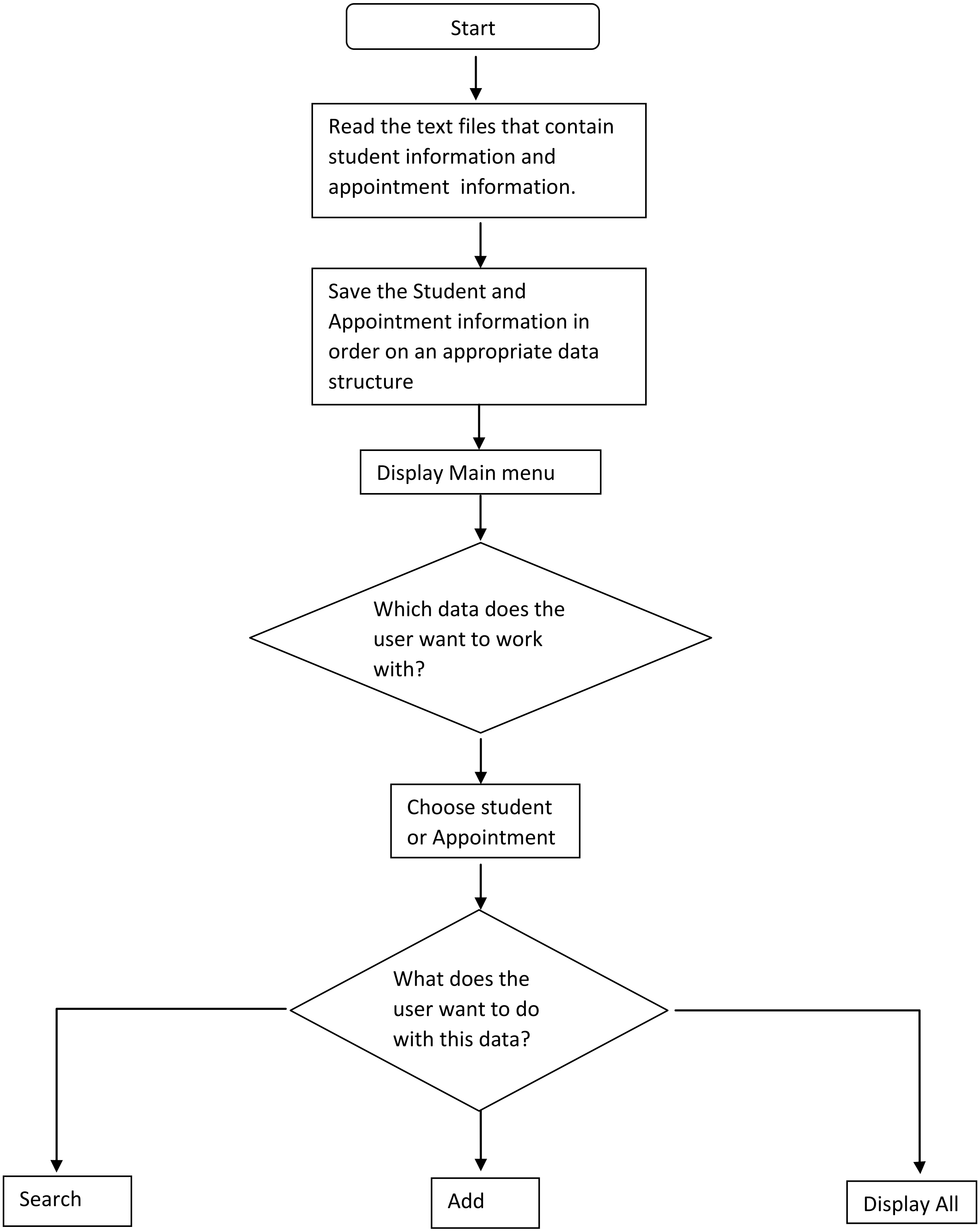
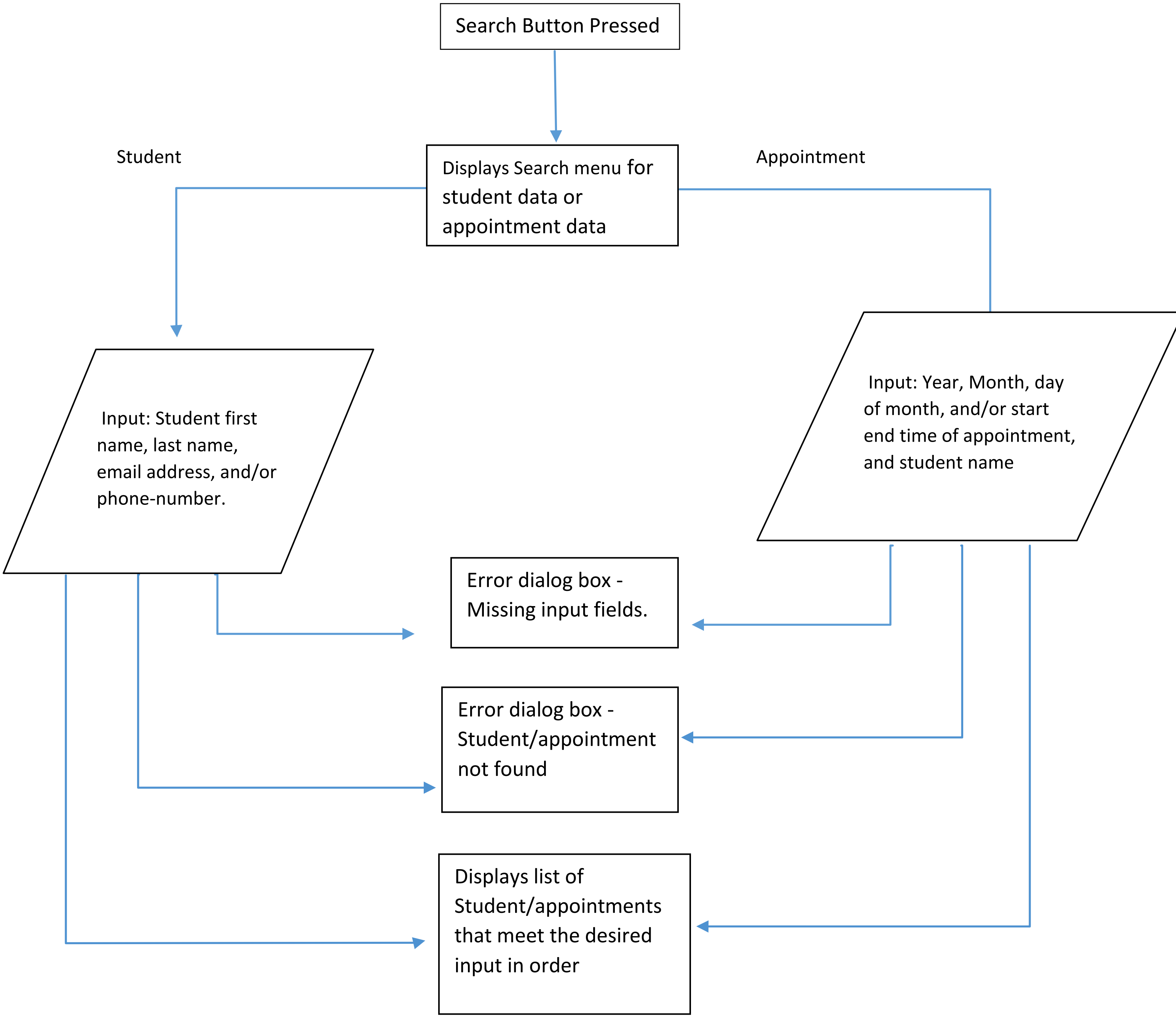
