record

Public Function GetStaff(ByVal RecNo As Integer) As StaffRec

'function for getting data from staff dat file

Dim Filenum As Integer = FreeFile()

GetStaff = Nothing

'opens the staff dat file

FileOpen(Filenum, "Staff.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Staff))

'gets data

FileGet(Filenum, GetStaff, RecNo)

'closes file

FileClose(Filenum)

End Function

### Public Function GetStaffAV

This function is triggered by being called. It has the parameter recNo which is an integer and stores the record number that the user is looking for. The function getstaffav is in the structure of an staffav rec. it opens the staffav.dat file and goes along the record number’s worth of record lengths and reads in the record then.

This is used to retrieve a staffav record by record number

'retrives a staff availiability record

Public Function GetStaffAV(ByVal RecNo As Integer) As StaffAvRec

'function for getting data from staffav dat file

Dim Filenum As Integer = FreeFile()

GetStaffAV = Nothing

'opens the staffav dat file

FileOpen(Filenum, "StaffAV.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(StaffAv))

'gets data

FileGet(Filenum, GetStaffAV, RecNo)

'closes file

FileClose(Filenum)

End Function

### Public Function GetStudAV

This function is triggered by being called. It has the parameter recNo which is an integer and stores the record number that the user is looking for. The function getstudav is in the structure of an studav rec. it opens the studav.dat file and goes along the record number’s worth of record lengths and reads in the record then.

This is used to retrieve a studav record by record number

'retrives a student availability record

Public Function GetStudAV(ByVal RecNo As Integer) As StudAvRec

'function for getting data from studAV dat file

Dim Filenum As Integer = FreeFile()

GetStudAV = Nothing

'opens the studav dat file

FileOpen(Filenum, "StudAV.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(StudAv))

'gets data

FileGet(Filenum, GetStudAV, RecNo)

'closes file

FileClose(Filenum)

End Function

### Public Function GetStudent

This function is triggered by being called. It has the parameter recNo which is an integer and stores the record number that the user is looking for. The function getstudent is in the structure of an student rec. it opens the student.dat file and goes along the record number’s worth of record lengths and reads in the record then.

This is used to retrieve a student record by record number

' retrives a student record

Public Function GetStudent(ByVal RecNo As Integer) As StudRec

'function for getting data from student dat file

Dim Filenum As Integer = FreeFile()

GetStudent = Nothing

'opens the student dat file

FileOpen(Filenum, "Student.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(student))

'gets data

FileGet(Filenum, GetStudent, RecNo)

'closes file

FileClose(Filenum)

End Function

### Public Sub importLessonsStudent

this is a subroutine that is triggerd by being called. It opens up the visual basic text reader and reads in the csv one line at a time. It then splits up the line by the commas and puts it in to the array of strings current row. It then for each row creates a lesson record with the lesson number the current line the staff number the lesson code and the student number the studno.

It imports lesson information from csv to dat file witht eh staff information a lesson code.

Public Sub importLessonsStudent()

'opens microsoft file reader and sets the file to be read as tutor.csv

Dim TextFileReader As New Microsoft.VisualBasic.FileIO.TextFieldParser("studentclass.csv")

TextFileReader.TextFieldType = FileIO.FieldType.Delimited

TextFileReader.SetDelimiters(",")

Dim CurrentRow As String()

Dim OnRec As Integer = 0

Dim FileNum As Integer = FreeFile()

'opens the file

FileOpen(FileNum, "lesson.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Lesson))

While Not TextFileReader.EndOfData

Try

CurrentRow = TextFileReader.ReadFields()

If Not CurrentRow Is Nothing Then

OnRec = OnRec + 1

'puts data into file structure staff

With Lesson

Try

.LessonNO = OnRec

.StaffNO = CurrentRow(1)

.StudNO = CurrentRow(0)

Catch

'this is trigered if there was a problem with inputing to the lesson structure

'it stops the importing and telles the suer what has happened

stopimport = True

MsgBox("error with studentclass.csv")

Exit Sub

End Try

End With

'puts data in the lesson file structure into the lesson dat file

FilePut(FileNum, Lesson, OnRec)

End If

Catch ex As \_

Microsoft.VisualBasic.FileIO.MalformedLineException

'error in text sends error message and ends try

MsgBox("Line " & ex.Message & "is not valid and will be skipped.")

