Title: Student Time Management System

Name: Williams, Alexander Pierce

School: Campion College

Centre #: 100016

Candidate #: 1000163072

Table of Contents

Introduction

Background

Campion College is a top-ranking Catholic secondary school located on the Caribbean Island of Jamaica. Founded in January of 1960, Campion has built a reputation of elite academics hence associating its students with the best in the island and by extension the Caribbean. The school has approximately 1400 students enrolled in total; around 160 of those students being in Lower Sixth Form (6B). Each of these 160 6B students is required to do one school mandated subject as well as 3 or 4 subjects of their choosing. Classes are scheduled to occur in six periods from 8:00 - 15:00 with timetables varying from student to student. Most students have a minimum of one break every school day that lasts for a minimum of one hour and ten minutes.

Encouraging students to be well-rounded, the school mandates that all students partake in a minimum of two co-curricular activities which is an additional component to an already busy schedule. 6B students are expected to be responsible in managing their time in order to complete assignments and stay up to date in all their classes. The school has allotted longer breaks for these students hoping that they will use the extra time wisely but at the end of the day it is the student's choice as to what they do with their time. That being said, time is a crucial factor for 6B students and the school stresses the importance of maximising the productivity of 'free-time'.

Problem Context

As they day begins students check their schedule to see the lineup of classes that they have. The schedule issued by the school simply displays a table that shows the subjects taken by the student and their respective time periods.

On a day to day basis 6B students receive assignments in each class. These assignments may be in the form of homework, classwork, or tests. Some write it down in a book dedicated to keeping track of these assignments but many leave it up to their brain to remember. For those who record assignments in a book, they usually store it in their school bags.

When writing down assignments in a book or remembering assignments three things are usually noted by students; the type of assignment, the subject and the due date.

After the scheduled school day many take part in co-curricular activities that can end at times ranging from 16:00 to 19:00 and end up arriving at home with school work to face. They refer back to their assignment book or try and remember what they received that day.

Most students therefore spend the rest of the night preparing for or completing certain assignments. Students then go to bed when they feel they have accomplished a sufficient amount of work.

The next day students turn in completed assignments or sit tests that they should have prepared for; repeating the process that they use to record and keep track of assignments.

This process is repeated day after day, week after week, term after term.

Problem Description

- When the assignments are kept track of in the book there is no idea of the amount of time to be allocated to prepare for or do each one. This often leads to students underestimating the amount of time required for completion hence causing shorter amounts of sleep. Less tasks are completed when this occurs which leads to an accumulation of uncompleted assignments.
- Those who keep track of assignments via memory are at a greater disadvantage as they are susceptible to forgetting what they have to do hence leading to stress and anxiety upon the realisation that they haven't started.
- The book that students record assignments in can easily be lost and has a finite amount of pages. When the book is finished a new one has to be purchased and the total amount of money spent on these books adds up over time.
- Students continue to take down assignments due in the future while still not having cleared or completed all of the previously recorded assignments. This leads to a build up of stress in the student.
- The school issued schedules do not include the time frame for which the students' co-curricular activities take place and is not able to be manipulated by students.

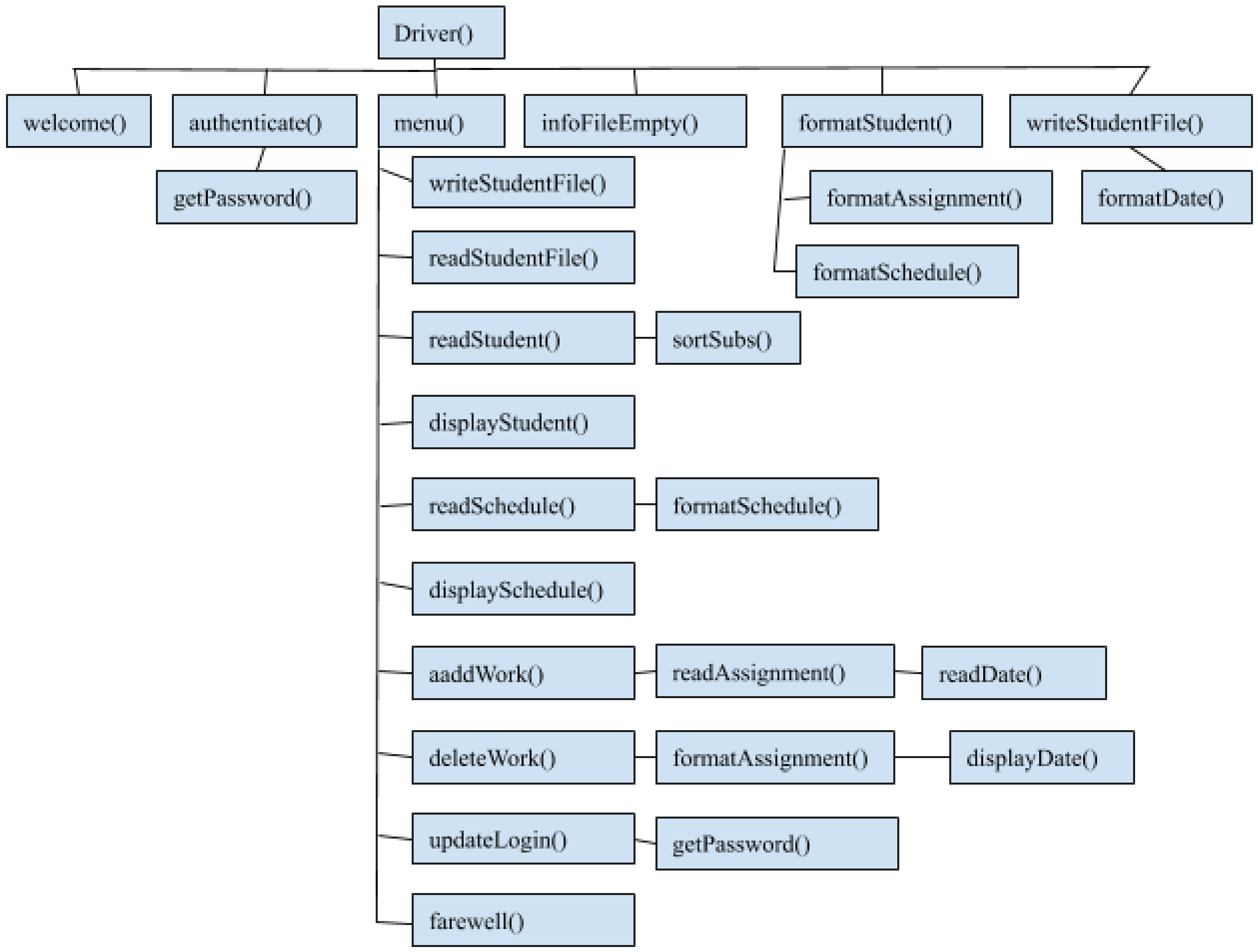
Objectives

The goal is to implement a computer based system that is able to:

- Provide an easy-to-use interface for the user to input data and navigate.
- Help facilitate students' timely completion of school assignments.
- Save a student's information to a file for future use and manipulation.
- Provide a password protected environment for data entry, manipulation and viewing.
- Store the student's schedule and facilitate easy manipulation and displaying of said schedule.
- Store and display a student's pending assignments.

Design Specification

Structured Chart



Narratives

Driver Module:

1. Call the welcome() module.

2. Assign to loggedIn the value of the authenticate module.

3. If the value of infoFileEmpty is 0 then assign formatStudent to student and pass student to writeStudentFile

4. If loggedIn is 1 then pass student to menu, if not then print an error

message.

infoFileEmpty Module:

1. Open "StudentInfo.txt" for appending and assign it to fp.

2. Return 0 if fp is not equal to NULL and is empty but return 1 if it is not equal to NULL and is not empty.

3. Return -1 if the file pointer is equal to NULL.

authenticate Module:

```
1. Open "Login.txt" for appending and reading and assign it to
fpRead.

2. If fpRead is not equal to NULL and is not empty then

    -read the username and password from file "Login.txt"

    -until it is equal to the one read from file "Login.txt"
prompt the user for the username and password and assign 1 to
loggedIn

    -close fpRead

3.If fpRead is not equal to NULL and is empty then

    -close fpRead

    -prompt and read the username and password from the user

    -open "Login.txt" for writing and assign to fpWrite

    -write the username and password to file and close fpWrite

    -assign 1 to the login status.

3. If the file could not be opened then assign 0 to loggedIn

4. Return loggedIn
```

getPassword Module:

```
1. Initialise counter to 0.

2. Read character from user.

3. While the character is not the enter key

    -assign the character to password[counter]

    -print an asterisk

    -increment the counter

    -read another character.

4. Return the password array.
```

welcome Module:

```
1. Display welcome message
```

menu Module:

```
1. Display menu options:
    1 Edit Information
    2 Display Information
    3 Edit Schedule
    4 Display Schedule
    5 Add Assignments
    6 Mark Assignments as Complete
    7 Display Assignments
    8 Update Login Information
    9 Exit
2. Prompt and read the user's choice
3. If the choice is 1 then assign value of readStudent to st and
pass st to writeStudentFile
4. If the choice is 2 then
    -assign value of readStudentFile to st
    -Pass st to displayStudent if st is not formatted, if it is
then display  warning message
5. If the choice is 3 then
    -assign value of readStudentFile to st
    -if st is not formatted then
```

```
      -pass st to readSchedule and assign its value back to st
      -pass st to writeStudentFile
    -if st.schedule is formatted then display warning message
6. If choice is 4 then
    -assign value of readStudentFile to st
    -if st.schedule is not formatted then
      -pass st.schedule to displaySchedule
    -if st.schedule is formatted then display warning message
7. If choice is 5 then
    -assign value of readStudentFile to st
    -if st is not formatted and st.assignmentNum is less than
ten then
      -pass st to addWork and assign its value back to st
      -pass st to writeStudentFile
    -if st is formatted or st.assignment num is greater than or
equal to ten then display warning message
8.  If choice is 6 then
    -assign value of readStudentFile to st
    -if st.assignmentNum is not 0 and st is not formatted then
      -pass st to deleteWork and assign its value back to st
      -pass st to writeStudentFile
    -if st.assignmentNum is 0 and st is formatted then
      -display warning message
9. If choice is 7 then
    -assign value of readStudentFile to st
    -if st.assignment num is not 0 then
      -pass st to displayWork
    -if st.assignment num is 0 then
      -display warning message
10. If the choice is not on the menu display error message
11. Repeat steps 2-10 until choice is 9
```

readStudentFile Module:

```
1. Open the "StudentInfo.txt" file for reading and assign it to
fp
2. If the fp is NULL then
    -print an error message
3. If it is  not then
    -read st from the file fp
    -close the file fp
4. Return st
```

writeStudentFile Module:

```
1. Open the "StudentFile.txt" file for writing and assign it to fp.
2. If fp is NULL then
    -print an error message
3. If it is not then
    -write st to file fp
    -close fp
```

updateLogin module:

```
1. Open the "Login.txt" file for writing and assign it to fp.
2. If fp is not NULL then
    -prompt and read the uname from the user
    -prompt for the password and read by assigning getPassword to pwd
    -write uname and pwd to file fp
    -close fp
3. If fp is NULL then
    -display error message
```

farewell Module:

```
1. Display farewell message
```

displayWork Module:

```
1. For the counter is equal to zero to st.assignmentNum - 1
    -pass st.assignment[counter] to displayAssignment
```

displayAssignment Module:

```
1. Display as.name
2. Pass as.dateGiven to displayDate
3. Pass as.dateDue to displayDate
4. Display as.type, as.subject and as.time
```

displayDate Module:

```
1. Display dt.day, dt.month, dt.year separated by '/'
```

deleteWork Module:

```
1. For counter is equal to zero to assignmentNum-1
    -assign formatAssignment to st.assignment[counter]
2. Initialise st.assignmentNum to 0
3. Display success message
4. Return st
```

addWork Module:

```
1. Initialise i as st.assignmentNum

2. Prompt and read the number of assignments to add, num

3. Assign i+num to num

4. For i is equal to zero to assignmentNum-2

    -pass st to readAssignment and assign its value to st.assignment[i]

    -assign st.assignmentNum+1 to st.assignmentNum

5. Return st
```

readAssignment Module:

```
1. Prompt for and read as.name.

2. Prompt for as.dateGiven and assign readDate to it.

3. Prompt for as.dateDue and assign readDate to it.

4. Display assignment type options and prompt and read choice.

5. Assign assignment type to as.type based on the choice.

6. Display user's subjects as options and prompt and read choice.

7. Assign subject to as.subject based on the choice and subject.num.

8. Assign time value to as.time based on as.subject and as.type.
```

```
9. Prompt the user for whether or not they would like to
override the assigned time, as.time, and read choice.

10. If choice is yes then:

    -prompt and read new value for as.time

11. Return as
```

readDate Module:

```
1. Prompt and read dt.day, dt.month, dt.year

2. Return dt
```

displaySchedule:

```
1. Initialise day array with days of the week and intialise
period array with time periods of schedule.

2. For i is equal to 0 to 5

    -display day[i]

3. For r is equal to 0 to 6

    -display period[r]

    -For c is equal to 0 to 4

      -display schedule[r][c]
```

readSchedule Module:

```
1. Initialise day array with days of the week

2. Assign formatSchedule to st.schedule

3. For c is equal to 0 to 4

    -display day[c]

    -display the users subjects as options and prompt and read choice

    -For r is equal to 0 to 5

      -assign subject to st.schedule[r][c] based on choice and subject.num

    -If st.activityNum is not 0 Then

      -display the users activities as options and prompt and read choice

      -assign activity to st.schedule[6][c] based on choice
```

formatStudent Module:

```
1. Assign arbitrary values to st.fName, st.lName, st.subjectNum, the elements of st.subject, st.activityNum

2. For i is equal to 0 to 9

    -assign formatAssignment to st.assignment[i]

3. Assign formatSchedule to st.schedule

4. Return st
```

formatAssignment:

```
1. Assign arbitrary values to as.name, as.type, as.subject,
as.time

2. Assign formatDate to as.dateGiven as well as as.dateDue

3. Return as
```

formatDate Module:

```
1. Assign arbitrary values to dt.day, dt.month, dt.year.
```

formatSchedule:

```
1. For c is equal to 0 to 4
    -For r is equal to 0 to 5
      -assign arbitrary value to schedule[r][c]
    -assign arbitrary value fo schedule [r][c]
2. Return schedule
```

displayStudent Module:

```
1. Prompt and read st.fName and st.lName
2. For counter is equal to 0 to st.subjectNum-1
    -display st.subject[counter]
3.For counter is equal to 0 to st.activityNum-1
    -display st.activity[counter]
```

readStudent:

```
1. Initialise st.subject[0] to "Communication Studies"

2. Prompt and read st.fName, st.lName, st.SubjectNum

3. Assign st.subjectNum+1 to st.subjectNum

4. Display the list of subjects offered to Campion 6B students
as options.

5. Prompt for choice

6. For i is equal to 1 to st.subjectNum-1

    -Read choice from user

    -Assign subject to st.subject[i] based on the choice

7.Pass st.subject and st.subjectNum to sortSubs and assign its
value to st.subject

8. Prompt and read st.activityNum

9. Display the list of clubs and sports offered to Campion 6B
students as options

10. Prompt for choice

11. For i is equal to 0 to activityNum-1

    -Read choice

    -Assign activity to st.activity[i] based on the choice

12. Return st
```

sortSubs Module:

```
1. For i is equal to subNum-1

    -Display subject[i] as an options

2. For i is equal to 0 to subNum-1

    -Prompt and read choice

    -Assign subject[i] to hold[i] based on the choice

3. For i is equal to 0 to subNum-1

    -Assign hold[i] to subject[i]

4. Return subject
```