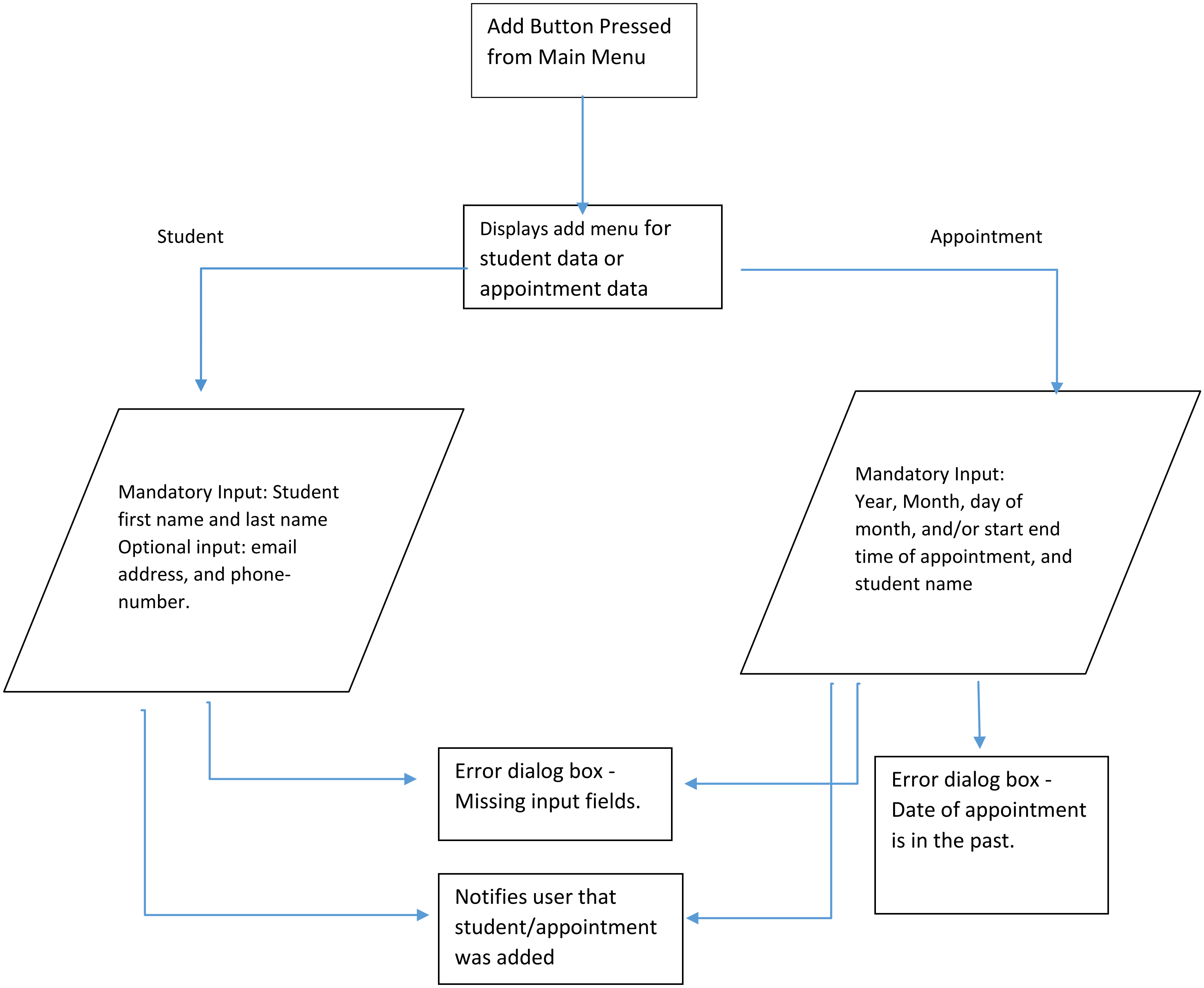