End Try

End While

Nlesson = OnRec

'sends message box notifying admin that student part of lessons have been imported and how many have been imported

MsgBox("Student half imported")

FileClose(FileNum)

TextFileReader.Dispose()

End Sub

### Public Sub importLessonStaff

this is a subroutine that is triggerd by being called. It opens up the visual basic text reader and reads in the csv one line at a time. It then splits up the line by the commas and puts it in to the array of strings current row. It then for each row finds an lesson records with a lesson code the same as that row and then replaces the records value with the staffNO.

It reads a the classslots csv file and finds and inserts the teacher for each lesson.

Public Sub importLessonStaff()

'opens microsoft file reader and sets the file to be read as classslots.csv

Dim TextFileReader As New Microsoft.VisualBasic.FileIO.TextFieldParser("classSlots.csv")

TextFileReader.TextFieldType = FileIO.FieldType.Delimited

TextFileReader.SetDelimiters(",")

Dim lastlesson As Integer = -1

Dim currentrow As String() = Nothing

Dim onrec As Integer = 0

OnRec = 0

While Not TextFileReader.EndOfData

Try

currentrow = TextFileReader.ReadFields()

If (Not currentrow Is Nothing) And (currentrow(0) <> lastlesson.ToString) Then

onrec = onrec + 1

lastlesson = currentrow(0)

For counter As Integer = 1 To Nlesson

Getlesson(counter)

Try

If Lesson.StaffNO = currentrow(0) Then

Lesson.StaffNO = currentrow(3)

End If

Catch

'this is trigered if there was a problem with inputing to the lesson structure

'it stops the importing and telles the suer what has happened

stopimport = True

MsgBox("error with classslots.csv")

Exit Sub

End Try

Putlesson(Lesson, Lesson.LessonNO)

Next

End If

Catch ex As \_

Microsoft.VisualBasic.FileIO.MalformedLineException

'error in text sends error message and ends try

MsgBox("Line " & ex.Message & "is not valid and will be skipped.")

End Try

End While

'sends message box notifying admin that the staff side of lessons have been imported

MsgBox("staff half imported")

TextFileReader.Dispose()

End Sub

### Public Sub ImportStaff

this is a subroutine that is triggerd by being called. It opens up the visual basic text reader and reads in the tutorcsv one line at a time. It then splits up the line by the commas and puts it in to the array of strings current row. It then for each row creates a staff record with the staff number the current line and the staff id and other pieces of information is from one of the different parts of the row.

It imports staff information from csv to dat file.

'imports staff into thier dat file

Public Sub ImportStaff()

'opens microsoft file reader and sets the file to be read as tutor.csv

Dim TextFileReader As New Microsoft.VisualBasic.FileIO.TextFieldParser("tutor.csv")

TextFileReader.TextFieldType = FileIO.FieldType.Delimited

TextFileReader.SetDelimiters(",")

Dim CurrentRow As String()

Dim OnRec As Integer = 0

Dim FileNum As Integer = FreeFile()

'opens the file

FileOpen(FileNum, "Staff.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Staff))

Dim parts() As String

While Not TextFileReader.EndOfData

Try

CurrentRow = TextFileReader.ReadFields()

If Not CurrentRow Is Nothing Then

OnRec = OnRec + 1

'puts data into file structure staff

With Staff

Try

.StaffNO = CurrentRow(0)

'forename and surname are saved in the same field on the parent file so need to be broken up

parts = Split(CurrentRow(1), " ")

.Surname = parts(1)

.Forename = parts(0)

.staffID = CurrentRow(2)

If CurrentRow(3) = 0 Then

.admin = False

Else : .admin = True

End If

Catch

'this is trigered if there was a problem with inputing to the staff structure

'it stops the importing and telles the suer what has happened

stopimport = True

MsgBox("error with staff.csv")

Exit Sub

End Try

End With

'puts data in file structure staff into the staff dat file

FilePut(FileNum, Staff, OnRec)

End If

Catch ex As \_

Microsoft.VisualBasic.FileIO.MalformedLineException

'error in text sends error message and ends try

MsgBox("Line " & ex.Message & "is not valid and will be skipped.")