Algorithm

```
Record Date
  day: Integer
  month: Integer
  year: Integer
EndDate


Record Assignment
     name: String
     dateGiven: Date
     dateDue: Date
     type: String
     subject: String
     time: Integer
EndAssignment


Record Student
     fName: String
     lName: String
     subjectNum: Integer
     subject[5]: String
```

```
      activityNum: Integer

      activity[5]: String

      assignmentNum: Integer

      assignment[10]: Assignment

      schedule[7][5]: String

EndStudent

sortSubs(subject[5]: String, subNum: Integer): String

    choice, i: Integer

      hold[5]: String


      For i = 0 To subNum-1 Do

      Print i+1, subject[i]

    Endfor


      For i = 0 subNum-1 Do

      Print "Number",i+1,"easiest subject from the above list"

      Read choice


      CASE OF choice

            CASE OF 1:

                 hold[i] = subject[0]
            CASE OF 2:
                  hold[i] = subject[1]
            CASE OF 3:
                  hold[i] = subject[2]
            CASE OF 4:
                  hold[i] = subject[3]
            CASE OF 5:
                  hold[i] = subject[4]
            OTHER:
                  Print "Invalid Choice"
      ENDCASE
      Endfor

      For i = 0 To subNum-1
      subject[i] = hold[i]
      Endfor

    return subject
EndsortSubs

readStudent(): Student
```

```
    st: Student
    choice, i: Integer

     st.subject[0] = "Communication Studies"

     Print "First Name> "
     Read st.fName
     Print "Last Name> "
     Read st.lName
     Print "Number of Subjects (exclusive of Communication
Studies)[3/4]> "
     Read st.subjectNum

     st.subjectNum = st.subjectNum + 1

     Print "    1.  Physics
          2.  Chemistry
          3.  Biology
          4.  Computer Science
          5.  Pure Mathematics
          6.  Digital Media
```

```
      7.  Management of Business
      8.  Literatures in English
      9.  French
      10. Spanish
      11. Economics
      12. Law
      13. Principles of Accounts
      14. Sociology
      15. Geography
      16. History "

Print "Select", st.subjectNum-1, "subjects: "
For i = 1 To st.subjectNum Do
Read choice
CASE OF choice
      CASE OF 1:
            st.subject[i] = "Physics"
     CASE OF 2:
            st.subject[i] = "Chemistry"
     CASE OF 3:
            st.subject[i] = "Biology"
```

```
CASE OF 4:

      st.subject[i] = "Computer Science"

CASE OF 5:

      st.subject[i] = "Pure Mathematics"

CASE OF 6:

      st.subject[i] = "Digital Media"

CASE OF 7:

      st.subject[i] = "Management of Business"

CASE OF 8:

      st.subject[i] = "Literatures in English"

CASE OF 9:

      st.subject[i] = "French"

CASE OF 10:

      st.subject[i] = "Spanish"

CASE OF 11:

      st.subject[i] = "Economics"

CASE OF 12:

      st.subject[i] = "Law"

CASE OF 13:

      st.subject[i] = "Principles of Accounts"

CASE OF 14:
```

```
            st.subject[i] = "Sociology"
      CASE OF 15:
            st.subject[i] = "Geography"
      CASE OF 16:
            st.subject[i] = "History"
ENDCASE
Endfor

st.subject = sortSubs(st.subject, st.subjectNum)

Print "Number of Co-Curricular Activities\n[1-5]> "
Read  st.activityNum

Print " 1.  Aeronautics Club
      2.  Angels of Love
      3.  Animal Club
      4.  Animation
      5.  Art Club
      6.  Campion Coders
      7.  Campion Theatre Ensemble
      8.  Catholic Club
```

9. Chapel Choir
10. Christian Life Community
11. Computer and Media Club
12. Dance Society
13. Debating Society
14. Disaster Preparedness
15. D.I.Y.
16. Engineering Club
17. Gavel Club
18. Gourmet Club
19. Girl Code
20. Green Generation
21. I.S.C.F.
22. Interact Club
23. Key Club
24. Lego Yuh Mind Robotics Club
25. Leo Club
26. Mathematics Club
27. Media and Production Club
28. Medics Club
29. Ministry Outreach Program