Start
Read the text files that contain student information and appointment information.
Save the Student and Appointment information in order on an appropriate data structure
Display Main menu
Which data does the user want to work with?
Choose student or Appointment
What does the user want to do with this data?
Search
Add
Display All

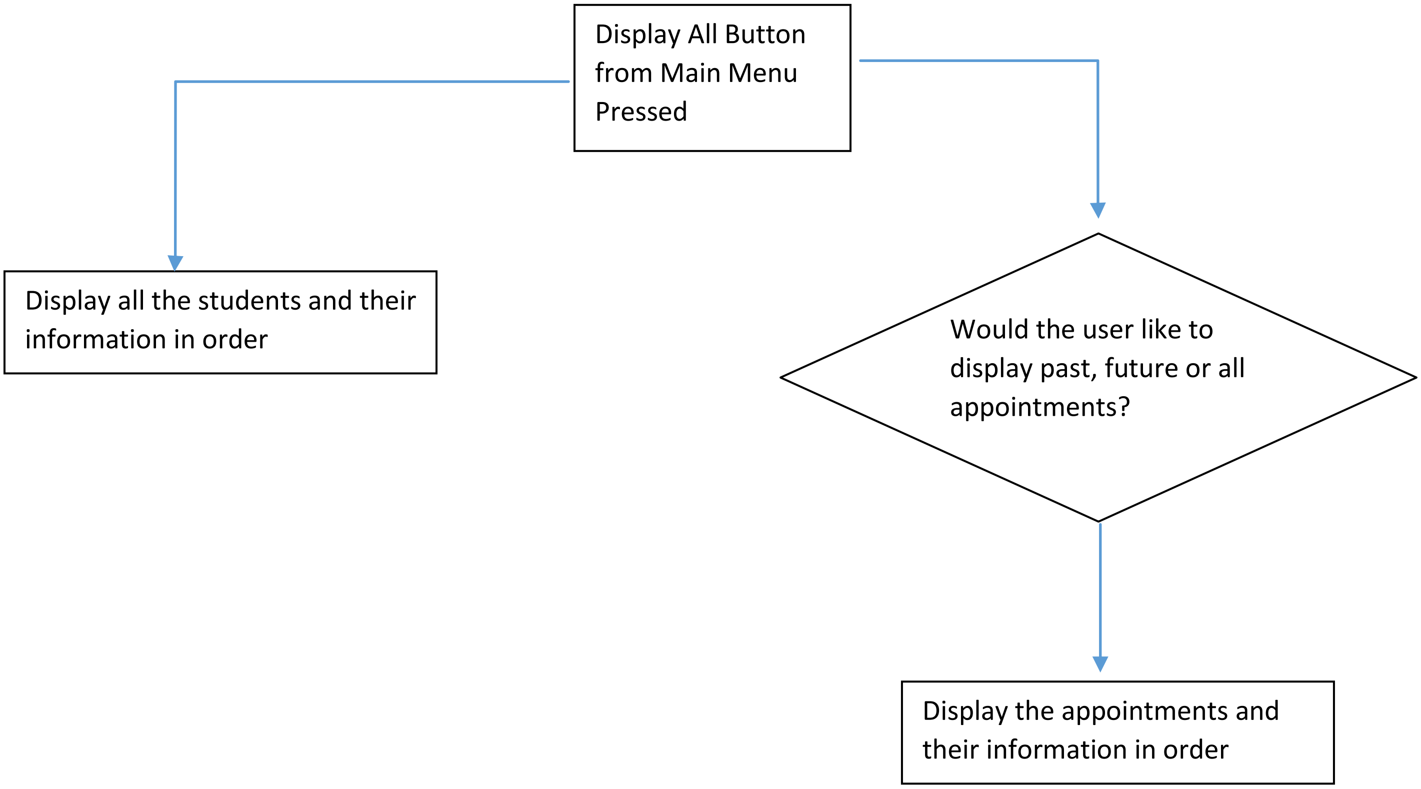
Search Logic for:


Search Button Pressed
Student
Appointment
Displays Search menu for student data or appointment data
Input: Student first name, last name, email address, and/or phone-number.
Input: Year, Month, day of month, and/or start end time of appointment, and student name
Error dialog box - Missing input fields.
Error dialog box - Student/appointment not found
Displays list of Student/appointments that meet the desired input in order

Add Logic

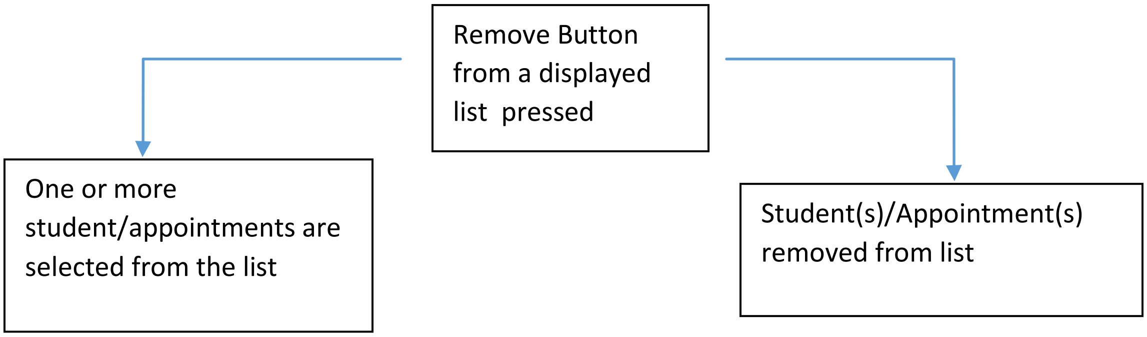

Add Button Pressed from Main Menu
Displays add menu for student data or appointment data
Student
Appointment
Mandatory Input: Student first name and last name Optional input: email address, and phone-number.
Mandatory Input: Year, Month, day of month, and/or start end time of appointment, and student name
Error dialog box - Missing input fields.
Error dialog box - Date of appointment is in the past.
Notifies user that student/appointment was added

Display All Data Logic


Display All Button from Main Menu Pressed
Display all the students and their information in order
Would the user like to display past, future or all appointments?
Display the appointments and their information in order