End Try

End While

Nstaff = OnRec

'sends message box notifying student that staff have been imported and how many have been

MsgBox(Nstaff & " Staff imported")

FileClose(FileNum)

TextFileReader.Dispose()

End Sub

### Public Sub ImportStudents

this is a subroutine that is triggerd by being called. It opens up the visual basic text reader and reads in the studentscsv one line at a time. It then splits up the line by the commas and puts it in to the array of strings current row. It then for each row creates a student record with the staff number the current line and the student id and other pieces of information is from one of the different parts of the row.

It imports student information from csv to dat file.

'reading csv files and creating dat files

'imports the students into thier dat file

Public Sub ImportStudents()

'opnes up file reader and sets it to read students.csv the file in which the student data is stored

Dim TextFileReader As New Microsoft.VisualBasic.FileIO.TextFieldParser("students.csv")

TextFileReader.TextFieldType = FileIO.FieldType.Delimited

TextFileReader.SetDelimiters(",")

Dim CurrentRow As String()

Dim OnRec As Integer = 0

Dim FileNum As Integer = FreeFile()

'opens file

FileOpen(FileNum, "Student.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(student))

While Not TextFileReader.EndOfData

Try

CurrentRow = TextFileReader.ReadFields()

If Not CurrentRow Is Nothing Then

OnRec = OnRec + 1

'puts data into the studet structure

With student

Try

.StudNO = CurrentRow(0)

.StudID = CurrentRow(1)

.Surname = CurrentRow(2)

.Forename = CurrentRow(3)

.Year = CurrentRow(4)

Catch

'this is trigered if there was a problem with inputing to the student structure

'it stops the importing and telles the suer what has happened

stopimport = True

MsgBox("error with student.csv")

Exit Sub

End Try

End With

'puts data from the student structure into the student dat file

FilePut(FileNum, student, OnRec)

End If

Catch ex As \_

Microsoft.VisualBasic.FileIO.MalformedLineException

'if error then error message is sent and try ends

MsgBox("Line " & ex.Message & "is not valid and will be skipped.")

End Try

End While

'message box is sent saying that the students are imported and how many

Nstudents = OnRec

MsgBox(NStudents & " students imported")

'file is closed

FileClose(FileNum)

TextFileReader.Dispose()

End Sub

### Public Function militarytime

This is a function that is triggered by being called. It has a perameter timeNO which is an appointment time as a number from 0 to 287. It works out how many 12’s go into time number and that is the number of hours. It then fills out that string with 0’s if needed to make it 2 characters. For the minuets it does the same except it uses the remmander after divideing by 12 and multiplies it by 5. The string “ hours” and “minuets” are then joined together and returned

It converts a number into a time in 24 hour time.

'fucntion that changes a number from 0 to 287 into its coresponidng 24 hour clock time

Public Function militarytime(ByVal timeNO As Integer) As String

Dim hours As String

Dim minuets As String

'works out how many hours there are

hours = (timeNO \ 12).ToString

'puts in the place filler zeroes to keep it 2 characters

If Len(hours) = 1 Then

hours = "0" + hours

ElseIf Len(hours) = 0 Then

hours = "00"

End If

'works out how many minuets remain not counting the hours

minuets = (timeNO - ((timeNO \ 12) \* 12))

minuets = minuets \* 5

'puts in the place filling zeroes if need to keep it to 2 characters

If Len(minuets) = 1 Then

minuets = "0" + minuets

ElseIf Len(minuets) = 0 Then

minuets = "00"

End If

'puts the 2 halves to gether to be returned

militarytime = hours + minuets

End Function

### Public Sub populateStartEndDaySettings

This is a subroutine that is triggered by being called. It checks the appointment length and gets the dayrecord and populates cmbstart from 0 to day.finish less 12 if appoiment length is 5 or 18 if 10. It then populates cmbstart from cmb.start +12 or 18 depending on length to 288. For each time it is passed through military time to give a time the user can comprehend.

It populates cmbstart and cmbend on frmdaysettings.