30. Modern Language Club
31. Music Club
32. Chords
33. Drum Ensemble
34. Steel Band
35. Peer Counseling
36. Rangers
37. Readers Association
38. Red Cross
39. Science Club
40. Sign Language Club
41. Sixth Form Association
42. Software Engineering Club
43. Student Council
44. Students for Democracy
45. TED - ED
46. The Students' Voice
47. Tourism Action Club
48. United Nations Club
49. Young Entrepreneurial Society
50. Basketball

```
            51. Chess

            52. Fitness & Weightlifting

            53. Football

            54. Hockey

            55. Lawn Tennis

            56. Swimming

            57. Table Tennis

            58. Track and Field

            59. Volleyball

            60. Water Polo"


Print "Select", st.activityNum, "activities(y): "

For i = 0 To st.activityNum-1 Do

Read choice


CASE OF choice

     CASE OF 1:

          st.activity[i] = "Aeronautics Club"

     CASE OF 2:

          st.activity[i] = "Angels of Love"

     CASE OF 3:
            st.activity[i] = "Animal Club"

        CASE OF 4:

            st.activity[i] = "Animation"

        CASE OF 5:

            st.activity[i] = "Art Club"

        CASE OF 6:

            st.activity[i] = "Campion Coders"

        CASE OF 7:

            st.activity[i] = "Campion Theatre Ensemble"

        CASE OF 8:

            st.activity[i] = "Catholic Club"

        CASE OF 9:

            st.activity[i] = "Chapel Choir"

        CASE OF 10:

            st.activity[i] = "Christian Life Community"

        CASE OF 11:

            st.activity[i] = "Computer and Media Club"

        CASE OF 12:

            st.activity[i] = "Dance Society"

        CASE OF 13:

            st.activity[i] = "Debating Society"
```

```
CASE OF 14:
     st.activity[i] = "Disaster Preparedness"
CASE OF 15:
     st.activity[i] = "D.I.Y."
CASE OF 16:
     st.activity[i] = "Engineering Club"
CASE OF 17:
     st.activity[i] = "Gavel Club"
CASE OF 18:
     st.activity[i] = "Gourmet Club"
CASE OF 19:
     st.activity[i] = "Girl Code"
CASE OF 20:
     st.activity[i] = "Green Generation"
CASE OF 21:
     st.activity[i] = "I.S.C.F."
CASE OF 22:
     st.activity[i] = "Interact Club"
CASE OF 23:
     st.activity[i] = "Key Club"
CASE OF 24:
```

```
          st.activity[i] = "Lego Yuh Mind Robotics Club"

      CASE OF 25:

          st.activity[i] = "Leo Club"

      CASE OF 26:

          st.activity[i] = "Mathematics Club"

      CASE OF 27:

          st.activity[i] = "Media and Production Club"

      CASE OF 28:

          st.activity[i] = "Medics Club"

      CASE OF 29:

          st.activity[i] = "Ministry Outreach Program"

      CASE OF 30:

          st.activity[i] = "Modern Language Club"

      CASE OF 31:

          st.activity[i] = "Music Club"

      CASE OF 32:

          st.activity[i] = "Chords"

      CASE OF 33:

          st.activity[i] = "Drum Ensemble"

      CASE OF 34:

          st.activity[i] = "Steel Band"
```

```
CASE OF 35:

     st.activity[i] = "Peer Counseling"

CASE OF 36:

     st.activity[i] = "Rangers"

CASE OF 37:

     st.activity[i] = "Readers Association"

CASE OF 38:

     st.activity[i] = "Red Cross"

CASE OF 39:

     st.activity[i] = "Science Club"

CASE OF 40:

     st.activity[i] = "Sign Language Club"

CASE OF 41:

     st.activity[i] = "Sixth Form Association"

CASE OF 42:

     st.activity[i] = "Software Engineering Club"

CASE OF 43:

     st.activity[i] = "Student Council"

CASE OF 44:

     st.activity[i] = " Students for Democracy"

CASE OF 45:
```

```
          st.activity[i] = "TED - ED"

     CASE OF 46:

          st.activity[i] = "The Students' Voice"

     CASE OF 47:

          st.activity[i] = "Tourism Action Club"

     CASE OF 48:

          st.activity[i] = "United Nations Club"

     CASE OF 49:

          st.activity[i] = "Young Entrepreneurial Society"

     CASE OF 50:

          st.activity[i] = "Basketball"

     CASE OF 51:

          st.activity[i] = "Chess"

     CASE OF 52:

          st.activity[i] = "Fitness & Weightlifting"

     CASE OF 53:

          st.activity[i] = "Football"

     CASE OF 54:

          st.activity[i] = "Hockey"

     CASE OF 55:

          st.activity[i] = "Lawn Tennis"
          CASE OF 56:
               st.activity[i] = "Swimming"
          CASE OF 57:
               st.activity[i] = "Table Tennis"
          CASE OF 58:
               st.activity[i] = "Track and Field"
          CASE OF 59:
               st.activity[i] = "Volleyball"
          CASE OF 60:
               st.activity[i] = "Water Polo"
           OTHER:
                Print "Invalid option"
     ENDCASE OF
     Endfor

    return st
EndreadStudent


displayStudent(st: Student)
    i: Integer
```

```
        Print "Your Name> ", st.fName, st.lName


        Print "Your Subjects: "
        For i = 0 To st.subjectNum-1 Do
        Print st.subject[i]
        Endfor


        Print "Your Activities: "
        For i = 0 To st.activityNum-1 Do
        Print st.activity[i]
      Endfor
EnddisplayStudent


formatSchedule(): String
        r,c: Integer
      schedule[7][5]: String
        For c = 0 To 4 Do
        For r = 0 To 5 Do
            schedule[r][c] = "NO CLASS"
         Endfor
        schedule[r][c] = "NO ACTIVITY"
      Endfor

      return schedule
EndformatSchedule

formatDate(): Date
        dt: Date

      dt.day = 0
        dt.month = 0
        dt.year = 2019

      return dt
EndformatDate

formatAssignment(): Assignment
        as: Assignment

      as.name = "--"
        as.dateGiven = formatDate()
```

```
     as.dateDue = formatDate()

     as.type = "--"

     as.subject = "--"

     as.time = 0


    return as

EndformatAssignment


formatStudent(): Student

     i: Integer

    st: Student


     st.fName = "--"

     st.lName = "--"

     st.subjectNum = 0


     For i = 0 To 4 Do

     st.subject[i] = "--"

     Endfor


     st.activityNum = 0
```

```
      For i = 0 To 4 Do
      st.activity[i] = "--"
    Endfor

      st.assignmentNum = 0

      For i = 0 To 9 Do
      st.assignment = formatAssignment()
      Endfor

      st.schedule = formatSchedule()

    return st
EndformatStudent


readSchedule(Student st): Student
      i, r, c, choice: Integer
      day[5] = {"Monday", "Tuesday", "Wednesday", "Thursday",
"Friday"}: String
```

```
st.schedule = formatSchedule()


For c = 0 To 4 Do

Print "Enter Classes for ",day[c], ":"


For i = 0 To st.subjectNum-1 Do

     Print i+1, st.subject[1]

Endfor


 Print i+1, "No Class"


For r = 0 To 5 Do

     Print "Period", r+1, "Class> "

     Read choice


     If st.subjectNum = 4 Then

          CASE OF choice

               CASE OF 1:

                    st.schedule[r][c] = st.subject[0]

               CASE OF 2:
                  st.schedule[r][c] = st.subject[1]
            CASE OF 3:
                  st.schedule[r][c] = st.subject[2]
            CASE OF 4:
                  st.schedule[r][c] = st.subject[3]
            CASE OF 5:


             OTHER:
            Print "Invalid Choice"
        ENDCASE OF
Else
      CASE OF choice
            CASE OF 1:
            st.schedule[r][c] = st.subject[0]
            CASE OF 2:
            st.schedule[r][c] = st.subject[1]
            CASE OF 3:
            st.schedule[r][c] = st.subject[2]
            CASE OF 4:
            st.schedule[r][c] = st.subject[3]
            CASE OF 5:
                st.schedule[r][c] = st.subject[4]
                 CASE OF 6:


                 OTHER:
                      Print "Invalid Choice"
           ENDCASE
      Endif
Endfor


If st.activityNum <> 0 Then
      Print "Activity for", day[c], ":"


      For i = 0 To st.activityNum-1 Do
           Print i+1, st.activity[i]
      Endfor


       Print i+1, "No Activity"


      Print "Activity: "
      Read choice
```

```
CASE OF choice

          CASE OF 1:

                st.schedule[6][c] = st.activity[0]

          CASE OF 2:

                If st.activity[1] = "--" Then



          Else

               st.schedule[6][c] = st.activity[1]

                Endif

          CASE OF 3:

          If st.activity[2] = "--" Then



          Else

                      st.schedule[6][c] = st.activity[2]

          Endif

          CASE OF 4:

          If st.activity[3] = "--" Then



          Else

                      st.schedule[6][c] = st.activity[3]

          Endif
```

```
                    CASE OF 5:
                          If st.activity[4] = "--" Then


                    Else
                               st.schedule[6][c] = st.activity[4]
                    Endif
                    CASE OF 6:


                     OTHER:
                          Print "Invalid Option"
          ENDCASE
     Endif
     Endfor


    return st
EndreadSchedule


displaySchedule(schedule[7][5]: String)
     r, c, i: Integer
     day[6] = {"","Monday", "Tuesday", "Wednesday", "Thursday"},
"Friday": String
```

```
    period[7] = {"8:00-9:10", "9:10-10:20", "10:20-11:30",
"11:30-12:40", "12:40-1:50", "1:50-3:00", "Activity"}: String


    For i = 0 to 5 Do

    Print day[i]

    Endfor



    For r = 0 To 6 Do

    Print period[r]



    For c = 0 To 4 Do

        Print schedule[r][c]

    Endfor

    Endfor

EnddisplaySchedule


readDate(): Date

    dt: Date



    Print "Day> "

    Read dt.day
    Print "Month> "
    Read dt.month

    dt.year = 2019

    return dt
EndreadDate

readAssignment(Student st): Assignment
    as: Assignment
    choice: Character
    i: Integer

    Print "Assignment Name> "
    Read as.name
    Print "Date Given: "
    as.dateGiven = readDate()
    Print "Date Due: "
    as.dateDue = readDate()
```

```
Print "1. HW"

Print "2. CW"

Print "3. TEST"

Print "Assignment Type> "

Read choice


CASE OF choice

CASE '1':

     as.type = "HW"

CASE '2':

     as.type = "CW"

CASE '3':

     as.type = "TEST"

ENDCASE


For i = 0 To st.subjectNum-1 Do

Print i+1, st.subjec[1]

Endfor


Print "Assignment Subject> "

Read choice
```

```
If st.subjectNum = 5 Then
CASE OF choice
     CASE OF '1':
          as.subject = st.subject[0]
          If as.type = "HW" Then
               as.time = 30
          Else
               If as.type = "CW" Then
               as.time = 45
               Else
               as.time = 60
                Endif
          Endif
     CASE OF '2':
          as.subject = st.subject[1]
          If as.type = "HW" Then
               as.time = 45
          Else
               If as.type = "CW" Then
               as.time = 60
               Else
```

```
                as.time = 75
                 Endif
           Endif
      CASE OF '3':
           as.subject = st.subject[2]
           If as.type = "HW" Then
                as.time = 60
           Else
                If as.type = "CW" Then
                as.time = 75
                Else
                as.time = 90
                Endif
           Endif
      CASE OF '4':
           as.subject = st.subject[3]
           If as.type = "HW" Then
                as.time = 75
           Else
                If as.type = "CW" Then
                as.time = 90
                Else
                as.time = 105
                 Endif
            Endif
        CASE OF '5':
            as.subject = st.subject[4]
            If as.type = "HW" Then
                as.time = 90
            Else
                If as.type = "CW" Then
                as.time = 105
                Else
                as.time = 120
                Endif
            Endif
        OTHER:
            Print "Invalid Choice"
    ENDCASE
    ELSE
    CASE OF choice
        CASE OF '1':
```

```
    as.subject = st.subject[0]
    If as.type = "HW" Then
        as.time = 30
    Else
        If as.type = "CW" Then
        as.time = 45
        Else
        as.time = 60
        Endif
    Endif
CASE OF '2':
    as.subject = st.subject[1]
    If as.type = "HW" Then
        as.time = 45
    Else
        If as.type = "CW" Then
        as.time = 60
        Else
        as.time = 75
        Endif
    Endif
```

```
CASE OF '3':
     as.subject = st.subject[2]
     If as.type = "HW" Then
          as.time = 60
     Else
          If as.type = "CW" Then
          as.time = 75
          Else
          as.time = 90
          Endif
     Endif
CASE OF '4':
     as.subject = st.subject[3]
     If as.type = "HW" Then
          as.time = 75
     Else
          If as.type = "CW" Then
          as.time = 90
          Else
          as.time = 105
          Endif
```

```
                Endif
            OTHER:
                Print "Invalid Choice"
        ENDCASE
        Endif

        Print "Suggested time for preparation/completion is ", as.time,
" minutes. Would you like to override this time?[Y/N]"
      Read choice

        CASE OF choice
         CASE OF 'Y': CASE OF 'y':
            Print "New Time in minutes> "
            Read as.time
        CASE OF 'N': CASE OF 'n':

        OTHER:
            Print "Invalid Choice"
        ENDCASE

        return as
```

```
EndreadAssignment

addWork(Student st): Student
    i = st.assignmentNum, num: Integer

    Print "Number of assignments to add> "
    Read num

    num = i + num

    For i To num-1 Do
    st.assignment[i] = readAssignment(st)
    st.assignmentNum = st.assignmentNum + 1
    Endfor

   return st
EndaddWork

deleteWork(Student st): Student
    i: Integer
```

```
      For i = 0 To i < st.assignmentNum-2 Do

      st.assignment[i] = formatAssignment()

      Endfor


      st.assignmentNum = 0


      Print "Assignments Have Been Cleared"


    return st

EnddeleteWork


displayDate(Date dt)

      Print dt.day"/",dt.month,"/",dt.year

EnddisplayDate


displayAssignment(Assignment as)

      Print "Assignment Name> ", as.name

      Print "Date Given: ", displayDate(as.dateGiven)

      Print "Date Due: ", displayDate(as.dateDue)

      Print "Assignment Type: ", as.type

      Print "Assignment Subject: " as.subject
      Print "Estimated Time for Completion/Preparation: ", as.time,
"mins"
EnddisplayAssignment


displayWork(Student st)
      i: Integer


      For i = 0 To st.assignmentNum-1 Do
      Print "Assignment #", i+1, displayAssignment(st.assignment[i])
      Endfor
EnddisplayWork


farewell()
    Print "Exiting system..."
Endfarewell


updateLogin()
      uname, pwd: String
      fp: File


    fp = Open "Login.txt", for writing
      If fp <> NULL Then
      Print "New Username> "
      Read uname
      Print "New Password> "
       pwd = getPassword()

       Write uname, pwd to file fp
      close fp
      Else
      Print "Cannot Update at This Time"
      Endif
EndupdateLogin

writeStudentFile(st: Student)
      fp: File

    fp = Open "StudentInfo.txt", for writing
      If fp = NULL Then
      Print "ERROR: Required File Could Not Be Opened"
      Else
      Write st to file fp
```

```
       close fp
      Endif
EndwriteStudentFile

readStudentFile(): Student
      fp: File
      st: Student

     fp = Open "StudentInfo.txt", for reading
      If fp = NULL Then
      Print "ERROR: Required File Could Not Be Opened"
      Else
      Read st from file fp
       close fp
     Endif

      return st
EndreadStudentFile

menu(Student st)
      choice: Character
```

```
Print "    1 Edit Information
        2 Display Information
        3 Edit Schedule
        4 Display Schedule
        5 Add Assignments
        6 Mark Assignments as Complete
        7 Display Assignments
        8 Update Login Information
        9 Exit"

 Repeat
  Read choice

 CASE OF choice
      CASE OF '1':
            st = readStudent()
            writeStudentFile(st)
      CASE OF '2':
            st = readStudentFile()
            If st.fName <> "--" Then
```

```
                displayStudent(st)
          Else
                Print "Please Enter Your Information First!"
          Endif
    CASE OF '3':
          st = readStudentFile()
          If st.subjectNum <> 0 Then
                st  = readSchedule(st)
                writeStudentFile(st)
          Else
                Print "Please Enter Your Information First!"
          Endif
    CASE OF '4':
          st = readStudentFile()
          If st.schedule[0][0] <> "NO CLASS" Then
                displaySchedule(st.schedule)
          Else
                Print "Please Enter Your Schedule First!"
           Endif
    CASE OF '5':
          st = readStudentFile()
```

```
                    If st.subjectNum <> 0 AND st.assignmentNum <10 Then
                         st = addWork(st)
                         writeStudentFile(st)
                    Else
                         If st.assignmentNum =10 Then
                         Print "Unable to store anymore assignments!"
                         Else
                              Print "Please Enter Your Information
First!"
                         Endif
                    Endif
               CASE OF '6':
                    st = readStudentFile()
                    If st.assignmentNum <> 0 AND st.subjectNum <> 0 Then
                         st = deleteWork(st)
                         writeStudentFile(st)
                    Else
                          Print "Assignments are clear! No pending
assignments due..."
                    Endif
               CASE OF '7':
                    st = readStudentFile()
```

```
                If st.assignment[0].name <> "--" Then
                     displayWork(st)
                Else
                     Print "No pending assignments due"
                Endif
           CASE OF '8':
                updateLogin()
           CASE OF '9':
                farewell()
           OTHER:
                Print "Please select an option from the menu"
      ENDCASE
      Until choice = 9
Endmenu


welcome()
   Print "Welcome to the Campion College 6B Student Time Management
System!"
   Print "The solution to all of your procrastination woes!"
Endwelcome
```

```
getPassword(): String
     c: Character
    i = 0: Integer
    password[25]: Character

    Read c
    While c <> ENTER Do
      password[i] = c
      Print "*"
      i = i + 1
      Read c
    Endwhile

    Return password
EndgetPassword

authenticate(): Integer
     nameOnFile, passwordOnFile, uname, pwd: String
     fpRead, fpWrite: File
    loggedIn: Integer
```

```
fpRead = Open "Login.txt", for appending and reading
 If fpRead <> NULL Then
 If fpRead NOT empty Then
      Read nameOnFile, passwordOnFile from file fpRead

      Print "Username> "
      Read uname
      Print "Password> "
      pwd = getPassword()

       While uname <> nameOnFile OR pwd <> passwordOnFile Do
           Print "Username> "
           Read uname
           Print "Password> "
           pwd = getPassword()
      Endwhile

      loggedIn = 1
 Else
       Print "Username> "
      Read uname
```

```
          Print "Password> "
          pwd = getPassword()

           fpWrite = Open "Login.txt", for writing

           If fpWrite <> NULL Then
               Write uname, pwd to file fWrite
               loggedIn = 1
               close fpWrite
          Else
               loggedIn = 0
           Endif
     Endif
      close fpRead
     Else
     loggedIn = 0
    Endif

    return loggedIn
Endauthenticate
```

```
infoFileEmpty(): Integer
     fp = Open "StudentInfo.txt", for appending

     If fp <> NULL Then
      If fp = empty Then
           close fp
           return 0
      Else
           close fp
           return 1
      Endif
     Else
     return -1
     Endif
EndinfoFileEmpty

Driver()
     student: Student
     loggedIn: Integer

     welcome()
```

```
    loggedIn = authenticate()


    If infoFileEmpty() = 0 Then
    student = formatStudent()
    writeStudentFile(student)
    Endif


    If loggedIn = 1 Then
    menu(student)
    Else
    Print "User could not be authenticated"
    Endif
EndDriver
```

Test Plans

Input Screen/Functionality	Input	Purpose of Test	Expected Results
1. Login Screen	Username: teacher Password: 5678	To validate that if an invalid username or password is entered the program will not proceed	The program will continuously prompt for the correct username password combination
2. Login Screen	Username: student Password: 1234	To validate that the program will proceed after the correct username and password combination is entered	The program will format the required structures if necessary and proceed to the main menu module
3. Main Menu	Choice: w	To validate that the program will not proceed if an invalid menu option is entered	The program will continuously prompt for an option from the menu
4. Student Info Screen	First Name: Alexander Last Name: Williams Number of Subjects: 7	To validate that the program will not proceed if an incorrect number of subjects is entered	The program will prompt for a value of 3 or 4

5. Student Info Screen	First Name: Alexander Last Name: Williams Number of Subjects: 4 Select 4 Subjects: 1 2 4 5 Number 1 easiest: 5 Number 2 easiest: 2 Number 3 easiest: 3 Number 4 easiest: 1 Number 5 easiest: 4 Number of Co-Curricular activities: 6	To validate that the program will not proceed if an incorrect number of activities is entered	The program will prompt for a value from 1-5

6. Student Info Screen	First Name: Alexander Last Name: Williams Number of Subjects: 4 Select 4 Subjects: 1 2 4 5 Number 1 easiest: 5 Number 2 easiest: 2 Number 3 easiest: 3 Number 4 easiest: 1 Number 5 easiest: 4 Number of Co-Curricular activities: 3 Select 3 Activities: 16 35 53	To validate that the program will proceed when valid data is entered	The program will display a success message and prompt the user's input to return to the main menu
7. Main Menu (Display Information Screen)	Choice: 2	To validate that the program displays the correct information when the user chooses to display it	The program will call the displayStudent module and then prompt the user's input to return to the main menu

8. Edit Schedule Screen	Monday Period 1: 9 Period 2: 1 Period 3: 5 Period 4: 6 Period 5: 3	To validate that the program displays an error message after an invalid choice is made	The program will state that an invalid choice has been made and continue to read choices for the schedule
9. Edit Schedule Screen -> Display Schedule Screen	Monday Period 1: 2 Period 2: 1 Period 3: 5 Period 4: 6 Period 5: 3 Period 6: 6 Activity: 2 Tuesday Period 1: 2 Period 2: 6 Period 3: 3 Period 4: 6 Period 5: 1 Period 6: 5 Activity: 3 Wednesday Period 1: 2 Period 2: 4 Period 3: 6 Period 4: 1	To validate that the program will display a valid schedule after the user enters the data for said schedule	The program will read the user's input correctly and display a valid schedule

	Period 5: 6 Period 6: 5 Activity: 1 Thursday Period 1: 3 Period 2: 6 Period 3: 6 Period 4: 1 Period 5: 5 Period 6: 4 Activity: 3 Friday Period 1: 6 Period 2: 4 Period 3: 6 Period 4: 3 Period 5: 6 Period 6: 2 Activity: 4 ENTER Menu Choice 4		
10. Add Assignments Screen	Number of Assignments to Add: 11	To validate that the program will not proceed if an invalid number of assignments is	The program will prompt the user to enter a valid number assignments based on the space

		added	left in the array storing said assignments
11. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 0	To validate that the program will not proceed if an invalid day value is entered	The program will prompt the user for a day value from 1-31
12. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 6 Month: 13	To validate that the program will not proceed if an invalid month value is entered	The program will prompt the user for a month value from 1-12
13. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 6 Month: 12 Day: 13 Month: 12 Type: 4	To validate that the program will not proceed if an invalid type value is entered	The program will prompt the user for a type value from 1-3
14. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 6 Month: 12 Day: 13 Month: 12 Type: 3	To validate that the program will not proceed if an invalid subject value is entered	The program will prompt the user for a subject value from 1-5