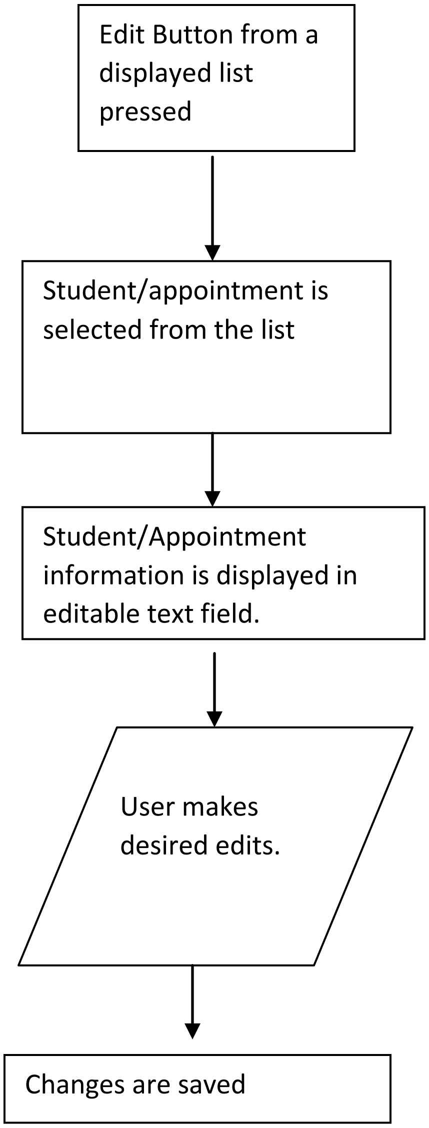
Remove Logic

*Remove option will be available on any window that displays a list. For example, on the window that pops up when user decides to display all or when the user makes a valid search.

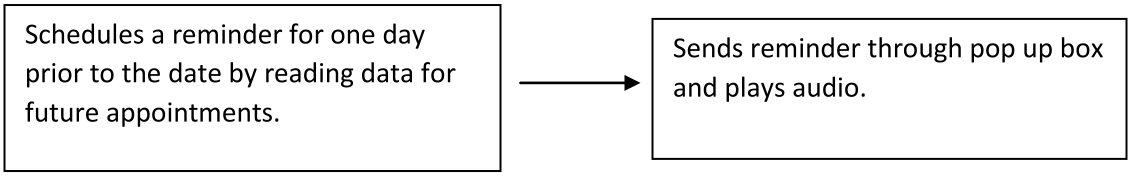

Remove Button from a displayed list pressed
One or more student/appointments are selected from the list
Student(s)/Appointment(s) removed from list

Edit Logic

*Edit option will be available on any window that displays a list. For example, on the window that pops up when user decides to display all or when the user makes a valid search.


Edit Button from a displayed list pressed
Student/appointment is selected from the list
Student/Appointment information is displayed in editable text field.
User makes desired edits.
Changes are saved

Background Processes: Setting reminders for future appointments.


Schedules a reminder for one day prior to the date by reading data for future appointments.
Sends reminder through pop up box and plays audio.

Tables Showing Fields, Data Types, and Methods for Each Class

*Classes extending JFrame do not include the variables for buttons, labels, comboboxes and such

Student Class

Constructor Name	**Parameter and Type**	**Description**	
Constructor	Student first and last name, phone number, email - Strings	Necessary information to create a student object. Saves information in an array size of 4	
Variables	**Public/private**	**Data Type**	**Description**
myStudentInfo	private	Array size 4	Holds the student information in the order: first name, last name , phone number, email address. Phone number and email address will be "---"if student does not have one
Method Name	**Parameter and Data Types**	**Object Returned and Data Type**	**Description**
Public getFirstName()	N/A	String	Returns the first name of a student
Public getlastName()	N/A	String	Returns the last name of a student
Public getEmailAdress()	N/A	String; Null pointer exception if there is no email address	Returns email address if the student has one
Public getPhoneNumber	N/A	String; Null pointer exception if there is no phone number	Returns phone number if the student has one
Public getStudentInfo()	N/A	Array: myStudentInfo[]	Returns all the students information in the form of an array
Public setStudentFirstname()	String	void	Sets or changes the first name of a student
Public setStudentLastname()	String	void	Sets or changes the last name of a student
Public setEmailAddress()	String	void	Sets or changes the email address of a student
Public setPhoneNumber()	String	void	Sets or changes the phone number of a student
Public toString	String	void	Returns student information in the form of a string in the format: first name, last name, email address, and

			phone number
Public compareTo	Student	int	Compares two students. Compares the in the order of first name, last name, phone number and emailaddress
Public static quicksort	ArrayList<Student>, int	void	Sorts an arraylist of students alphabetically
Private static Swap	ArrayList<Student> info, int x, int y	Void	Swaps the content of the two specified indexes in the given arraylist

