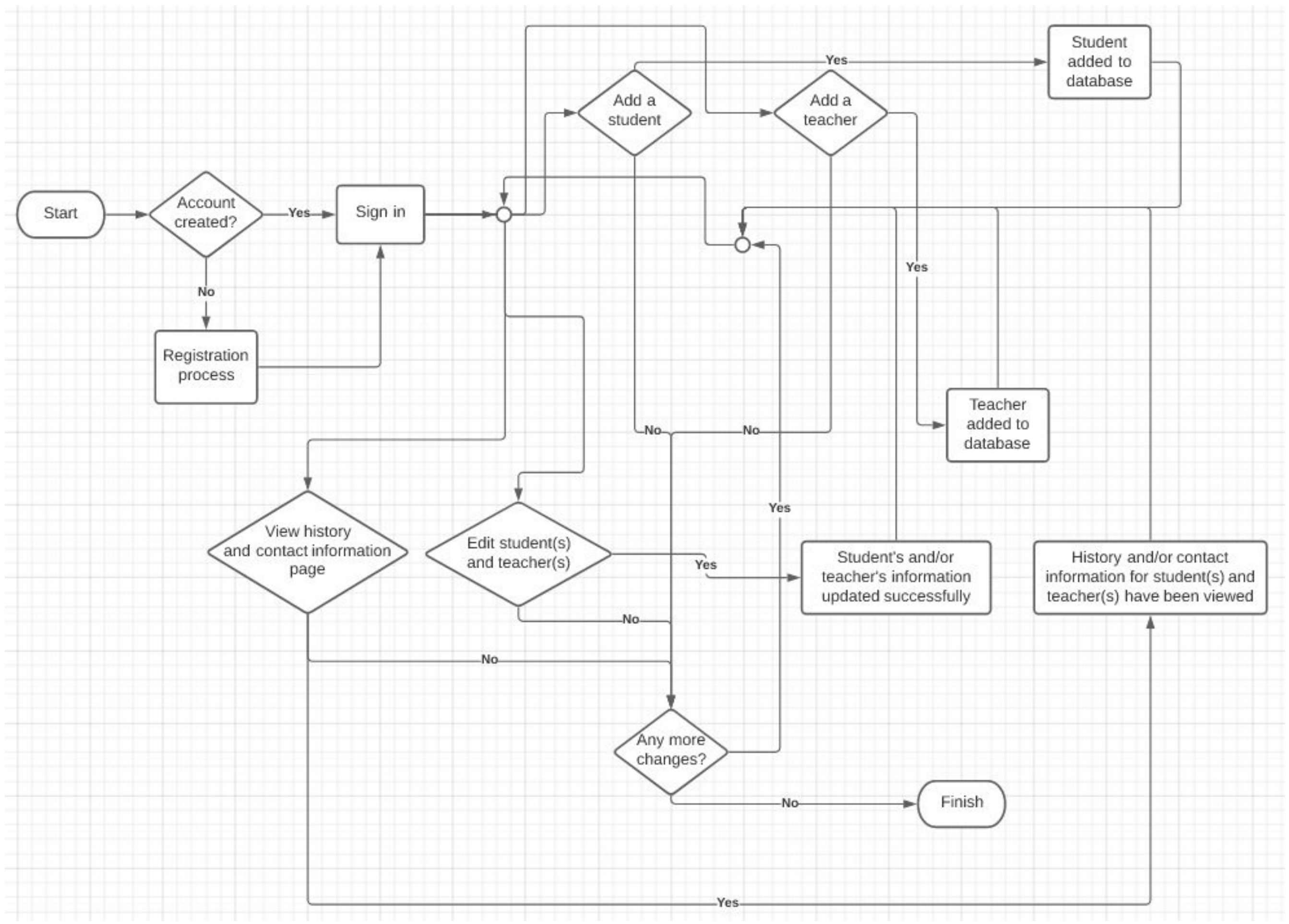
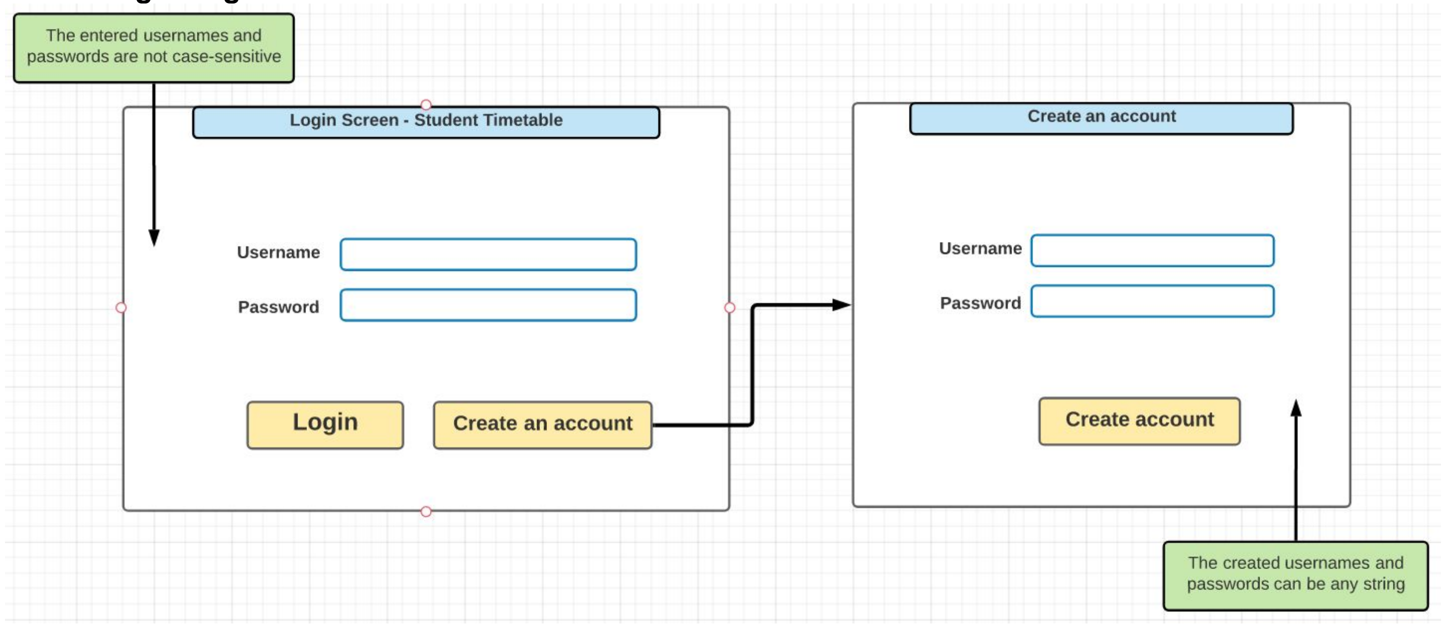
Criterion B: Design