	Subject: 7		
15. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 6 Month: 12 Day: 13 Month: 12 Type: 3 Subject: 2 Override?: m	To validate that the program will not proceed if an invalid choice is entered	The program will prompt the user for a choice of either Y or N
16. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 6 Month: 12 Day: 13 Month: 12 Type: 3 Subject: 2 Override?: Y New Time: 75	To validate that the program will proceed if all data entered is valid	The program will display a success message and prompt the user's input to return to the main menu.
17. Main Menu (Mark Assignments as Complete Screen)	Choice: 6	To validate that the program displays the correct information when the user chooses to display	The program will call the deleteWork module and display a success message. Then it will

		it	prompt for the user’s input to return to the main menu
18. Main Menu (Display Assignments Screen)	Choice: 7	To validate that the program displays the correct information when the user chooses to display it	The program will call the displayWork module and then prompt for the user’s input to return to the main menu
19. Update Login	New Username: student1 New Password: 4321	To validate that the program will change the username and password combination stored on file	The program will write the newly entered username and password to file
20. Main Menu (Farewell Screen)	Choice: 9	To validate that the program will exit when the user chooses to do so	The program will call the farewell module then end and return 0

Application Development

Code

Header File

```
/*

Programmer: Alexander Williams

Date:    30/03/19

File:    STMS.h

Purpose: Definition of structures/records used in the

         Student Time Management System as well as the

         prototypes for the required functions.

*/


#ifndef STMS_H_INCLUDED

#define STMS_H_INCLUDED


typedef struct ///Date Record

{

  int day;

  int month;

  int year;

}Date;


typedef struct ///Assignment Record
```

```
{

  char name[30];

  Date dateGiven;

  Date dateDue;

  char type[4]; ///type of assignment (HW, CW or TEST)

  char subject[30];

  int time; ///estimated time for completion of/preparation for the assignment

}Assignment;


typedef struct ///Student Record

{

  char fName[15];

  char lName[15];

  int subjectNum;

  char subject[5][30];

  int activityNum;

  char activity[5][30];

  int assignmentNum;

  Assignment assignment[10];

  char schedule[7][5][30];

}Student;
```

```
void sortSubs(char [][30],int); ///function to sort the user's subjects by ascending
difficulty

void readStudent(Student *); ///reads data into the members of a 'Student' structure

void displayStudent(Student); ///displays the data stored in the members of a
'Student' structure

void formatSchedule(char [][5][30]); ///formats the elements of a 3D array
representing a schedule with arbitrary data

void formatDate(Date *); ///formats the members of a 'Date' structure with arbitrary
data

void formatAssignment(Assignment *); ///formats the members of an 'Assignment'
structure with arbitrary data

void formatStudent(Student *); ///formats the members of a 'Student' structure with
arbitrary data

void readSchedule(Student *); ///reads data into the elements of a 3D array
representing the schedule of the user

void displaySchedule(char [][5][30]); ///prints/displays the elements of a 3D array
representing the schedule of the user

Date readDate(); ///reads data into the members of a 'Date' structure

Assignment readAssignment(Student ); ///reads data into the members of an
'Assignment' structure

void addWork(Student *); ///populates the 'assignment' member of a 'Student'
structure with a user determined amount of elements

void deleteWork(Student *); ///formats the elements of the 'assignment' member of a
'Student' structure

void displayDate(Date); ///displays the data stored in the members of a 'Date'
structure
```

```
void displayAssignment(Assignment); ///displays the data stored in the members of an 'Assignment' structure

void displayWork(Student ); ///displays the elements of the 'assignment' member of a 'Student' structure

void farewell(); ///displays a farewell message to the user when they exit the program through the menu

void menu(Student); ///displays menu options and calls their respective functions

void welcome(); ///displays a welcome message to the user when they start the program

void getPassword(char []); ///masks the user's input with '.' whenever they enter a password

void authenticate(int *); ///displays a login or sign-up screen and determines if the user is granted access

void updateLogin(); ///updates the username and password of the user that is stored on file

void writeStudentFile(Student); ///writes a 'Student' structure to a random access file

Student readStudentFile(); ///reads a 'Student' structure from a random access file and returns it

int infoFileEmpty(); ///checks if 'StudentInfo.txt' contains data


#endif // STMS_H_INCLUDED
```