Public Sub populateStartEndDaySettings()

'populates cmbstart with the times at the required appiontment length apart

Day = GetDay(frmDaySettings.cmbDay.SelectedItem)

If Appointmentlength = 5 Then

'populates cmbstart with 5 min appointments slots

For counter As Integer = 0 To (Day.finish - 12) Step 6

'each slot contained between the beginin and the finsish time is converted into 24hour

'military style time

frmDaySettings.cmbStart.Items.Add(militarytime(counter))

Next

'populates the cmbEnd list witht the available times

For counter As Integer = (Day.Start + 12) To 288 Step 6

'each slot contained between the beginin and the finsish time is converted into 24hour

'military style time

frmDaySettings.cmbEnd.Items.Add(militarytime(counter))

Next

Else

'populates cmbstart with 10 min appointments slots

For counter As Integer = 0 To (Day.finish - 18) Step 6

'each slot contained between the beginin and the finsish time is converted into 24hour

'military style time

frmDaySettings.cmbStart.Items.Add(militarytime(counter))

Next

'populates the cmbEnd list witht the available times

For counter As Integer = (Day.Start + 18) To 288 Step 6

'each slot contained between the beginin and the finsish time is converted into 24hour

'military style time

frmDaySettings.cmbEnd.Items.Add(militarytime(counter))

Next

End If

End Sub

### Public Sub Putappointment

This subroutine is triggered by being called. It has the perameter editedappointment and recNO. It opens the dat file goes along to the right record and overwrites the old record with editedappointment. It then closes the file.

It puts writes an edited record saving the changes.

'overwrites an appointment onto the appointment dat file

Public Sub Putappointment(ByVal Editedappointment As AppointmentsRec, ByVal RecNo As Integer)

'sub for putting data into the appointment dat file

Dim Filenum As Integer = FreeFile()

'opens appointment dat file

FileOpen(Filenum, "appointments.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Appointment))

'puts data into appointment dat file

FilePut(Filenum, Editedappointment, RecNo)

'closes the appointment dat file

FileClose(Filenum)

End Sub

### Public Sub Putday

This subroutine is triggered by being called. It has the perameter editedday and recNO. It opens the dat file goes along to the right record and overwrites the old record with editedday. It then closes the file.

It puts writes an edited record saving the changes.

'overwrites an Day onto the day dat file

Public Sub Putday(ByVal Editedday As DayRec, ByVal RecNo As Integer)

'sub for putting data into the day dat file

Dim Filenum As Integer = FreeFile()

'opens day dat file

FileOpen(Filenum, "day.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Day))

'puts data into day dat file

FilePut(Filenum, Editedday, RecNo)

'closes day dat file

FileClose(Filenum)

End Sub

### Public Sub Putlesson

This subroutine is triggered by being called. It has the perameter editedlesson and recNO. It opens the dat file goes along to the right record and overwrites the old record with editedlesson. It then closes the file.

It puts writes an edited record saving the changes.

'overwrites an lesson onto the lesson dat file

Public Sub Putlesson(ByVal Editedlesson As LessonRec, ByVal RecNo As Integer)

'sub for putting data into the lesson dat file

Dim Filenum As Integer = FreeFile()

'opens lesson dat file

FileOpen(Filenum, "Lesson.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Lesson))

'puts data into lesson dat file

FilePut(Filenum, Editedlesson, RecNo)

'closes lesson dat file

FileClose(Filenum)

End Sub

### Public Sub PutStaff

This subroutine is triggered by being called. It has the perameter editedstaff and recNO. It opens the dat file goes along to the right record and overwrites the old record with editedstaff. It then closes the file.

It puts writes an edited record saving the changes.

'overwrites an staff onto the staff dat file

Public Sub PutStaff(ByVal EditedStaff As StaffRec, ByVal RecNo As Integer)

'sub for putting data into the staff dat file

Dim Filenum As Integer = FreeFile()

'opens staff dat file

FileOpen(Filenum, "Staff.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(Staff))

'puts data into staff dat file

FilePut(Filenum, EditedStaff, RecNo)

'closes staff dat file

FileClose(Filenum)

End Sub

### Public Sub PutStaffAv

This subroutine is triggered by being called. It has the perameter editedstaffav and recNO. It opens the dat file goes along to the right record and overwrites the old record with editedstaffav. It then closes the file.

It puts writes an edited record saving the changes.

'overwrites an staffAv onto the staffAv dat file

Public Sub PutStaffAv(ByVal EditedStaffAv As StaffAvRec, ByVal RecNo As Integer)

'sub for putting data into the staffAv dat file

Dim Filenum As Integer = FreeFile()

'opens staffAv dat file

FileOpen(Filenum, "StaffAv.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(StaffAv))

'puts data into staffAv dat file

FilePut(Filenum, EditedStaffAv, RecNo)

'closes staffAv dat file

FileClose(Filenum)

End Sub

### Public Sub PutStudAv

This subroutine is triggered by being called. It has the perameter editedstudav and recNO. It opens the dat file goes along to the right record and overwrites the old record with editedstudav. It then closes the file.