Program plan and description



Start
Account created?
Yes
No
Sign in
Registration process
Add a student
Add a teacher
Yes
Student added to database
Yes
Teacher added to database
No
No
View history and contact information page
Edit student(s) and teacher(s)
Yes
Student's and/or teacher's information updated successfully
History and/or contact information for student(s) and teacher(s) have been viewed
No
No
Yes
Any more changes?
No
Finish
Yes

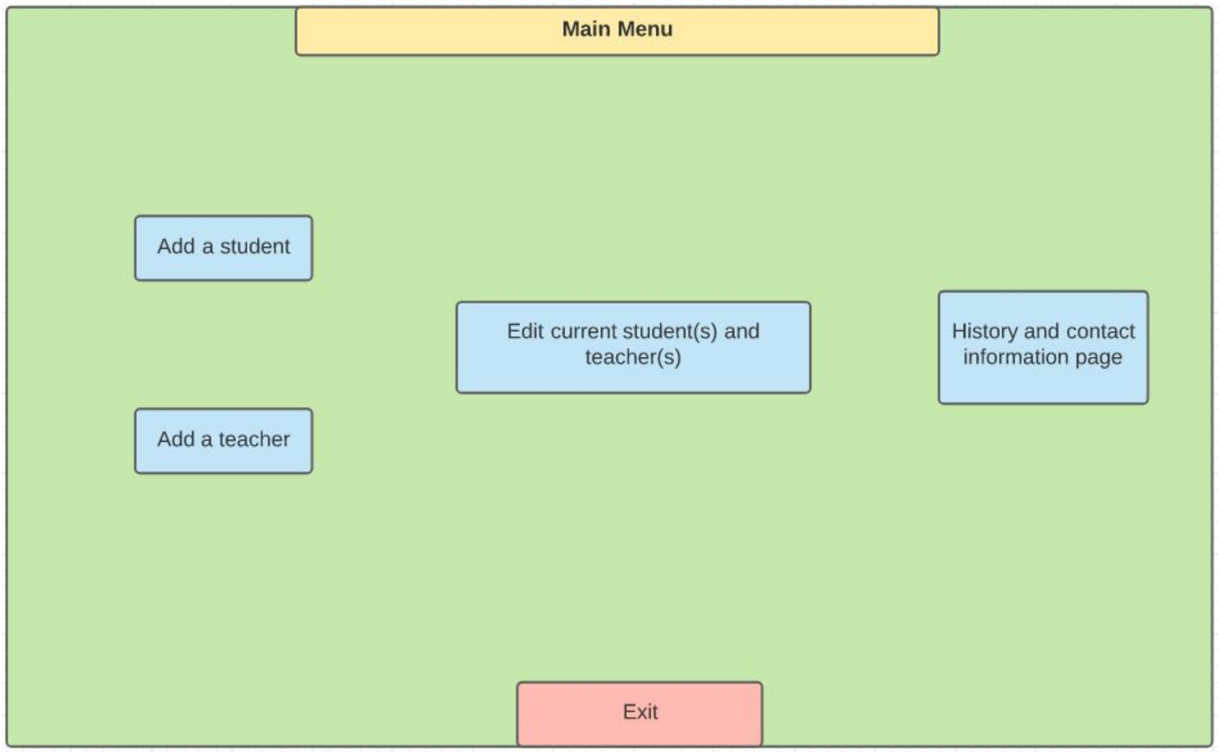
Login Page



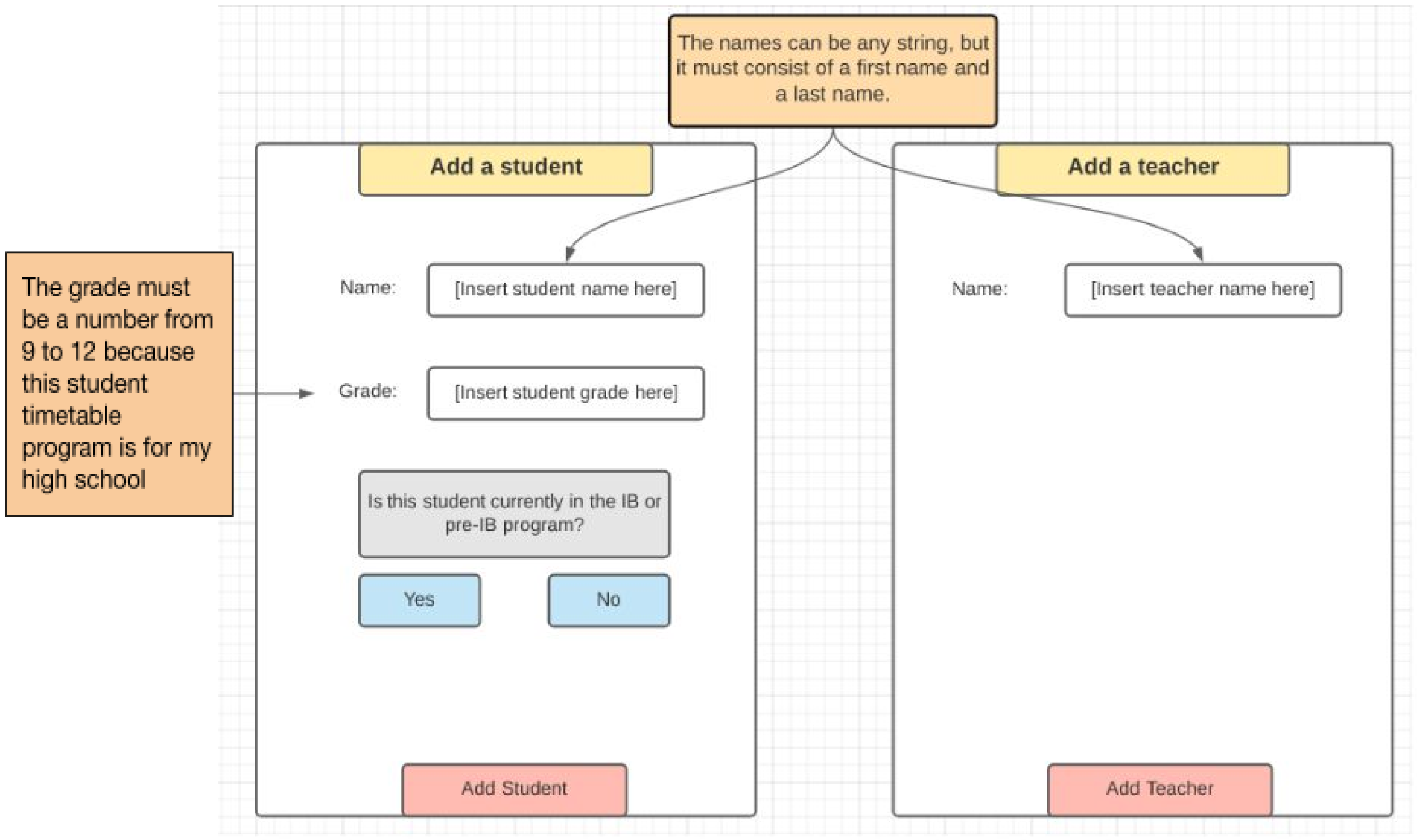
The entered usernames and passwords are not case-sensitive
Login Screen - Student Timetable
Username
Password
Login
Create an account
Create an account
Username
Password
Create account
The created usernames and passwords can be any string