Functions File

```
/*

Programmer: Alexander Williams

Date:    30/03/19

File:    STMSfunctions.c

Purpose: Definition of functions used in the main module for the

         Student Time Management System.

*/


#include <conio.h>

#include <ctype.h>

#include <stdlib.h>

#include <string.h>

#include <stdio.h>

#include "STMS.h"


///displays heading for the login section of module 'authenticate'

void loginHead()

{

  system("CLS");


  printf("  _             _            \n");

  printf(" | |___   __ _(_)_ __  _ \n");
  printf(" | |   / _ \\ / _` | | '_ \\  (_)\n");
  printf(" | |__| (_) | (_| | | | | |  _ \n");
  printf(" |_____\\___/ \\__, |_|_| |_| (_)\n");
  printf("          |___/              \n");
}

///displays heading for the sign up section of module 'authenticate'
void signupHead()
{
  system("CLS");

  printf("  ____  _              _   _       \n");
  printf(" / ___|(_) __ _ _ __   | | | |_ __ _ \n");
  printf(" \\___ \\| |/ _` | '_ \\  | | | | '_ \\  (_)\n");
  printf("  ___) | | (_| | | | | | |_| | |_) |  _ \n");
  printf(" |____/|_|\\__, |_| |_|  \\___/| .__/  (_)\n");
  printf("           |___/             |_|         \n");
}

///displays heading for the module 'displayWork'
void displayWorkHead()
{
  system("CLS");
```

```
printf("      _           _                                    _\n");

printf("/ \\   ___ ___(_) __ _ _ __  _ __ ___   ___ _ __ | |_ ___   _ \n");

printf("  / _ \\ / __/ __| |/ _` | '_ \\| '_ ` _ \\ / _ \\ '_ \\| __/ __| (_)\n");

printf(" / ___ \\\\__ \\__ \\ | (_| | | | | | | | | |  __/ | | | |_\\__ \\  _ \n");

printf(" /_/   \\_\\___/___/_|\\__, |_| |_|_| |_| |_|\\___|_| |_|\\__|___/ (_)\n");

printf("                   |___/                                      \n");




}



///displays heading for the module 'addWork'

void addWorkHead()

{

  system("CLS");



  printf("    _     _     _     _           _                                _
  \n");

  printf("    / \\   __| | __| | / \\   ___ ___(_) __ _ _ __  _ __ ___   ___ _ __ |
|_ ___   _ \n");

  printf("   / _ \\ / _` |/ _` |   / _ \\ / __/ __| |/ _` | '_ \\| '_ ` _ \\ / _ \\
'_ \\| __/ __| (_)\n");

  printf("  / ___ \\ (_| | (_| |  / ___ \\\\__ \\__ \\ | (_| | | | | | | | | |  __/
| | | |_\\__ \\  _ \n");
```

```
  printf(" /_/   \\_\\__,_|\\__,_| /_/   \\_\\___/___/_|\\__, |_| |_|_| |_|
|_|\\___|_| |_|\\__|___/ (_)\n");
  printf("                                          |___/
  \n");
}


///menu screen displayed when the user enters an invalid menu option
void errorMenuHead()
{
  system("CLS");


  printf("  __  __         _     __  __ \n");
  printf(" |  \\/  | __ _(_)_ __   |  \\/  | ___ _ __  _   _ \n");
  printf(" | |\\/| |/ _` | | '_ \\  | |\\/| |/ _ \\ '_ \\| | | |\n");
  printf(" | |  | | (_| | | | | | | |  | |  __/ | | | |_| |\n");
  printf(" |_|  |_|\\__,_|_|_| |_| |_|  |_|\\___|_| |_|\\__,_|\n");


  printf("\n[1] Edit Your Information\n");
  printf("[2] Display Your Information\n");
  printf("[3] Edit Your Schedule\n");
  printf("[4] Display Your Schedule\n");
  printf("[5] Add Assignments\n");
  printf("[6] Mark Assignments as Complete\n");
  printf("[7] Display Assignments\n");
  printf("[8] Update Login Info\n");

  printf("[9] Exit\n");

  printf("\nPlease enter an option from the menu:\n> ");

}



///default menu screen

void menuHead()

{

  system("CLS");



  printf("  __  __         _     __  __ \n");

  printf(" |  \\/  | __ _(_)_ __   |  \\/  | ___ _ __  _   _ \n");

  printf(" | |\\/| |/ _` | | '_ \\  | |\\/| |/ _ \\ '_ \\| | | |\n");

  printf(" | |  | | (_| | | | | | |  | |  __/ | | | |_| |\n");

  printf(" |_|  |_|\\__,_|_|_| |_| |_|  |_|\\___|_| |_|\\__,_|\n");



  printf("\n[1] Edit Your Information\n");

  printf("[2] Display Your Information\n");

  printf("[3] Edit Your Schedule\n");

  printf("[4] Display Your Schedule\n");

  printf("[5] Add Assignments\n");

  printf("[6] Mark Assignments as Complete\n");

  printf("[7] Display Assignments\n");
  printf("[8] Update Login Info\n");

  printf("[9] Exit\n");

  printf("\nSelect Option: \n> ");

}


///displays heading for the module 'displaySchedule'

void displayScheduleHead()

{



        printf("  ____    _           _     _           \n");

        printf(" / ___|  ___| |__   ___  __| |_   _| | ___   _ \n");

        printf(" \\___ \\ / __| '_ \\ / _ \\/ _` | | | | |/ _ \\ (_) \n");

        printf("  ___) | (__| | | |  __/ (_| | |_| | |  __/  _ \n");

        printf(" |____/ \\___|_| |_|\\___|\\__,_|\\__,_|_|\\___| (_)\n\n");




}


///displays heading for the module 'readSchedule'

void readScheduleHead()

{
  system("CLS");

  printf("  _____    _ _ _ ____        _            _     _           \n");
  printf(" | ____|__| (_) |_  / ___|  ___| |__   ___  __| |_   _| | ___   _ \n");
  printf(" |  _| / _` | | __| \\___ \\ / __| '_ \\ / _ \\/ _` | | | | |/ _ \\
(_)\n");
  printf(" | |__| (_| | | |_   ___) | (__| | | |  __/ (_| | |_| | |  __/  _ \n");
  printf(" |_____\\__,_|_|\\__| |____/ \\___|_| |_|\\___|\\__,_|\\__,_|_|\\___|
(_)\n");




}


///displays heading for the module 'readStudent'
void readStudentHead()
{
  system("CLS");

  printf("  ____  _              _           _     ___         __           \n");
  printf(" / ___|| |_ _   _  __| | ___ _ __ | |_  |_ _|_ __  / _| ___ _  \n");
  printf(" \\___ \\| __| | | |/ _` |/ _ \\ '_ \\| __|  | || '_ \\| |_ / _ \\  (_)
\n");
  printf("  ___) | |_| |_| | (_| |  __/ | | | |_   | || | | |  _| (_) |  _  \n");
  printf(" |____/ \\__|\\__,_|\\__,_|\\___|_| |_|\\__| |___|_| |_|_|  \\___/  (_)
\n");

}


///displays heading for the module 'displayStudent'

void displayStudentHead()

{

  system("CLS");



  printf(" __   __               ___         __          \n");

  printf(" \\ \\ / /__  _   _ _ __  |_ _|_ __  / _| ___  _ \n");

  printf("  \\ V / _ \\| | | | '__|  | || '_ \\| |_ / _ \\  (_)\n");

  printf("   | | (_) | |_| | |   | || | | |  _| (_) |  _ \n");

  printf("   |_|\\___/ \\__,_|_| |___|_| |_|_|  \\___/  (_)\n");

}


///awaits user input to continue program

void delay()

{

  printf("\nPress enter to continue...\n");

  getch();

}
```

```
///displays a welcome message to the user when they start the program
void welcome()
{

  printf("\n .d8888b. 88888888888 888b       d888  .d8888b.  \n");
  printf("d88P  Y88b 888    8888b   d8888 d88P  Y88b \n");
  printf("Y88b.      888    88888b.d88888 Y88b.      \n");
  printf(" \"Y888b.        888    888Y88888P888  \"Y888b.   \tWelcome to the Campion College 6B Student Time Management System!\n");
  printf("    \"Y88b.      888    888 Y888P 888      \"Y88b. \t\"The solution to all of your procrastination woes!\"\n");
  printf("    \"888 888    888  Y8P  888      \"888 \n");
  printf("Y88b  d88P 888    888   \"   888 Y88b  d88P \n");
  printf(" \"Y8888P\"      888    888    888  \"Y8888P\" \n");
  printf("\n\nMaximise window for the best experience");

  delay();
}

///function to sort the user's subjects by ascending difficulty
void sortSubs(char subject[][30],int subNum)
{
  int choice, i;
  char hold[5][30]; ///temporary array used to help sort subjects
  printf("\n");

  ///lists the user's subjects

  for(i = 0; i < subNum; i++)

  {

        printf("%i. ",i+1);

        puts(subject[i]);

  }


  ///populates the 'hold' array with the user's subjects in ascending order of
difficulty

  for(i = 0; i < subNum; i++)

  {

        printf("\nNumber %i easiest subject from the above list\n> ",i+1);

        scanf("%i", &choice);


        switch(choice)

        {

        case 1:

        strcpy(hold[i], subject[0]);

        break;

        case 2:

        strcpy(hold[i], subject[1]);
        break;

        case 3:

        strcpy(hold[i], subject[2]);

        break;

        case 4:

        strcpy(hold[i], subject[3]);

        break;

        case 5:

        strcpy(hold[i], subject[4]);

        break;

        default:

        printf("Invalid Choice");

        }

  }


  ///populates the subject member of the student structure with the contents of
hold.

  for(i = 0; i < subNum; i++)

  {

        strcpy(subject[i], hold[i]);

  }

}
```

```
///reads data into the members of a 'Student' structure

void readStudent(Student *st)

{

  int choice, i;



  readStudentHead();



  strcpy(st->subject[0], "Communication Studies"); ///assigns the compulsory subject
to the subject member of the student structure



  ///prompts for and reads the student's first and last name as well as their number
subjects into the respective student structure members

  printf("\nFirst Name> ");

  fflush(stdin);

  gets(st->fName);

  printf("\nLast Name> ");

  gets(st->lName);

  printf("\nNumber of Subjects (exclusive of Communication Studies)\n[3/4]> ");

  scanf("%i", &(st->subjectNum));



  ///ensures that only 3 or 4 is entered for the subject number

  while((st->subjectNum != 3) && (st->subjectNum != 4))

  {

        system("CLS");
        printf("Please enter 3 or 4\n> ");

        fflush(stdin);

        scanf("%i", &st->subjectNum);


  }


  (st->subjectNum)++; ///increments the subject number member so as to include the
compulsory subject in the overall number of subjects


  ///list of subjects offered for Campion College Lower Sixth (6B) Students

  printf("\n1. Physics                 ");

  printf("5. Pure Mathematics          ");

  printf("9. French                    ");

  printf("13. Principles of Accounts\n");

  printf("2. Chemistry                 ");

  printf("6. Digital Media             ");

  printf("10. Spanish                  ");

  printf("14. Sociology\n");

  printf("3. Biology                   ");

  printf("7. Management of Business   ");

  printf("11. Economics                ");

  printf("15. Geography\n");

  printf("4. Computer Science          ");
  printf("8. Literatures in English   ");

  printf("12. Law                       ");

  printf("16. History\n");


  ///prompts for and assigns the user's subjects based on the number of subjects
they do (4 or 5)

  printf("\nSelect %i subjects:", (st->subjectNum)-1);

  for(i = 1; i < st->subjectNum; i++)

  {

        printf("\n> " );

        scanf("%i", &choice);

        switch(choice)

        {

        case 1:

        strcpy(st->subject[i], "Physics");

        break;

        case 2:

        strcpy(st->subject[i], "Chemistry");

        break;

        case 3:

        strcpy(st->subject[i], "Biology");

        break;

        case 4:
```

```
strcpy(st->subject[i], "Computer Science");

break;

case 5:

strcpy(st->subject[i], "Pure Mathematics");

break;

case 6:

strcpy(st->subject[i], "Digital Media");

break;

case 7:

strcpy(st->subject[i], "Management of Business");

break;

case 8:

strcpy(st->subject[i], "Literatures in English");

break;

case 9:

strcpy(st->subject[i], "French");

break;

case 10:

strcpy(st->subject[i], "Spanish");

break;

case 11:

strcpy(st->subject[i], "Economics");

break;
```

```
	case 12:

	strcpy(st->subject[i], "Law");

	break;

	case 13:

	strcpy(st->subject[i], "Principles of Accounts");

	break;

	case 14:

	strcpy(st->subject[i], "Sociology");

	break;

	case 15:

	strcpy(st->subject[i], "Geography");

	break;

	case 16:

	strcpy(st->subject[i], "History");

	break;

	}

}


readStudentHead();


sortSubs(st->subject, st->subjectNum);


readStudentHead();
```

```
///prompts for and reads the user's number of activities

printf("\nNumber of Co-Curricular Activities\n[1-5]> ");

scanf("%i", &st->activityNum);


///ensures that the number entered is between 4 and 5

while((st->activityNum>5) || (st->activityNum<1))

{

        printf("Please enter a number between 1 and 5 \n> ");

        fflush(stdin);

        scanf("%i", &st->activityNum);

}


readStudentHead();


///list of Clubs and Sports offered by Campion College

printf("\n1.  Aeronautics Club             \t");

printf("21. I.S.C.F.                 \t");

printf("41. Sixth Form Association\n");

printf("2.  Angels of Love           \t");

printf("22. Interact Club            \t");

printf("42. Software Engineering Club\n");

printf("3.  Animal Club              \t");
printf("23. Key Club                 \t");

printf("43. Student Council\n");

printf("4.  Animation                \t");

printf("24. Lego Yuh Mind Robotics Club\t");

printf("44. Students for Democracy\n");

printf("5.  Art Club                 \t");

printf("25. Leo Club                 \t");

printf("45. TED - ED\n");

printf("6.  Campion Coders           \t");

printf("26. Mathematics Club         \t");

printf("46. The Students' Voice\n");

printf("7.  Campion Theatre Ensemble   \t");

printf("27. Media and Production Club  \t");

printf("47. Tourism Action Club\n");

printf("8.  Catholic Club            \t");

printf("28. Medics Club              \t");

printf("48. United Nations Club\n");

printf("9.  Chapel Choir             \t");

printf("29. Ministry Outreach Program  \t");

printf("49. Young Entrepreneurial Society\n");

printf("10. Christian Life Community   \t");

printf("30. Modern Language Club     \t");

printf("50. Basketball\n");
printf("11. Computer and Media Club  \t");

printf("31. Music Club               \t");

printf("51. Chess\n");

printf("12. Dance Society            \t");

printf("32. Chords                   \t");

printf("52. Fitness & Weightlifting\n");

printf("13. Debating Society         \t");

printf("33. Drum Ensemble            \t");

printf("53. Football\n");

printf("14. Disaster Preparedness    \t");

printf("34. Steel Band               \t");

printf("54. Hockey\n");

printf("15. D.I.Y.                   \t");

printf("35. Peer Counselling         \t");

printf("55. Lawn Tennis\n");

printf("16. Engineering Club         \t");

printf("36. Rangers                  \t");

printf("56. Swimming\n");

printf("17. Gavel Club               \t");

printf("37. Readers Association      \t");

printf("57. Table Tennis\n");

printf("18. Gourmet Club             \t");

printf("38. Red Cross                \t");
```

```
  printf("58. Track and Field\n");
  printf("19. Girl Code                \t");
  printf("39. Science Club             \t");
  printf("59. Volleyball\n");
  printf("20. Green Generation         \t");
  printf("40. Sign Language Club       \t");
  printf("60. Water Polo\n");

  ///prompts for and assigns the user's activities based on the number of activities
they do
  printf("\nSelect %i activities(y): ", st->activityNum);
  for(i = 0; i < st->activityNum; i++)
  {
      printf("\n> ");
      scanf("%i", &choice);

      ///ensures the user's choice is a number from 1-60
      while(((choice < 1)||(choice > 60)))
      {
      printf("\nPlease enter a number between 1 and 60\n> ");
      fflush(stdin);
      scanf("%i", &choice);
      }
```

```
switch(choice)

{

case 1:

             strcpy(st->activity[i], "Aeronautics Club");

             break;

       case 2:

             strcpy(st->activity[i], "Angels of Love");

             break;

       case 3:

             strcpy(st->activity[i], "Animal Club");

             break;

       case 4:

             strcpy(st->activity[i], "Animation");

             break;

       case 5:

             strcpy(st->activity[i], "Art Club");

             break;

       case 6:

             strcpy(st->activity[i], "Campion Coders");

             break;

       case 7:

             strcpy(st->activity[i], "Campion Theatre Ensemble");

             break;
```

```
case 8:

      strcpy(st->activity[i], "Catholic Club");

      break;

case 9:

      strcpy(st->activity[i], "Chapel Choir");

      break;

case 10:

      strcpy(st->activity[i], "Christian Life Community");

      break;

case 11:

      strcpy(st->activity[i], "Computer and Media Club");

      break;

case 12:

      strcpy(st->activity[i], "Dance Society");

      break;

case 13:

      strcpy(st->activity[i], "Debating Society");

      break;

case 14:

      strcpy(st->activity[i], "Disaster Preparedness");

      break;

case 15:

      strcpy(st->activity[i], "D.I.Y.");
```

```
        break;
    case 16:
        strcpy(st->activity[i], "Engineering Club");
        break;
    case 17:
        strcpy(st->activity[i], "Gavel Club");
        break;
    case 18:
        strcpy(st->activity[i], "Gourmet Club");
        break;
    case 19:
        strcpy(st->activity[i], "Girl Code");
        break;
    case 20:
        strcpy(st->activity[i], "Green Generation");
        break;
    case 21:
        strcpy(st->activity[i], "I.S.C.F.");
        break;
    case 22:
        strcpy(st->activity[i], "Interact Club");
        break;
    case 23:
        strcpy(st->activity[i], "Key Club");

        break;

    case 24:

        strcpy(st->activity[i], "Lego Yuh Mind Robotics Club");

        break;

    case 25:

        strcpy(st->activity[i], "Leo Club");

        break;

    case 26:

        strcpy(st->activity[i], "Mathematics Club");

        break;

    case 27:

        strcpy(st->activity[i], "Media and Production Club");

        break;

    case 28:

        strcpy(st->activity[i], "Medics Club");

        break;

    case 29:

        strcpy(st->activity[i], "Ministry Outreach Program");

        break;

    case 30:

        strcpy(st->activity[i], "Modern Language Club");

        break;
```

```
case 31:

      strcpy(st->activity[i], "Music Club");

      break;

case 32:

      strcpy(st->activity[i], "Chords");

      break;

case 33:

      strcpy(st->activity[i], "Drum Ensemble");

      break;

case 34:

      strcpy(st->activity[i], "Steel Band");

      break;

case 35:

      strcpy(st->activity[i], "Peer Counselling");

      break;

case 36:

      strcpy(st->activity[i], "Rangers");

      break;

case 37:

      strcpy(st->activity[i], "Readers Association");

      break;

case 38:

      strcpy(st->activity[i], "Red Cross");
```

```
        break;

    case 39:

        strcpy(st->activity[i], "Science Club");

        break;

    case 40:

        strcpy(st->activity[i], "Sign Language Club");

        break;

    case 41:

        strcpy(st->activity[i], "Sixth Form Association");

        break;

    case 42:

        strcpy(st->activity[i], "Software Engineering Club");

        break;

    case 43:

        strcpy(st->activity[i], "Student Council");

        break;

    case 44:

        strcpy(st->activity[i], "Students for Democracy");

        break;

    case 45:

        strcpy(st->activity[i], "TED - ED");

        break;

    case 46:
        strcpy(st->activity[i], "The Students' Voice");

        break;

    case 47:

        strcpy(st->activity[i], "Tourism Action Club");

        break;

    case 48:

        strcpy(st->activity[i], "United Nations Club");

        break;

    case 49:

        strcpy(st->activity[i], "Young Entrepreneurial Society");

        break;

    case 50:

        strcpy(st->activity[i], "Basketball");

        break;

    case 51:

        strcpy(st->activity[i], "Chess");

        break;

    case 52:

        strcpy(st->activity[i], "Fitness & Weightlifting");

        break;

    case 53:

        strcpy(st->activity[i], "Football");

        break;
```

```
case 54:
      strcpy(st->activity[i], "Hockey");
      break;
case 55:
      strcpy(st->activity[i], "Lawn Tennis");
      break;
case 56:
      strcpy(st->activity[i], "Swimming");
      break;
case 57:
      strcpy(st->activity[i], "Table Tennis");
      break;
case 58:
      strcpy(st->activity[i], "Track and Field");
      break;
case 59:
      strcpy(st->activity[i], "Volleyball");
      break;
case 60:
      strcpy(st->activity[i], "Water Polo");
      break;
default:
      printf("Invalid option");
```

```
        }
    }
}

///displays the data stored in the members of a 'Student' structure
void displayStudent(Student st)
{
    int i;

    displayStudentHead();

    printf("\nYour Name\n> %s %s\n", st.fName, st.lName); ///displays user's name

    ///displays user's subjects (in ascending difficulty)
    printf("\nYour Subjects: \n");
    for(i = 0;i < st.subjectNum; i++)
    {
        printf(">");
        puts(st.subject[i]);
    }

    ///displays the activities done by the user
    printf("\nYour Activities: \n");
```

```
   for(i = 0;i < st.activityNum; i++)

   {

         printf(">");

         puts(st.activity[i]);

   }

}


///reads data into the elements of a 3D array representing the schedule of the user

void formatSchedule(char schedule[][5][30])

{

   int r,c;


   ///populates the array column by column with arbitrary values

   for(c = 0; c < 5; c++)

   {

         for(r = 0; r < 6; r++)

         {

         strcpy(schedule[r][c], "NO CLASS"); ///assigns the first 6 elements in every
column with the arbitrary value 'NO CLASS'

         }


         strcpy(schedule[r][c], "NO ACTIVITY"); ///assigns the last element in every
column with the arbitrary value 'NO ACTIVITY'

   }
```

```
}


///formats the members of a 'Date' structure with arbitrary data

void formatDate(Date *dt)

{

  dt->day = 0;

  dt->month = 0;

  dt->year = 2019;

}


///formats the members of an 'Assignment' structure with arbitrary data

void formatAssignment(Assignment *as)

{

  strcpy(as->name, "--");

  formatDate(&(as->dateGiven)); ///formats the date given member of the assignment
structure

  formatDate(&(as->dateDue)); ///formats the date due member of the assignment
structure

  strcpy(as->type, "--");

  strcpy(as->subject, "--");

  as->time = 0;

}


///formats the members of a 'Student' structure with arbitrary data
```

```
void formatStudent(Student *st)

{

  int i;



  strcpy(st->fName,"--");

  strcpy(st->lName,"--");

  st->subjectNum = 0;

  for(i = 0; i < 5; i++)

  {

        strcpy(st->subject[i],"--");

  }

  st->activityNum = 0;

  for(i = 0; i < 5; i++)

  {

        strcpy(st->activity[i],"--");

  }

  st->assignmentNum = 0;



  ///formats the elements of the assignment member

  for(i = 0; i < 10; i++)

  {

        formatAssignment(&(st->assignment[i]));

  }
  ///formats the users schedule

  formatSchedule(st->schedule);

}


///reads data into the elements of a 3D array representing the schedule of the user

void readSchedule(Student *st)

{

  int i, r, c, choice;

  char day[5][10] = {{"Monday"}, {"Tuesday"}, {"Wednesday"}, {"Thursday"},
{"Friday"}};


  formatSchedule(st->schedule); ///ensures that the schedule is formatted before it
is read


  readScheduleHead();


  ///populates schedule array column by column

  for(c = 0; c < 5; c++)

  {

        printf("\nEnter Classes for %s:\n\n", day[c]); /// prints the day of the
week the user is entering data into


        ///prints the user's subjects as options
        for(i = 0; i < st->subjectNum; i++)

        {

        printf("%i. ", i+1);

        puts(st->subject[i]);

        }

        printf("%i. No Class\n", i+1);



        for(r = 0; r < 6; r++)

        {

        printf("\nPeriod %i Class\n> ", r+1); ///reads the user's choice for class

        scanf("%i", &choice);



        /// assigns the subject to the current element position based on the user's
option

        if(st->subjectNum == 4)

        {

              switch(choice)

              {

              case 1:

                    strcpy(st->schedule[r][c], st->subject[0]);

                    break;

              case 2:

                    strcpy(st->schedule[r][c], st->subject[1]);
```

```
            break;
        case 3:
            strcpy(st->schedule[r][c], st->subject[2]);
            break;
        case 4:
            strcpy(st->schedule[r][c], st->subject[3]);
            break;
        case 5:
            break;
        default:
            printf("Invalid Choice");

        }
    }
    else
    {
        switch(choice)
        {
        case 1:
            strcpy(st->schedule[r][c], st->subject[0]);
            break;
        case 2:
            strcpy(st->schedule[r][c], st->subject[1]);
```

```
                break;
            case 3:
                strcpy(st->schedule[r][c], st->subject[2]);
                break;
            case 4:
                strcpy(st->schedule[r][c], st->subject[3]);
                break;
            case 5:
                strcpy(st->schedule[r][c], st->subject[4]);
                break;
            case 6:
                break;
            default:
                printf("Invalid Choice");
            }
        }
        }

        readScheduleHead();


        if(st->activityNum != 0) ///verifies that the user does activities before
prompting for the activity
```

```
        {
        printf("\nActivity for %s: \n\n", day[c]); ///prints the day of the week the
user is entering data into

        ///prints the user's activities as options
        for(i = 0; i < st->activityNum; i++)
        {
              printf("%i. ", i+1);
              puts(st->activity[i]);
        }
        printf("%i. No Activity\n", i+1);

        ///prompts and reads the users choice of activity
        printf("\nActivity\n> ");
        scanf("%i", &choice);

        ///assigns the activity for the current day based on the user's choice
        switch(choice)
        {
              case 1:
                    strcpy(st->schedule[6][c], st->activity[0]);
                    break;
              case 2:
```

```
        if(strcmp(st->activity[1], "--") == 0)
        {
        break;
        }
        else
        {
        strcpy(st->schedule[6][c], st->activity[1]);
        break;
        }
    case 3:
        if(strcmp(st->activity[2], "--") == 0)
        {
        break;
        }
        else
        {
        strcpy(st->schedule[6][c], st->activity[2]);
        break;
        }
    case 4:
        if(strcmp(st->activity[3], "--") == 0)
        {
        break;
```

```
                }
                else
                {
                strcpy(st->schedule[6][c], st->activity[3]);
                break;
                }
        case 5:
                if(strcmp(st->activity[4], "--") == 0)
                {
                break;
                }
                else
                {
                strcpy(st->schedule[6][c], st->activity[4]);
                break;
                }
        case 6:
                break;
        default:
                printf("Invalid Option");
}
}
readScheduleHead();
```

```
  }
}

///prints/displays the elements of a 3D array representing the schedule of the user
void displaySchedule(char schedule[][5][30])
{
  int r, c, i;
  char day[6][10] = {{""},{"Monday"}, {"Tuesday"}, {"Wednesday"}, {"Thursday"},
{"Friday"}};
  char period[7][12] =
{{"8:00-9:10"},{"9:10-10:20"},{"10:20-11:30"},{"11:30-12:40"},{"12:40-1:50"},{"1:50-
3:00"},{"Activity"}};

  displayScheduleHead();

  ///displays the array row by row so as to get even spacing between columns
  for(i = 0; i < 6; i++)
  {
        printf("%-30s", day[i]);
  }
  printf("\n");
  for(r = 0; r < 7; r++)
  {
        printf("%-30s", period[r]);
```

```
          for(c = 0; c < 5; c++)
          {
          printf("%-30s", schedule[r][c]);
          }
          printf("\n");
   }
}

///reads data into the members of a 'Date' structure
Date readDate()
{
   Date dt;

   ///prompts and reads day component for the date
   printf("\nDay> ");
   scanf("%i", &dt.day);

   ///ensures the day entered is from 1-31
   while(dt.day > 31 || dt.day < 1)
   {
          printf("Please enter a number between 1 and 31\n> ");
          fflush(stdin);
```

```
        scanf("%i", &dt.day);
    }

    ///prompts and reads month component for the datei
    printf("Month> ");
    scanf("%i", &dt.month);
    while((dt.month > 12 || dt.month < 1))
    {
        printf("Please enter a number between 1 and 12\n> ");
        fflush(stdin);
        scanf("%i", &dt.month);
    }

    dt.year = 2019; ///assumes that all date values use the year 2019

    return dt; ///returns the created date structure to the calling function
}

///reads data into the members of an 'Assignment' structure
 Assignment readAssignment(Student st)
{
    Assignment as;
    char choice;
```

```
int i;


///prompts for and reads the assignment's name, date given and due date
fflush(stdin);
printf("\nAssignment Name> ");
gets(as.name);
printf("\nDate Given: ");
as.dateGiven = readDate();
printf("\nDate Due: ");
as.dateDue = readDate();


addWorkHead();


///prompts for and reads the assignment's type
printf("\n1. HW");
printf("\n2. CW");
printf("\n3. TEST\n");
printf("\nAssignment Type> ");
fflush(stdin);
choice = getch();
///ensures that a number from 1-3 is entered
while(
      choice != '1' &&
	choice != '2' &&

	choice != '3'

	)

{

	printf("Please enter a number from 1-3> ");

	choice = getch();

}

switch(choice)

{

	case '1':

	strcpy(as.type, "HW");

	break;

	case '2':

	strcpy(as.type, "CW");

	break;

	case '3':

	strcpy(as.type, "TEST");

	break;

}


///displays the user's subjects

printf("\n\n");

for(i = 0; i < st.subjectNum; i++)
```

```
  {
        printf("%i. ", i+1);
        puts(st.subject[i]);
  }

  ///prompts for and reads the assignment's subject and therefore assigns an
estimated time based on the subject selected
  printf("\nAssignment Subject> ");
  fflush(stdin);
  choice = getch();
  if(st.subjectNum == 5)
  {
        ///ensures that the number entered is from 1-5
        while(
        choice != '1' &&
        choice != '2' &&
        choice != '3' &&
        choice != '4' &&
        choice != '5'
        )
        {
        printf("Please enter a number from 1-5> ");
        choice = getch();
```

```
}

switch(choice)

{

case '1':

        strcpy(as.subject, st.subject[0]);

        if((strcmp(as.type, "HW"))==0)

        {

                as.time = 30;

        }

        else

        {

                if((strcmp(as.type, "CW"))==0)

                {

                as.time = 45;

                }

                else

                {

                as.time = 60;

                }

        }

        break;

case '2':

        strcpy(as.subject, st.subject[1]);
```

```
		if((strcmp(as.type, "HW"))==0)
		{
			as.time = 45;
		}
		else
		{
			if((strcmp(as.type, "CW"))==0)
			{
			as.time = 60;
			}
			else
			{
			as.time = 75;
			}
		}
		break;
	case '3':
		strcpy(as.subject, st.subject[2]);
		if((strcmp(as.type, "HW"))==0)
		{
			as.time = 60;
		}
		else
```

```
        {
                if((strcmp(as.type, "CW"))==0)
                {
                as.time = 75;
                }
                else
                {
                as.time = 90;
                }
        }
        break;
case '4':
        strcpy(as.subject, st.subject[3]);
        if((strcmp(as.type, "HW"))==0)
        {
                as.time = 75;
        }
        else
        {
                if((strcmp(as.type, "CW"))==0)
                {
                as.time = 90;
                }
                else
                {
                as.time = 105;
                }
        }
        break;
case '5':
        strcpy(as.subject, st.subject[4]);
        if((strcmp(as.type, "HW"))==0)
        {
                as.time = 90;
        }
        else
        {
                if((strcmp(as.type, "CW"))==0)
                {
                as.time = 105;
                }
                else
                {
                as.time = 120;
                }
        }
```

```
            break;
        default:
            printf("Invalid Choice");
        }
    }
    else
    {
        ///ensures that the number entered is from 1-4
        while(
        choice != '1' &&
        choice != '2' &&
        choice != '3' &&
        choice != '4'
        )
        {
        printf("Please enter a number from 1-4> ");
        choice = getch();
        }
        switch(choice)
        {
        case '1':
            strcpy(as.subject, st.subject[0]);
            if((strcmp(as.type, "HW"))==0)
              {
                  as.time = 30;
              }
              else
              {
                  if((strcmp(as.type, "CW"))==0)
                  {
                  as.time = 45;
                  }
                  else
                  {
                  as.time = 60;
                  }
              }
              break;
        case '2':
              strcpy(as.subject, st.subject[1]);
              if((strcmp(as.type, "HW"))==0)
              {
                  as.time = 45;
              }
              else
              {
```

```
                        if((strcmp(as.type, "CW"))==0)
                        {
                        as.time = 60;
                        }
                        else
                        {
                        as.time = 75;
                        }
                }
                break;
        case '3':
                strcpy(as.subject, st.subject[2]);
                if((strcmp(as.type, "HW"))==0)
                {
                        as.time = 60;
                }
                else
                {
                        if((strcmp(as.type, "CW"))==0)
                        {
                        as.time = 75;
                        }
                        else
                        {
                        as.time = 90;
                        }
                }
                break;
        case '4':
                strcpy(as.subject, st.subject[3]);
                if((strcmp(as.type, "HW"))==0)
                {
                        as.time = 75;
                }
                else
                {
                        if((strcmp(as.type, "CW"))==0)
                        {
                        as.time = 90;
                        }
                        else
                        {
                        as.time = 105;
                        }
                }
                break;
        default:

              printf("\nInvalid Choice: ");

        }

  }


  ///reads a new value for time if the user wants to

  printf("\n\nSuggested time for preparation/completion is %i minutes. Would you
like to override this time?", as.time);

  printf("\n[Y/N]> ");

  fflush(stdin);

  choice = getch();

  ///ensures that either a y or n is entered

  while(

        (choice != 'Y' && choice != 'y') &&

        (choice != 'N' && choice != 'n')

        )

  {

        printf("Please enter Y or N> ");

        choice = getch();

  }

  switch(choice)

  {

  case 'Y': case 'y':
        printf("New Time in minutes> ");
        scanf("%i", &as.time);
        break;
        case 'N': case 'n':
        break;
        default:
        printf("Invalid Choice");
  }

  return as;
}

///populates the 'assignment' member of a 'Student' structure with a user determined
amount of elements
void addWork(Student *st)
{
  int i = st->assignmentNum, num;

  addWorkHead();

  printf("\nNumber of assignments to add> ");
  scanf("%i", &num);
```

```
   ///ensures that the user doesn't enter more assignments than which can be stored
   while((num >= (10-st->assignmentNum)))
   {
          printf("\nPlease enter a number less than or equal to %i",
10-st->assignmentNum);
          printf("\n> ");
          fflush(stdin);
          scanf("%i", &num);
   }

   num = i + num; ///obtains terminal index by adding the number of assignments to
the number of assignments being read

   ///prompts for and reads assignments into elements of the assignment member
   for(; i < num ; i++)
   {
          addWorkHead();
          fflush(stdin);
          st->assignment[i] = readAssignment(*st);
          st->assignmentNum++;
   }
}

///formats the elements of the 'assignment' member of a 'Student' structure
```

```
void deleteWork(Student *st)
{
  int i;

  for(i = 0; i < st->assignmentNum; i++)
  {
        formatAssignment(&(st->assignment[i]));
  }

  st->assignmentNum = 0; ///resets the number of assignments to 0

  printf("Assignments Have Been Cleared");
}

///displays the data stored in the members of a 'Date' structure
void displayDate(Date dt)
{
  printf("%i/%i/%i", dt.day, dt.month, dt.year);
}

///displays the data stored in the members of an 'Assignment' structure
void displayAssignment(Assignment as)
{
  printf("\nAssignment Name> %s", as.name);

  printf("\nDate Given: ");

  displayDate(as.dateGiven);

  printf("\nDate Due: ");

  displayDate(as.dateDue);

  printf("\nAssignment Type: %s", as.type);

  printf("\nAssignment Subject: %s", as.subject);

  printf("\nEstimated Time for Completion/Preparation: %i mins", as.time);

  printf("\n--------------------------------------------------------\n");

}


///formats the elements of the 'assignment' member of a 'Student' structure

void displayWork(Student st)

{

  int i;


  displayWorkHead();


  for(i = 0; i<st.assignmentNum; i++)

  {

        printf("\nAssignment #%i", i+1);

        displayAssignment(st.assignment[i]);

  }
```

```
}


///displays a farewell message to the user when they exit the program through the
menu
void farewell()
{


  printf(" .d8888b. 88888888888 888b   d888  .d8888b.  \n");
  printf("d88P  Y88b 888   8888b   d8888 d88P  Y88b \n");
  printf("Y88b.      888   88888b.d88888 Y88b.     \n");
  printf(" \"Y888b.        888   888Y88888P888  \"Y888b.   \n");
  printf("    \"Y88b.      888   888 Y888P 888     \"Y88b. \n");
  printf("    \"888  888   888  Y8P  888     \"888 \n");
  printf("Y88b  d88P 888   888   \"   888 Y88b  d88P \n");
  printf(" \"Y8888P\"      888   888   888  \"Y8888P\" \n");
  printf("Exiting System...");
}


///displays menu options and calls their respective functions
void menu(Student st)
{
  char choice;
```

```
do ///repeat until exit option is triggered

{


        menuHead();


        fflush(stdin);

        choice = getch();


        ///ensures the choice is from numbers 1-9

        while(choice !='1' &&

                choice !='2' &&

                choice !='3' &&

                choice !='4' &&

                choice !='5' &&

                choice !='6' &&

                choice !='7' &&

                choice !='8' &&

                choice !='9'

                )

        {

        errorMenuHead();

        choice = getch();

        }
        switch(choice)

        {

        case '1':

                system("CLS");

                readStudent(&st); ///reads data into the members of a student
structure

                writeStudentFile(st); ///writes the recently modified student
structure to a random access file

                break;

        case '2':

                system("CLS");



                st = readStudentFile(); ///reads a 'Student' structure from a random
access file and assigns it to the student structure



                ///ensures that the student structure isn't empty/formatted

                if(strcmp(st.fName, "--") != 0)

                {

                        displayStudent(st);

                        delay();

                }

                else

                {

                        printf("Please Enter Your Information First!");
                        delay();
                }
                break;
        case '3':
                system("CLS");
                st = readStudentFile();

                ///ensures that the student structure isn't empty/formatted
                if(st.subjectNum != 0)
                {
                        readSchedule(&st);
                        writeStudentFile(st);
                }
                else
                {
                        printf("Please Enter Your Information First!");
                        delay();
                }
                break;
        case '4':
                system("CLS");
                st = readStudentFile();
                ///ensures that the schedule isn't empty/formatted
                if(strcmp(st.schedule[0][0], "NO CLASS") != 0)
                {
                        displaySchedule(st.schedule);
                        delay();
                }
                else
                {
                        printf("Please Enter Your Schedule First!");
                        delay();
                }
                break;
        case '5':
                system("CLS");
                st = readStudentFile();

                ///ensures that the student structure isn't formatted and that the
    number of assignments does not exceed the capacity
                if(st.subjectNum != 0 && st.assignmentNum <10)
                {
                        addWork(&st);
                        writeStudentFile(st);
                }
```

```
        else
        {
            if(st.assignmentNum == 10)
            {
            printf("Unable to store anymore assignments!");
            delay();
            }
            else
            {
            printf("Please Enter Your Information First!");
            delay();
            }
        }
        break;
    case '6':
        system("CLS");
        st = readStudentFile();

        ///ensures that the student structure isn't empty/formatted
        if(st.assignmentNum != 0 && st.subjectNum != 0)
        {
            deleteWork(&st);
            writeStudentFile(st);
            }
            else
            {
                  printf("\nAssignments are clear! No pending assignments
due...");
                  delay();
            }
            break;
      case '7':
            system("CLS");
            st = readStudentFile();

            ///ensures the assignments are not empty/formatted
            if(strcmp(st.assignment[0].name, "--") != 0)
            {
                  displayWork(st);
                  delay();
            }
            else
            {
                  printf("No pending assignments due");
                  delay();
            }
```

```
                break;

        case '8':

                system("CLS");

                updateLogin(); ///updates the username and password of the user that
is stored on file

                break;

        case '9':

                system("CLS");

                farewell(); ///displays a farewell message to the user when they exit
the program through the menu

                break;

        default:

                printf("Please select an option from the menu");

        }

   }

   while(choice != '9');

}


///masks the user's input with '.' whenever they enter a password

void getPassword(char password[])

{

   char ch;

   int i = 0;
```

```
while(1)

{

       ch = getch();

       if (ch == 13)

       {

       password[i] = '\0';

       break;

       }

       else if (ch == 8)

       {

              if (i > 0)

              {

                     i--;

                     printf("\b \b");

              }

       }

       else if ((ch == 37)||(ch == 38)||(ch == 39)||(ch == 40))

       {

              printf("\b \b");

       }

       else

       {

              password[i] = ch;
```

```
                i++;

                printf(".");

            }

            }

}


///displays a login or sign-up screen and determines if the user is granted access

void authenticate(int *loggedIn)

{

  char nameOnFile[25], passwordOnFile[25], uname[25], pwd[25];

  FILE *fpRead, *fpWrite;

  long size;


  system("CLS");


  if((fpRead = fopen("Login.txt", "a+")) != NULL) ///opens 'Login.txt' for reading
as well as appending

  {

        fseek(fpRead, 0, SEEK_END);

        size = ftell(fpRead);


        ///checks if to sign up or login

        if(size != 0)/// ensures that something is on file
        {

        fseek(fpRead, 0, SEEK_SET);

        fscanf(fpRead, "%s %s", nameOnFile, passwordOnFile); ///reads and stores the
username and password that is on file


        ///login screen


        loginHead();

        printf("\nUsername> ");

        scanf("%s", uname);

        printf("\nPassword> ");

        getPassword(pwd);


        ///ensures that access is not granted unless the name and password entered
are identical to the name and password on file

        while(strcmp(uname, nameOnFile) != 0 || strcmp(pwd, passwordOnFile) != 0)

        {

              loginHead();

              printf("Incorrect Password or Username...Try Again\n");

              printf("\nUsername> ");

              scanf("%s", uname);

              printf("\nPassword> ");

              getPassword(pwd);

        }
```

```
*loggedIn = 1; ///access granted

}

else

{

///sign up screen

do

{



	signupHead();

	printf("\nUsername> ");

	scanf("%s", uname);

	printf("\nPassword> ");

	getPassword(pwd);

}

while(strcmp(uname, "") == 0 || strcmp(pwd, "") == 0);



///write the new user to file

if((fpWrite = fopen("Login.txt", "w")) != NULL)

{

	fprintf(fpWrite, "%s %s", uname, pwd);

	*loggedIn = 1;///user logged in

	fclose(fpWrite);
```