It puts writes an edited record saving the changes.

'overwrites an studavrec onto the studav dat file

Public Sub PutStudAv(ByVal EditedStudAv As StudAvRec, ByVal RecNo As Integer)

'sub for putting data into the studAv dat file

Dim Filenum As Integer = FreeFile()

'opens studAv dat file

FileOpen(Filenum, "StudAv.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(StudAv))

'puts data into studAv dat file

FilePut(Filenum, EditedStudAv, RecNo)

'closes studAv dat file

FileClose(Filenum)

End Sub

### Public Sub PutStudent

This subroutine is triggered by being called. It has the perameter editedstudent and recNO. It opens the dat file goes along to the right record and overwrites the old record with editedstudent. It then closes the file.

It puts writes an edited record saving the changes.

'overwrites an student onto the student dat file

Public Sub PutStudent(ByVal EditedStudent As StudRec, ByVal RecNo As Integer)

'sub for putting data into the student dat file

Dim Filenum As Integer = FreeFile()

'opens student dat file

FileOpen(Filenum, "Student.dat", OpenMode.Random, OpenAccess.Default, OpenShare.Default, Len(student))

'puts data into student dat file

FilePut(Filenum, EditedStudent, RecNo)

'closes student dat file

FileClose(Filenum)

End Sub

### Public Sub SendEmails2

This subroutine is triggerd by being called. It has 8 parametres. Fromaddress which is the address it is being sent from subject which is the subject of the email. Body which is the main text, username and password which are for logging onto the account the emails are being sent from. Recipant the recivers email. Server and port are optional and will likely not used. It links to an smtpsever in this case gmail and then constructs and sends the email from the parametres.

Its sends the email that has been designed in send email part 1

Public Sub SendEmails2(ByVal FromAddress As String, \_

ByVal Subject As String, \_

ByVal Body As String, \_

ByVal UserName As String, \_

ByVal Password As String, \_

ByVal recipient As String, \_

Optional ByVal Server As String = "smtp.gmail.com", \_

Optional ByVal Port As Integer = 587)

Dim Email As New MailMessage()

'trys to send the email

Try

Dim SMTPServer As New SmtpClient

'fills in the senders email adress from the fromaddress parameter

Email.From = New MailAddress(FromAddress)

'puts in the recipent for the mail

For Each Recipient As String In Recipients

Email.To.Add(Recipient)

Next

'adds subject body and server, host and such information

Email.Subject = Subject

Email.Body = Body

SMTPServer.Host = Server

SMTPServer.Port = Port

SMTPServer.Credentials = New System.Net.NetworkCredential(UserName, Password)

SMTPServer.EnableSsl = True

'sends it

SMTPServer.Send(Email)

'clears it

Email.Dispose()

'notificaltion if smtp failed

Catch ex As SmtpException

Email.Dispose()

MsgBox("Sending Email Failed. Smtp Error.")

'notification if portnuimber owas wrong

Catch ex As ArgumentOutOfRangeException

Email.Dispose()

MsgBox("Sending Email Failed. Check Port Number.")

'notification if portnunber is wrong

Catch Ex As InvalidOperationException

Email.Dispose()

MsgBox("Sending Email Failed. Check Port Number.")

End Try

End Sub

### Public Sub sendemailspart1

This subroutine is triggerd by being called. I goes through every student and staff member and finds all there appointments. It then puts the time and the names of the other part of each appointment and the time into a variable used to contain the body of an email. Once it has all f the appoitments compiled into the body string it calls the second email function.

It finds all the appionments for every one and in passes them on to be emailed.

public sub sendemailspart1()

Dim subject As String = "Consultation evening appointments"

Dim body As String = ""

Dim username As String = "sim.bellows@gmail.com"

Dim password As String = "l09m3e?!"