ControlSystem class

Constructor Name	**Parameter and Type**	**Description**	
Constructor	N/A	Initializes all privates variables: the arraylist of students, appointments and years, and the scanner required to read the file. Throws a FileNotFoundException if the file does not exist	
Variables	**Public/private**	**Data Type**	**Description**
myFutureAppointments	Private	ArrayList<Appointment>	Holds all the upcoming appointments in order from nearest to farthest
myPastAppointment	Private	ArrayList<Appointment>	Holds all the past appointments in order from oldest to recent
myAllAppointments	Private	ArrayList<Appointment>	Holds all the existing appointments in order of oldest to nearest
myStudents	Private	ArrayList<Student>	Holds all the Student objects in alphabetical order
Method names and type	**Parameter and Type**	**Return Type**	**Description**
Public static scheduleReminders	ArrayList<Appointment>	Void	Sets a reminder 1 day earlier for all the appointments in the arraylist
public static void scheduleReminders	Appointment appt	Viod	Sets a reminder 1day earlier for the appointment
public static void	Appointment apt	Date	Creates a Date object

scheduleDateReminder			that represents a day before the appointment. Date object is returned
Public open	N/A	void	Begins storing all student and appointment information
Private readStudentInfo()	N/A	Void	Begins reading student.txt and stores all the student information as student objects in myStudents
Private readAppointmetnInfo()	N/A	Void	Reads the appointment.txt file and stores all the appointment information as appointment objects in myApppointments
Public addStudent	String firstName, String lastName, String phoneNumber, String emailAddress		
Public getAllStudents	N/A	ArrayList<Student>	Returns an arraylist of students
Public getAllAppointments	N/A	ArrayList<Appointment>	Returns arraylist myAllAppointments
Public getPastAppointments()	N/A	ArrayList<Appointment>	Returns arraylist myPastAppointments
Public getFutureAppointments	N/A	ArrayList<Appointment>	Returns arraylist myFutureAppointments
Public removeStudent	Student student	boolean	Removes the specified student from the myStudent arraylist. Reurns false if the student is not foud
Public close	N/A	Void	Begins to save the data in the arraylists to the text files
private saveStudentData	FileWriter rt	Void	Reads the student information from the myStudents arraylist and saves to student.txt
private saveAppointmentData	FileWriter rt	Void	Reads the appointment information from the myAppointment arraylist and saves to

			appointment.txt

Appointment Class

Constructor Name	**Parameter and Type**	**Description**	
Construtctor	String name, int year, int month, int date, int startTime, int endTime	Saves the appointment information to an arraysize of 5 in the order of Name, Month, Date, start time, end time. Creates a date object to represent the start time of the Appointment	
Variables	**Public/Private**	**Data type**	**Description**
myApptInfo	Private	Int []	Holds the year, month, day of month, start time and end time of appointment. Size of five
numApptParam	Private	Int	The number of parameters(5) required to create an appointment object.
myStartTimeDate	Private	Date	Date object that represents the start time of the appointment
Method	**Parameter and Type**	**Return type**	
Public getName	N/A	String	Returns name of Student
Public getApptInfo	N/A	Int []	Returns the date and time of the appointment in an array
Public getYear	N/A	String	Returns the year of appointment
Public getMonth	N/A	String	Returns month of appointment
Public getDayOfMonth	N/A	String	Returns day of the month of appointment
Public getStartTime	N/A	String	Returns start time of appointment
Public getEndTime	N/A	String	Returns end time of appointment
Public getDate	N/A	Date	Returns the date object of appointment
Public static convertToMilit	Int hours, int minutes, String time of day	String	Converts time to military time using

			hours, minutes and time of day
Public setYear	Int year	Void	Changes the year of appointment
Public setMonth	Int month	Void	Changes month of appointment
Public setDate	Int date	Void	Changes date of the appointment
Public setStartTimeHrs	Int startTimeHrs	Void	Changes the hour at which appointment starts
Public setStartTime	Int startTime	Void	Changes the whole
Public setEndTimeHrs	Int endTimeHrs	void	Changes the hour at which appointment ends
Public setEndTime	Int endTime	Void	
Public setStudent	String studentName	Void	Changes the students name
Public compareTo	Appointment	int	Compares two Appointment. Compares the in the order of Year, Month, day of month, start time, end time, name
Public static quicksort	ArrayList<Appointment>, int	void	Sorts an arraylist of appointments chronologically - newest to oldest
Private static Swap	ArrayList<Appointment> info, int x, int y	Void	Swaps the content of the two specified indexes in the given arraylist

List.java extends JFrame: This will be a generic class to display both students and appointments.