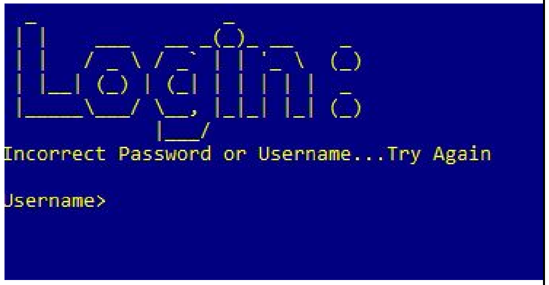
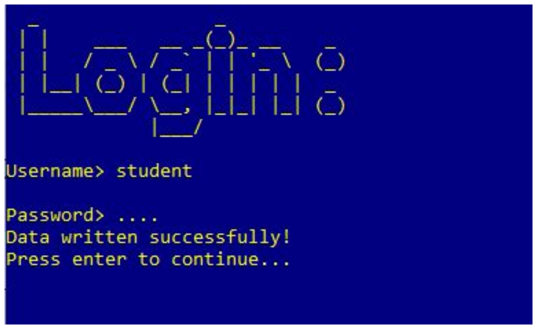
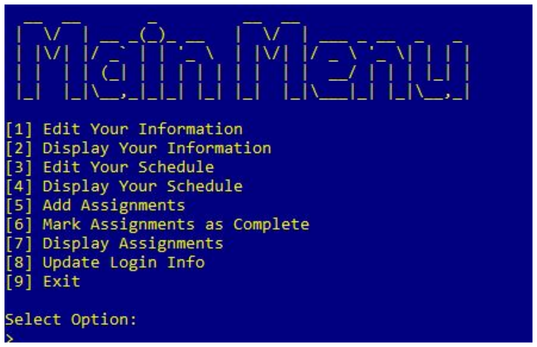
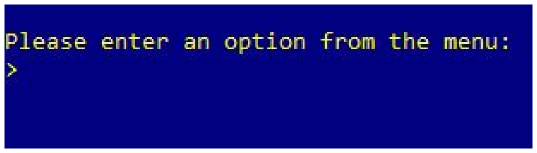
```
		}else
		{
			*loggedIn = 0;///user not logged in
		}
		}
		fclose(fpRead);
	}
	else
	{
		*loggedIn = 0;///user not logged in
	}
}

///updates the username and password of the user that is stored on file
void updateLogin()
{
	char uname[25], pwd[25];
	FILE *fp;

	system("cls");

	if((fp = fopen("Login.txt", "w")) != NULL)
	{
        do

        {



        printf("  _   _          _      _     _                  _           \n");

        printf(" | | | |_ __   __| | __ _| |_ ___  | |    ___   __ _(_)_ __  _ \n");

        printf(" | | | | '_ \\ / _` |/ _` | __/ _ \\ | |   / _ \\ / _` | | '_ \\
(_)\n");

        printf(" | |_| | |_) | (_| | (_| | ||  __/ | |__| (_) | (_| | | | | |  _
\n");

        printf("  \\___/| .__/ \\__,_|\\__,_|\\__\\___| |_____\\___/ \\__, |_|_| |_|
(_)\n");

        printf("    |_|                                   |___/              \n");



        ///prompts and reads the new username and password

        printf("\nNew Username> ");

        scanf("%s", uname);

        printf("\nNew Password> ");

        getPassword(pwd);

        }

        while(strcmp(uname, "") == 0 || strcmp(pwd, "") == 0);



        fprintf(fp, "%s %s", uname, pwd); ///writes the new username and password to
file

        fclose(fp);
        printf("\n\nUsername and Password successfully updated!");

        delay();

    }else

    {

        printf("\n CANNOT UPDATE AT THIS TIME...");

        getch();

    }

}


///updates the username and password of the user that is stored on file

void writeStudentFile(Student st)

{

    FILE *fp;


    if((fp = fopen("StudentInfo.txt", "wb")) == NULL) ///opens 'StudentInfo.txt' for
random/binary writing

    {

        printf("ERROR: Required File Could Not Be Opened");

        delay();

    }

    else

    {
            fwrite(&st, sizeof(Student), 1, fp); ///writing student structure to file

            fprintf(fp, "\n");

            fclose(fp);

            printf("\nData written successfully!");

            delay();


    }

}


///reads a 'Student' structure from a random access file and returns it

Student readStudentFile()

{

    FILE *fp;

    Student st;


    if((fp = fopen("StudentInfo.txt", "rb")) == NULL) ///opens 'StudentInfo.txt' for
binary/random reading

    {

            printf("ERROR: Required File Could Not Be Opened");

    }

    else

    {

            fread(&st, sizeof(Student), 1, fp); ///reads student structure from file
into a structure of the same type (student)
```