Dim recipient As String

'loop that cycles through each student so each gets an email

For counter1 As Integer = 1 To Nstudents

'loads current students details

student = GetStudent(counter1)

'generates the students school email

recipient = student.StudID + "@WMSF.ac.uk"

'puts the initial greeting for the email and clears the old message

body = "Dear " & student.Forename & " " & student.Surname & vbNewLine & vbNewLine

'finds each appointment of the student

For counter2 As Integer = 1 To NAppointment

If Appointment.studNO = student.StudNO Then

'gets the name of the member of staff the appointment is with

Staff = GetStaff(Appointment.StaffNO)

'puts in the details of the appiontment into the email

body = body & Staff.Forename & " " & Staff.Surname & "" & militarytime(Appointment.start) & " day " & Appointment.day & vbNewLine

End If

Next

'puts the sign off of the email into the text

body = body & vbNewLine & "thank you very much" & vbNewLine & "simon bellows" & vbNewLine & vbNewLine & "deputy head"

'calls the routine to send the email

Call SendEmails2("sim.bellows@gmail.com", subject, body, username, password, recipient)

Next

'loop that cycles through each staff member so each gets an email

For counter1 As Integer = 1 To Nstaff

'loads current staff members details

Staff = GetStaff(counter1)

'genereates the staff school email

recipient = Staff.staffID + "@WMSF.ac.uk"

'puts the intitial greeting for the email and clears the old message

body = "Dear " & Staff.Forename & " " & Staff.Surname & vbNewLine & vbNewLine

'finds each appoointment of the staff member

For counter2 As Integer = 1 To NAppointment

If Appointment.StaffNO = Staff.StaffNO Then

'gets the name of the student the appointmetn is with

student = GetStudent(Appointment.studNO)

'puts in the details of the appointment into the email

body = body & student.Forename & " " & student.Surname & "" & militarytime(Appointment.start) & " day " & Appointment.day & vbNewLine

End If

Next

'puts the sign off of the email into the message

body = body & vbNewLine & "thank you very much" & vbNewLine & "simon bellows" & vbNewLine & vbNewLine & "deputy head"

'calls the routine to send the email

Call SendEmails2("sim.bellows@gmail.com", subject, body, username, password, recipient)

Next

end sub

### Public Sub TheSortingAlogorithm

This subroutine is triggerd by being called. For each student it finds all the lessons they have. It then loads up information on the first teacher. Its finds the first time they are mutualy available and generates that appointment. It changes block and availability information for the students other availability records so that no appointment can be outside a single block can be used. Then it repeats with the next lesson.

It is a subroutine that is used to divvy out appointments for students.

Public Sub TheSortingAlogorithm()

Dim OnAppointment As Integer = 0

Dim lowerbound As Integer = 0

'for loop goes through every student

For counter1 As Integer = 0 To Nstudents

'gets studnet information

student = GetStudent(counter1)

'finds all the lessons with the student and retives the teacher

For counter2 As Integer = 0 To Nlesson

Lesson = Getlesson(counter2)

If Lesson.StudNO = student.StudNO Then

Staff = GetStaff(counter2)

'for each students availablity it looks for avaliable spots that are also available for the teacher

For counter3 As Integer = 0 To NStudAv

StudAv = GetStudAV(counter3)

If StudAv.StudNo = student.StudNO And StudAv.available = True Then

For counter4 As Integer = 0 To NStaffAv

StaffAv = GetStaffAV(counter4)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Available = True Then

'an appointment slot has been found for which both the studen and staff memver are available for

'the record for the appointment is populated

Appointment.AppointmentNO = OnAppointment

Appointment.studNO = StudAv.StudNo

Appointment.StaffNO = StaffAv.StaffNO

Appointment.day = StudAv.DayNO

Appointment.StaffNO = StudAv.Appointment

'handels block values for the student so as to make sure that the values are only within one block.

If Appointmentlength = 5 Then

For counter5 As Integer = 0 To NStudAv

studav2 = GetStudAV(counter5)

'checks if teh block value of the current spot is 21 because if it is and the other is 20 then they are

'not in the same block and it needs to be set as unavailable

If StudAv.Block = 21 And studav2.Block = 20 Then

studav2.Block = 0

studav2.available = False

PutStudAv(studav2, studav2.studAVNO)

'checks if teh block value of the current spot is 20 because if it is and the other is 21 then they are

'not in the same block and it needs to be set as unavailable

ElseIf StudAv.Block = 20 And studav2.Block = 21 Then

studav2.Block = 0

studav2.available = False

PutStudAv(studav2, studav2.studAVNO)

End If

Next

Else

'handles 10 min cases

For counter5 As Integer = 0 To NStudAv

studav2 = GetStudAV(counter5)