Constructor Name	**Parameter**	**Description**	
List	String listType, ArrayList<Student> students, ArrayList<Student> allStudents	Displays a list of all the students in the arraylist student.	
List	ArrayList<Appointment> appointments, ArrayList<Appointment>all appointments	Displays a list of all the appointments in the arraylist appointment	
Variables	**Public/Private**	**Data type**	**Description**
myStudents	Private	ArrayList<Student>	Holds the student objects that will be

			displayed
myAllStudents	Private	ArrayList<Student>	Reference to the arraylist in ControlSystem.java: myStudents
myAppointments	Private	ArrayList<Appointment>	Holds the appointments that will be displayed
myAllAppointments	Private	ArrayList<Appointment>	Reference to the arraylist in ControlSystem.java: myAllAppointments
myListType	private	String	Used to determine whether the list that needs to be displayed will contain student or appointment objects
Method	**Parameter and Type**	**Return type**	**Description**
Public initializeList	N/A	Void	Determines whether to initialize a list of students or appts using the string myListType
Private addApptsToList	N/A	Void	Displays a list of the appointments in myAppointments
Private addStudentsToList	N/A	Void	Displays a list of the appointments in myStudents
Private Remove	Java action/event **Remove button press*	void	Executed when remove button is pressed. Removes student everywhere in program
removeAppts	int firstRowSelected, int lastRowSelected	Void	Removes appointments between the first and last row seleted
removeStudents	int firstRowSelected, int lastRowSelected	Void	Removes appointments between the first and last row seleted
Private editData	Java action/ event **Edit button*	Void	Determines whether appointment or student is selected. Displays a form that allows the user to make necessary changes

MainActivity Class extends JFrame

Constructor name	**Parameter and Type**	**Description**	
MainActivityClass	N/A	Initializes all GUI components so the Jframe displays; Initializes a ControlSystem ;Adds a window listener so the program knows when the application is about to close and can begin to save the data.	
Variables Name	**Public/Private**	**Data type**	**Description**
myControlSystem	Private	ControlSystem	Gives access to all the data that was read from txt files
myStudentLabal	Private final	Static Final String	Represents the string "Student"
myAppointmentLabel	Private	Static Final string	Represents the string "Appointment"
Method	**Parameter and Type**	**Return type**	**Description**
scheduleReminders	N/A	Void	Uses the arraylist of appointments that can be accessed using the myControlSystem object to set reminders for the client
add	Java action/event _*Add button is pressed_	Void	Determines if the client is modifying student or appointment data. Opens the appropriate Jframe that allows a student or appointment object to be added
Search	Java action/ event *_Search button is pressed_	Void	Determines if the client is modifying student or appointment data. Opens the appropriate Jframe that allows a student or appointment object to be searched.
displayAll	Java action/ event *_Display All button is pressed_	Void	Determines if the client is viewing student or appointment data. Initializes a list to display the appointments or

			students.

ModifyAppointment.java extends JFrame

*An appointment form displays all the information needed to be filled to make an appointment or edit one

Constructor name	**Parameter and Type**	**Description**	
ModifyAppointment	ArrayList<Appointment> appointments, String option	Initializes all private variables. Displays an appointment form to be filled out	Used when the client wants to search or add an appointment
ModifyAppointment	ArrayList<Appointment> appointments, Appointment apptToEdit, String option	Initializes all private variables. Displays an appointment form to be filled out	Used when the client wants to edit an appointment
Variables Name	**Public/Private**	**Data type**	**Description**
myAppointments	Private	ArrayList<Appointment>	Reference to the arraylist in ControlSystem.java: myAllAppointments
myNumParams	Private	Int	Number of parameters required to fill out in the appointment form
myParameters	Private	ArrayList<String>	Holds the input after user completes appointment form
myOption	Private	String	Represents whether the user wants to add or edit appointment
myApptToEdit	Private	Appointment	Points to the appointment object that the user wants to edit
Method	**Parameter and Type**	**Return type**	**Description**
modifyAppt	Java action/event **Search/Add button is pressed*	Void	Determines if the user is adding, searching or editing and executes the appropriate task.
addParams	N/A	Void	Reads the inputs on the appointment form and saves to myParameters
Edit	N/A	Void	Allows the user to edit an appointment by displaying an empty appointment form. Requests the user to reenter all the information with the changes

isValidAppt	N/A	Boolean	Returns if the appointment is possible to be scheduled by checking if all parameters are entered and date is not in past
isValidParam	N/A	Boolean	Returns true if the user has filled in all the parameters
isValidDate	N/A	Boolean	Returns true if the date of the appointment is not in the past
checkOverLap	Appointment appt	Boolean	Returns true if the date of the appointment does not conflict with other appointments
createDate	N/A	Date	Creates and returns a date object based on the user input on the appointment form

SearchAppt.java extends JFrame- Allows the user to narrow appointments with the following parameters: year, month and date