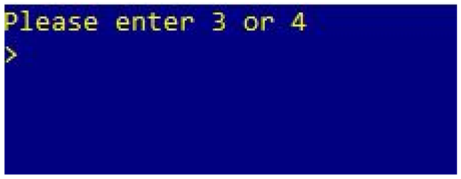
```
        fclose(fp);
  }


  return st; ///returns what is read from file
}


///checks if 'StudentInfo.txt' contains data
int infoFileEmpty()
{
  FILE *fp = fopen("StudentInfo.txt","ab");


  long fsize = 0;


  if(fp != NULL)
  {
         fseek(fp, 0, SEEK_END); /// Goes to end of the file
         fsize = ftell(fp); ///determines size of file based on position (end of
file) and stores it in a variable
        rewind(fp); ///returns to the beginning of the file
         fclose(fp);
         return (fsize == 0) ? 0 : 1; ///returns 0 or 1 whether the file is empty or
not
  }
  else
  {
        return -1; ///returns -1 in case of an error
  }
}
```

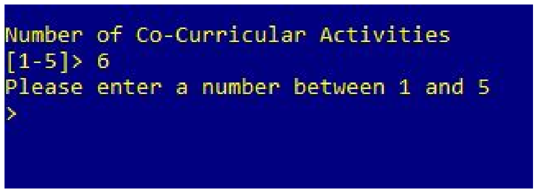
Main File

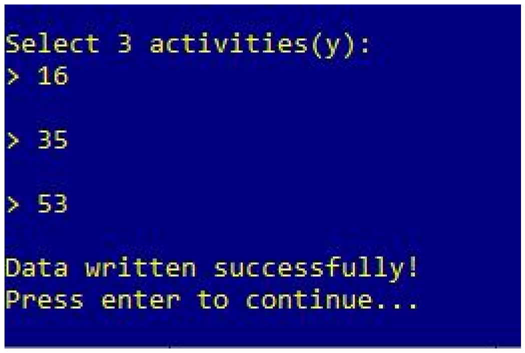
```
/*

Programmer: Alexander Williams

Date:    30/03/19

File:    main.c

Purpose: Main/Driver module for the Student Time Management System.

*/



#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include "STMS.h"



int main()

{

  Student student;

  int loggedIn;



  system("COLOR 1E"); ///console background colour - blue, console text colour -
yellow



  welcome(); ///displays a welcome message to the user when they start the program
  authenticate(&loggedIn); ///displays a login or sign-up screen and determines if
the user is granted access


  ///formats the student structure and writes it to file if 'StudentInfo.txt' is
empty
  if(infoFileEmpty() == 0)
  {
        formatStudent(&student);
        writeStudentFile(student);
  }


  ///runs menu if the login is successful
  if(loggedIn == 1)
  {
        menu(student);
  }
  else
  {
        printf("User could not be authenticated");
  }


  return 0;
}
```

Test Plan Results

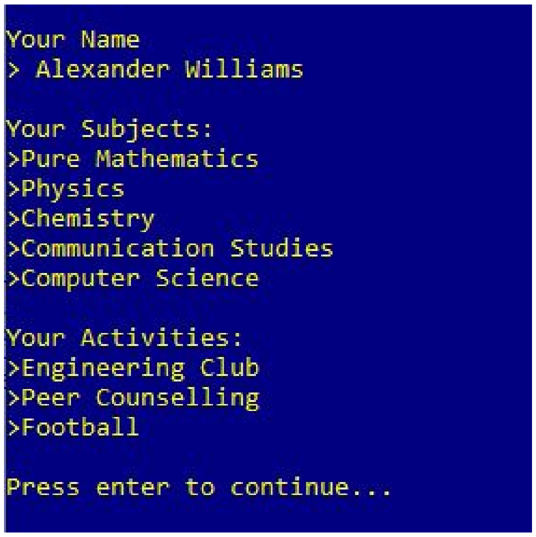
Input Screen/Functionality	Input	Result Image
1. Login Screen	Username: teacher Password: 5678	 Incorrect Password or Username...Try Again Username>
2. Login Screen	Username: student Password: 1234	 Username> student Password> .... Data written successfully! Press enter to continue...  [1] Edit Your Information [2] Display Your Information [3] Edit Your Schedule [4] Display Your Schedule [5] Add Assignments [6] Mark Assignments as Complete [7] Display Assignments [8] Update Login Info [9] Exit Select Option: >
3. Main Menu	Choice: w	 Please enter an option from the menu: >

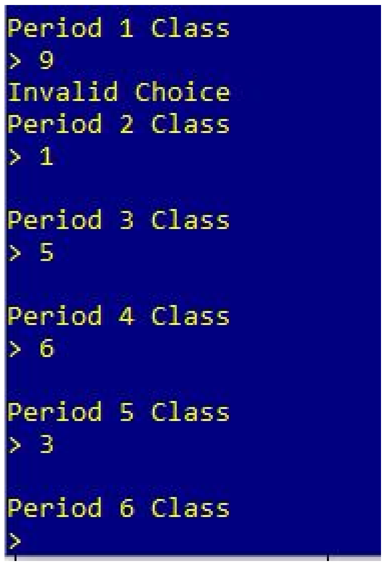
4. Student Info Screen	First Name: Alexander
	Last Name: Williams
	Number of Subjects: 7
		