'checks if teh block value of the current spot is 20 because if it is and the other is 22 or 23 then they are

'not in the same block and it needs to be set as unavailable

If StudAv.Block = 20 And (studav2.Block = 22 Or studav2.Block = 23) Then

studav2.Block = 0

studav2.available = False

PutStudAv(studav2, studav2.studAVNO)

'checks if teh block value of the current spot is 21 because if it is and the other is 23 then they are

'not in the same block and it needs to be set as unavailable

ElseIf StudAv.Block = 21 And studav2.Block = 23 Then

studav2.Block = 0

studav2.available = False

PutStudAv(studav2, studav2.studAVNO)

'checks if teh block value of the current spot is 23 because if it is and the other is 20 or 21 then they are

'not in the same block and it needs to be set as unavailable

ElseIf StudAv.Block = 23 And (studav2.Block = 20 Or studav2.Block = 21) Then

studav2.Block = 0

studav2.available = False

PutStudAv(studav2, studav2.studAVNO)

'checks if teh block value of the current spot is 22 because if it is and the other is 20 then they are

'not in the same block and it needs to be set as unavailable

ElseIf StudAv.Block = 22 And studav2.Block = 20 Then

studav2.Block = 0

studav2.available = False

PutStudAv(studav2, studav2.studAVNO)

End If

Next

End If

'sets the staffav availablitity

StaffAv.Available = False

PutStaffAv(StaffAv, StaffAv.staffAVNO)

'sets the stud blocks and availablitity for appointment blocks

'cycles through each studavrecord

For counter5 As Integer = 0 To NStudAv

studav2 = GetStudAV(counter5)

'for appointmetn length 5

If Appointmentlength = 5 Then

'works out the first appointment of the half hour

lowerbound = (studav2.Appointment \ 6) \* 6

'checks if the studav is not already been set in the earlier check and if it is for the right student

If studav2.StudNo = student.StudNO And StudAv.Block <> 21 Or 20 Then

Select Case studav2.Appointment

'too early

Case Is <= (lowerbound - 7)

studav2.available = False

studav2.Block = 0

'too late

Case Is >= (lowerbound + 12)

studav2.available = False

studav2.Block = 0

'half hour early

Case (lowerbound - 6) To (lowerbound - 1)

studav2.Block = 20

'half hour later

Case (lowerbound + 6) To (lowerbound + 11)

studav2.Block = 21

'appointmetn before

Case (StudAv.Appointment - 1)

studav2.available = False

studav2.Block = 0

'appointment after

Case (StudAv.Appointment + 1)

studav2.available = False

studav2.Block = 0

'the appointmetn in question

Case StudAv.Appointment

studav2.Block = 0

studav2.available = False

End Select

End If

Else

'works out the first appointment of the half hour

lowerbound = (studav2.Appointment \ 6) \* 6

'checks if the studav is not already been set in the earlier check and if it is for the right student

If studav2.StudNo = student.StudNO And StudAv.Block <> 21 Or 20 Then

Select Case studav2.Appointment

'too early

Case Is <= (lowerbound - 13)

studav2.available = False

studav2.Block = 0

'too late

Case Is >= (lowerbound + 18)

studav2.available = False

studav2.Block = 0

'half hour early

Case (lowerbound - 6) To (lowerbound - 1)

studav2.Block = 21

'half hour later

Case (lowerbound + 6) To (lowerbound + 11)

studav2.Block = 22

'hour early

Case (lowerbound - 12) To (lowerbound - 7)

studav2.Block = 20

'hour later

Case (lowerbound + 12) To (lowerbound + 17)

studav2.Block = 23

'appointmetn before

Case (StudAv.Appointment - 1)

studav2.available = False

studav2.Block = 0

'appointment after

Case (StudAv.Appointment + 1)

studav2.available = False

studav2.Block = 0

'the appointmetn in question

Case StudAv.Appointment

studav2.Block = 0

studav2.available = False

End Select

End If

End If

Next

End If

Next

End If

Next

End If

Next

Next

End Sub

# List of variables

## Global variables

Public Appointment As AppointmentsRec = Nothing

*mod1*

This variable is used to store appointment records when they are being manipulated by the program

Public Appointmentlength As Integer = -1

*mod1*

This variable is used to store the appointment length of the parents evening.