Constructor name	**Parameter and Type**	**Description**	
SearchAppt	ArrayList<Appointment>	Initializes all privates. Displays a search form that requests the user to enter at least one parameter in order.	Ex. Year only can be searched; however Month cannot be entered without year.
Variables Name	**Public/Private**	**Data type**	**Description**
myAppointments	Private	ArrayList<Appointment>	Reference to the arraylist in ControlSystem.java: myAllAppointments
myParams	Private	String	Holds all the user input as Strings
Method	**Parameter and Type**	**Return type**	**Description**
Search	Java action/event _*Search button is pressed_	Void	Displays a list of appointments that matched users input
findValidAppts	N/A	ArrayList<Appointment>	Returns an arraylist of students that matched the user input stored in myParams.
isValidParms	N/A	Boolean	Returns true if at least the first parameter(year) is entered

addParams	N/A	Boolean	Returns true if the user input was added successfully to myParams;

ModifyStudent.java extends JFrame

Constructor name	**Parameter and Type**	**Description**	
ModifyStudent	ArrayList<Student>, String option	Initializes all privates. Displays a Student form to be filled out	Used if the user wants search or add a student. Any private that will not be used will be null
ModifyStudent	ArrayList<Student>, Student studentString option	Initializes all privates. Displays a Student form to be filled out	Used if the user wants to edit a student's information
Variables Name	**Public/Private**	**Data type**	**Description**
myStudents	Private	Arraylist<Student>	Reference to the arraylist in ControlSystem.java: myStudents
myParams	Private	Arraylist<String>	Holds the input after user completes Student form
myOption	Private final	String	Represents whether the user wants to add, search or edit appointment
myStudentToEdit	Private	Student	Reference to the student that user wants to edit
Method	**Parameter and Type**	**Return type**	**Description**
modifyStudent	Java action/event **Search/Add/Edit button is pressed*	Void	Reads the users input. Determines whether the user wants to add, search or edit and executes that task
addParams	Boolean	Void	Adds the user input to myParams and returns true if at least one of the parameters was entered
Edit	N/A	Void	Edits the students information and reorganizes the myStudent arraylist
Search	N/A	Void	Displays the student/s and their information in a list if they matched the user's

			inputs. Displays error if no student was found.
findValidStudents	N/A	ArrayList<Student>	Returns a list of students that match the user's inputs
Add	N/A	Void	Creates a student object based on user input and adds it to myStudents. Reorganizes myStudents

<u>Test Plan</u>

Success Criteria Tested	**Goal of Test**	**Strategy for Testing**
A	Main menu is able to open.	Run the Program
B	All the Student information is displayed correctly in a list and in alphabetical order	Select "Students" and press the Display All Button to see a list of students. Compare the information presented to the one student.txt file where the information was obtained from
C	All the Appointment information is displayed correctly in a list and in chronological order	Select "Appointment" and press the Display All button to see a list of students. Compare the information presented in the list to the one in appointment.txt file where the information was obtained from and confirm that it is correct
D	New students can be added as long as they have at least a first name.	Select "Student" and press the Add button. 1) Add a student with first name, last name, email address, and phone number 2) Add a Student with entering only 3, 2 and 1 of the parameters
E	New appointments can be added as long as: all the required information is entered, its date is not in the past and does not conflict with another appointment	Select "Appointment" and press the Search button. 1) Attempt to add an appointment with missing fields such as: the year, month, date, start time, end time and name. An error should pop up notifying user of the missing fields. 2) Attempt to add an appointment that occurs in the past. An error should pop up notifying user that date of appointment is in the past. 3) Schedule one valid appointment successfully.

		4) Schedule another appointment that conflicts with the appointment in 3. An error should pop up notifying user of the conflict.
F	Students can be searched based on first name, last name, phone number and/or email address	Search for students first by: 1) Only by first name 2) Only by last name 3) Only by email address 4) Only by phone number Search for students using various combinations of two fields, three fields and four fields. A list should display if multiple students meet criteria. An error if no students meet criteria
G	Appointments can be searched by narrowing down the year, month and date	1) Enter year 2) Enter Year and month 3) Enter Year, Month and Date 4) Attempt to enter Date and/or Month without year 5) Attempt to enter Year and Date without month
H	Students/Appointments can be removed one at a time or multiple at once	1) Select Display All Appointments/Students 2) Select one appointment and press remove 3) Select 2,3,4, appointments and press remove 4) Select all appointments and press remove
I	Appointment/Student information can be edited	1) Select Display all Student/Appointment 2) Select a student and press Edit 3) Change student information 4) Close the list and reopen it to see if the changes were updated
J	A reminder is sent via a popup/alarm within day before an appointment	1) Schedule an appointment for tomorrow. A reminder should pop up automatically. 2) Schedule an appointment for today. A reminder should popup because its within a day 3) Schedule an appointment for two days later. No reminder should popup