		```
		Please enter 3 or 4
		>
		```

5. Student Info Screen	First Name: Alexander
	Last Name: Williams
	Number of Subjects: 4
	Select 4 Subjects:
	1
	2
	4
	5
	Number 1 easiest: 5
	Number 2 easiest: 2
	Number 3 easiest: 3
	Number 4 easiest: 1
	Number 5 easiest: 4
	Number of Co-Curricular activities: 6
		
		```
		Number of Co-Curricular Activities
		[1-5]> 6
		Please enter a number between 1 and 5
		>
		```

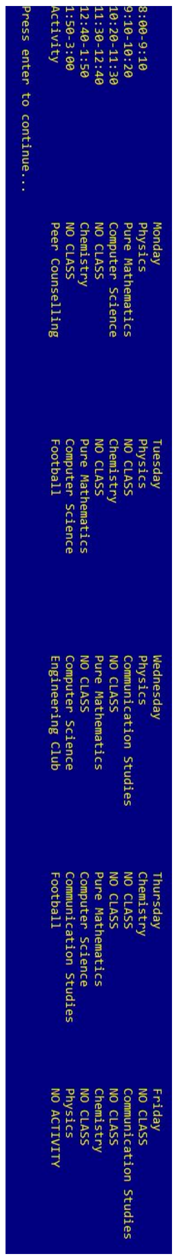
6. Student Info Screen	First Name: Alexander
	Last Name: Williams
	Number of Subjects: 4
	Select 4 Subjects:
	1
	2
	4
	5
		
		```
		Select 3 activities(y):
		> 16

		> 35

		> 53

		Data written successfully!
		Press enter to continue...
		```

	Number 1 easiest: 5 Number 2 easiest: 2 Number 3 easiest: 3 Number 4 easiest: 1 Number 5 easiest: 4 Number of Co-Curricular activities: 3 Select 3 Activities: 16 35 53	

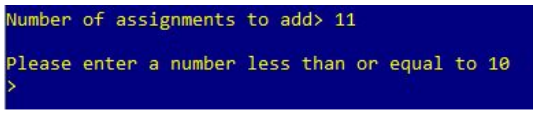
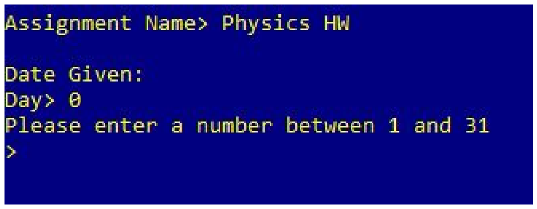
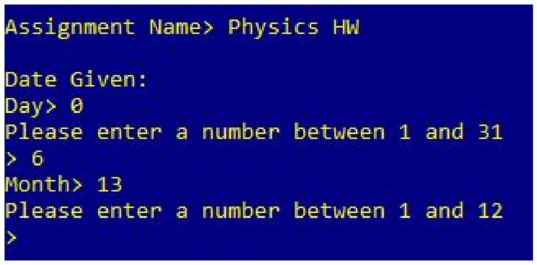
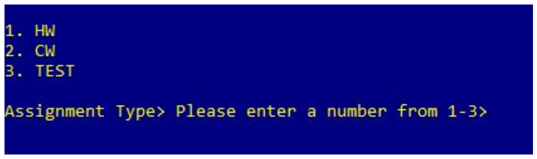
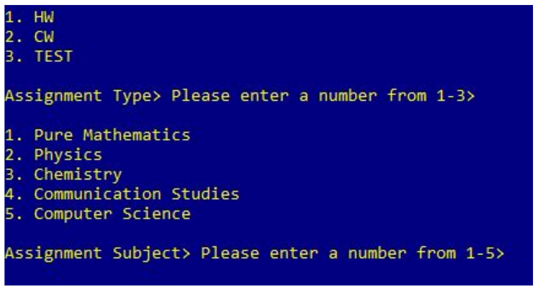
7. Main Menu (Display Information Screen)	Choice: 2	
		`Your Name`
		`> Alexander Williams`
		
		`Your Subjects:`
		`>Pure Mathematics`
		`>Physics`
		`>Chemistry`
		`>Communication Studies`
		`>Computer Science`
		
		`Your Activities:`
		`>Engineering Club`
		`>Peer Counselling`
		`>Football`
		
		`Press enter to continue...`

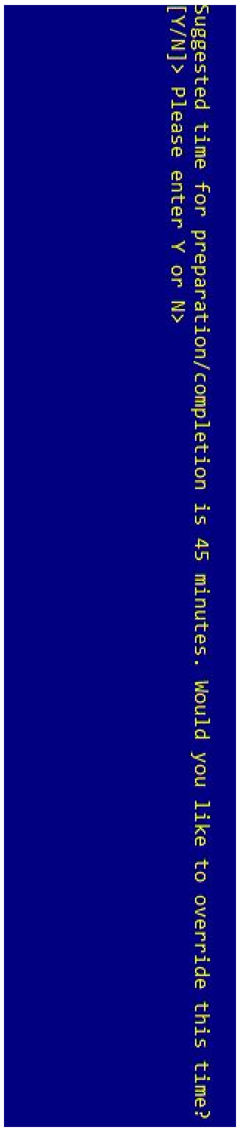
8. Edit Schedule Screen	Monday
	Period 1: 9
	Period 2: 1
	Period 3: 5
	Period 4: 6
	Period 5: 3
		
		`Period 1 Class`
		`> 9`
		`Invalid Choice`
		`Period 2 Class`
		`> 1`
		
		`Period 3 Class`
		`> 5`
		
		`Period 4 Class`
		`> 6`
		
		`Period 5 Class`
		`> 3`
		
		`Period 6 Class`
		`>`

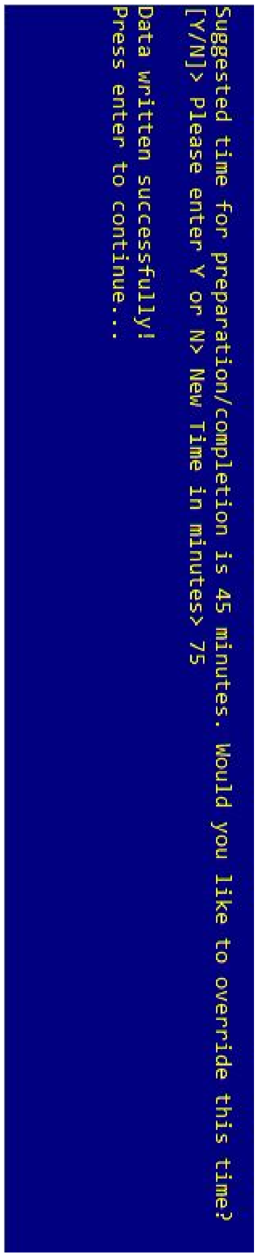
9. Edit Schedule Screen -> Display Schedule Screen	Monday Period 1: 2 Period 2: 1 Period 3: 5 Period 4: 6 Period 5: 3 Period 6: 6 Activity: 2 Tuesday Period 1: 2 Period 2: 6 Period 3: 3 Period 4: 6 Period 5: 1	

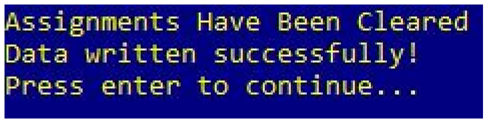
	Period 6: 5 Activity: 3 Wednesday Period 1: 2 Period 2: 4 Period 3: 6 Period 4: 1 Period 5: 6 Period 6: 5 Activity: 1 Thursday Period 1: 3 Period 2: 6 Period 3: 6 Period 4: 1 Period 5: 5 Period 6: 4 Activity: 3 Friday Period 1: 6 Period 2: 4 Period 3: 6 Period 4: 3 Period 5: 6 Period 6: 2 Activity: 4 ENTER	

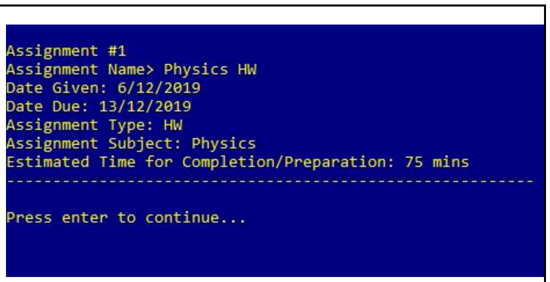
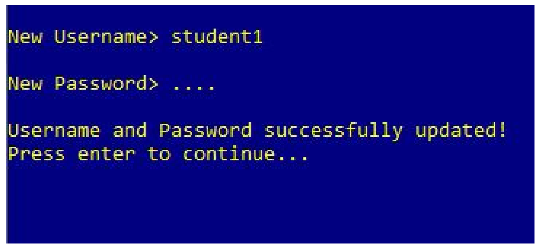
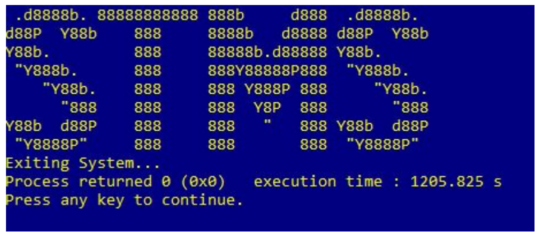
```
                              Monday                        Tuesday                       Wednesday                     Thursday                      Friday
8:00-9:10                     Physics                       Physics                       Physics                       Chemistry                     NO CLASS
9:10-10:20                    Pure Mathematics              NO CLASS                      Communication Studies         NO CLASS                      Communication Studies
10:20-11:30                   Computer Science              Chemistry                     NO CLASS                      NO CLASS                      NO CLASS
11:30-12:40                   NO CLASS                      NO CLASS                      Pure Mathematics              Pure Mathematics              Chemistry
12:40-1:50                    Chemistry                     Pure Mathematics              NO CLASS                      Computer Science              NO CLASS
1:50-3:00                     NO CLASS                      Computer Science              Computer Science              Communication Studies         Physics
Activity                      Peer Counselling              Football                      Engineering Club              Football                      NO ACTIVITY

Press enter to continue...
```

	Menu Choice 4	
10. Add Assignments Screen	Number of Assignments to Add: 11	 Number of assignments to add> 11 Please enter a number less than or equal to 10 >
11. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 0	 Assignment Name> Physics HW Date Given: Day> 0 Please enter a number between 1 and 31 >
12. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 6 Month: 13	 Assignment Name> Physics HW Date Given: Day> 0 Please enter a number between 1 and 31 > 6 Month> 13 Please enter a number between 1 and 12 >
13. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 6 Month: 12 Day: 13 Month: 12 Type: 4	 1. HW 2. CW 3. TEST Assignment Type> Please enter a number from 1-3>
14. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 6 Month: 12 Day: 13 Month: 12	 1. HW 2. CW 3. TEST Assignment Type> Please enter a number from 1-3> 1. Pure Mathematics 2. Physics 3. Chemistry 4. Communication Studies 5. Computer Science Assignment Subject> Please enter a number from 1-5>

	Type: 1 Subject: 7	
15. Add Assignments Screen	Number of Assignments to Add: 1 Name: Physics HW Day: 6 Month: 12 Day: 13 Month: 12 Type: 3 Subject: 2 Override?: m	 `Suggested time for preparation/completion is 45 minutes. Would you like to override this time?` `[Y/N]> Please enter Y or N>`

16. Add Assignments Screen	Number of Assignments to Add: 1
	Name: Physics HW
	Day: 6
	Month: 12
	Day: 13
	Month: 12
	Type: 3
	Subject: 2
	Override?: Y
	New Time: 75
		
		`Suggested time for preparation/completion is 45 minutes. Would you like to override this time?`
		`[Y/N]> Please enter Y or N> New Time in minutes> 75`
		`Data written successfully!`
		`Press enter to continue...`

17. Main Menu (Mark Assignments as Complete Screen)	Choice: 6	
		`Assignments Have Been Cleared`
		`Data written successfully!`
		`Press enter to continue...`

18. Main Menu (Display Assignments Screen)	Choice: 7	
19. Update Login	New Username: student1 New Password: 4321	
20. Main Menu (Farewell Screen)	Choice: 9	

```
Assignment #1
Assignment Name> Physics HW
Date Given: 6/12/2019
Date Due: 13/12/2019
Assignment Type: HW
Assignment Subject: Physics
Estimated Time for Completion/Preparation: 75 mins
----------------------------------------------------------

Press enter to continue...
```

```

New Username> student1

New Password> ....

Username and Password successfully updated!
Press enter to continue...
```

```
 .d8888b. 88888888888 888b     d888  .d8888b.
d88P  Y88b    888     8888b   d8888 d88P  Y88b
Y88b.         888     88888b.d88888 Y88b.
 "Y888b.      888     888Y88888P888  "Y888b.
    "Y88b.    888     888 Y888P 888     "Y88b.
      "888    888     888  Y8P  888       "888
Y88b  d88P    888     888   "   888 Y88b  d88P
 "Y8888P"     888     888       888  "Y8888P"
Exiting System...
Process returned 0 (0x0)   execution time : 1205.825 s
Press any key to continue.
```

Documentation

Assumptions

It was assumed that

- the user was a Campion College 6B student.
- the user was only capable of doing a maximum of 5 activities and 5 subjects.

Conclusion

By the grace of god and will of man most (if not all) objectives for the new system were satisfactorily met and to a larger extent, expectations for the project as well. Apart from the fact that the user interface is basic and that the assignments cannot be individually marked as complete, there are a couple of improvements that could be made to the system; these are:

- implementing a networking capability to back relevant data up to different devices and maybe even the internet.
- enabling the ability to have more than one students use the same system (multiple users).

Appendix

References

Cover Photo:

https://assicurato.re/tacito-rinnovo-caos-mediatico-generato-dal-ddl-concorrenza/brain-2062057/

Code:

https://stackoverflow.com/

http://patorjk.com/software/taag/#p=display&f=Graffiti&t=Type%20Something%20

https://www.youtube.com/watch?v=axJREnDphIc

https://www.codingunit.com/c-tutorial-binary-file-io

https://www.geeksforgeeks.org/readwrite-structure-file-c/

https://cc-jam.moodle.renweb.com/course/view.php?id=223

https://stackoverflow.com/questions/30133210/check-if-file-is-empty-or-not-in-c

https://www.codewithc.com/mini-project-in-c-library-management-system/

https://drive.google.com/file/d/1HhyXrDTQgx208ryNALyJklIIhsMPd35e/view?usp=sharing

Notes from Computer Science Class 2 at Campion College

Peers from Computer Science Course at Campion College