Public Day As DayRec = Nothing

*mod1*

This variable is used to store day records when they are being manipulated by the program

Public Lesson As LessonRec = Nothing

*mod1*

This variable is used to store lesson records when they are being manipulated by the program

Public NAppointment As Integer = -1

*mod1*

This variable is used to store the number of appointment records in the appointment.dat file

Public NDay As Integer = -1

*mod1*

This variable is used to store the number of day records in the day.dat file

Public Nlesson As Integer = -1

*mod1*

This variable is used to store the number of lesson records in the lesson.dat file

Public Nstaff As Integer = -1

*mod1*

This variable is used to store the number of staff records in the staff.dat file

Public NStaffAv As Integer = -1

*mod1*

This variable is used to store the number of staff availability records in the staffav.dat file

Public NStudAv As Integer = -1

*mod1*

This variable is used to store the number of student availability records in the studav.dat file

Public Nstudents As Integer = -1

*mod1*

This variable is used to store the number of student records in the students.dat file

Public Staff As StaffRec = Nothing

*mod1*

This variable is used to store staff records when they are being manipulated by the program

Public StaffAv As StaffAvRec = Nothing

*mod1*

This variable is used to store staff availability records when they are being manipulated by the program

Public stopimport As Boolean = False

*mod1*

this variable is used to store whether or not to stop an import due to errors in the data being inputed

Public StudAv As StudAvRec = Nothing

*mod1*

This variable is used to store student availability records when they are being manipulated by the program

Public studav2 As StudAvRec = Nothing

*mod1*

This variable is used to store student availability records when they are being manipulated by the program when 2 student availability records are being compared

Public student As StudRec = Nothing

*mod1*

This variable is used to store student records when they are being manipulated by the program

Public user As String

*mod1*

This variable is used to store the string that the user used as his network username

Public usertype As Byte = 0

*mod1*

This variable is used to store the user type of the user, 0 is not on the system 1 is a student and 2 is a student.

## Module-level variables

**frmDaySettings**

Public change As Boolean = False

This variable is used to store whether or not the change was done by the user or by a piece of code triggering it self

Public change1 As Boolean = False

This variable is used to store whether or not the change was done by the user or by a piece of code triggering it self

**mod1.AppointmentsRec**

Public AppointmentNO As Byte

Public day As integer

Public StaffNO As integer

Public start As integer

Public studNO As integer

This set of variables is saved as a structure called appointments rec and is the format of an appointment record.

**mod1.DayRec**

Public DayNO As Byte

Public finish As Integer

Public Start As Integer

This set of variables is saved as a structure called dayrec and is the format of a day record.

**mod1.LessonRec**

Public LessonNO As Integer

Public StaffNO As Short

Public StudNO As Short

This set of variables is saved as a structure called lessonrec and is the format of a lesson record.

**mod1.StaffAvRec**

Public Appointment As Integer

Public Available As Boolean

Public Block As Byte

Public DayNO As Byte

Public staffAVNO As integer

Public StaffNO As Byte

This set of variables is saved as a structure called staffavrec and is the format of a staff availablility record.

**mod1.StaffRec**

Public admin As Boolean

Public Forename As String

Public staffID As String

Public StaffNO As Byte

Public Surname As String

This set of variables is saved as a structure called staffrec and is the format of a staff record.

**mod1.StudAvRec**

Public Appointment As Integer

Public available As Boolean

Public Block As Byte

Public DayNO As Byte

Public studAVNO As integer

Public StudNo As Short

This set of variables is saved as a structure called studavrec and is the format of a student avability record.

**mod1.StudRec**

Public Forename As String

Public StudID As String

Public StudNO As Short

Public Surname As String

Public Year As Byte

This set of variables is saved as a structure called studrec and is the format of a student record.

## Data type summary

**Variables and parameters**

**Type As Type $%&!#@^ Implicit Total**

Boolean 6 6

Byte 10 10

Object 1 1

Other 47 47

Short 4 4

String, $ 22 22

String(), $() 7 7

Structure 15 15

integer 84 84

Total 196 0 0 196

**Summary Total %**

Numeric 104 53%

String 22 11%

Array 7 4%

Other 63 32%

**Total 196 100%**

### Glossary

As Type: Variable declared with regular 'As Datatype' clause.

$%&!#@^: Variable declared with type character.

Implicit: Variable declared with no explicit datatype. Compiler decides type.