Once logged in, the user is presented with the main menu page.

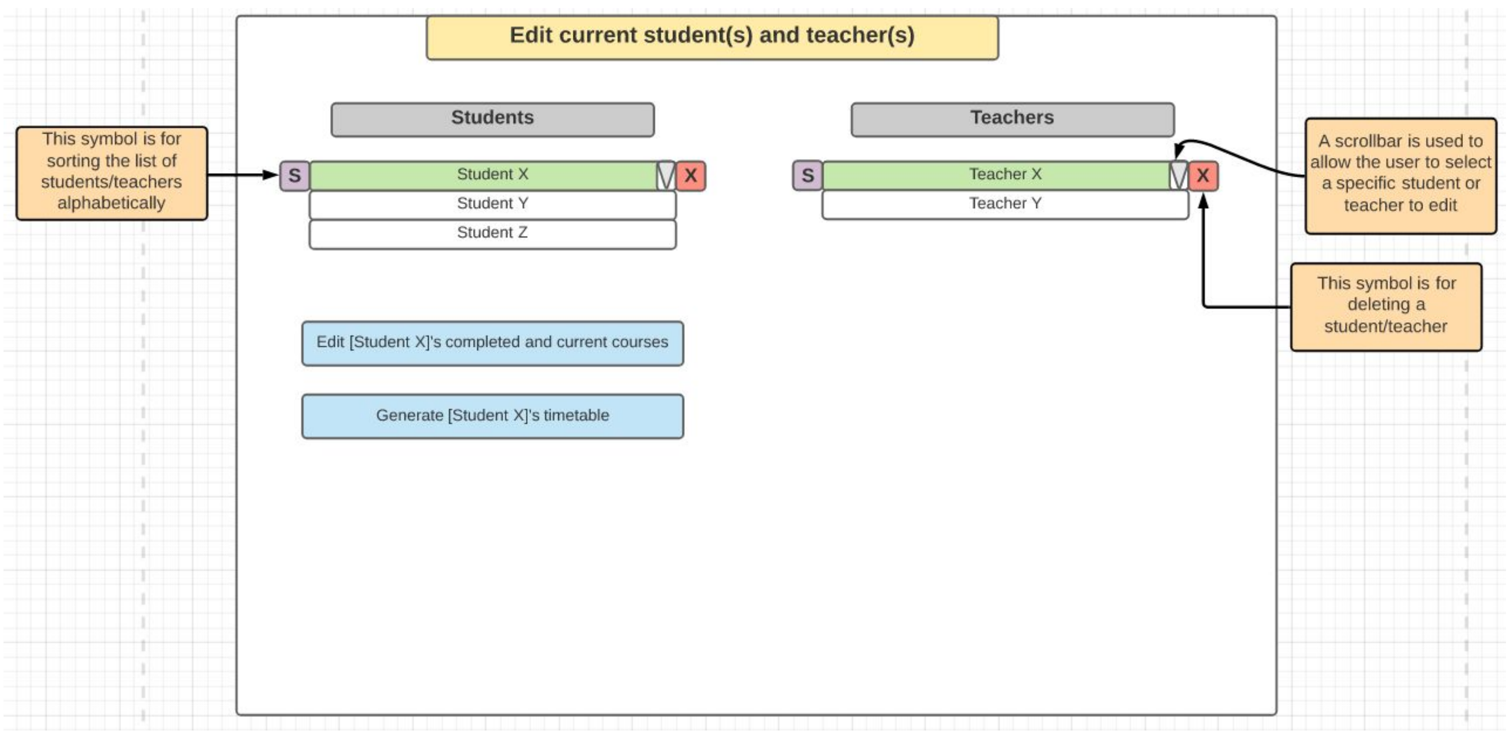
Main Menu Page


Main Menu
Add a student
Edit current student(s) and teacher(s)
History and contact information page
Add a teacher
Exit

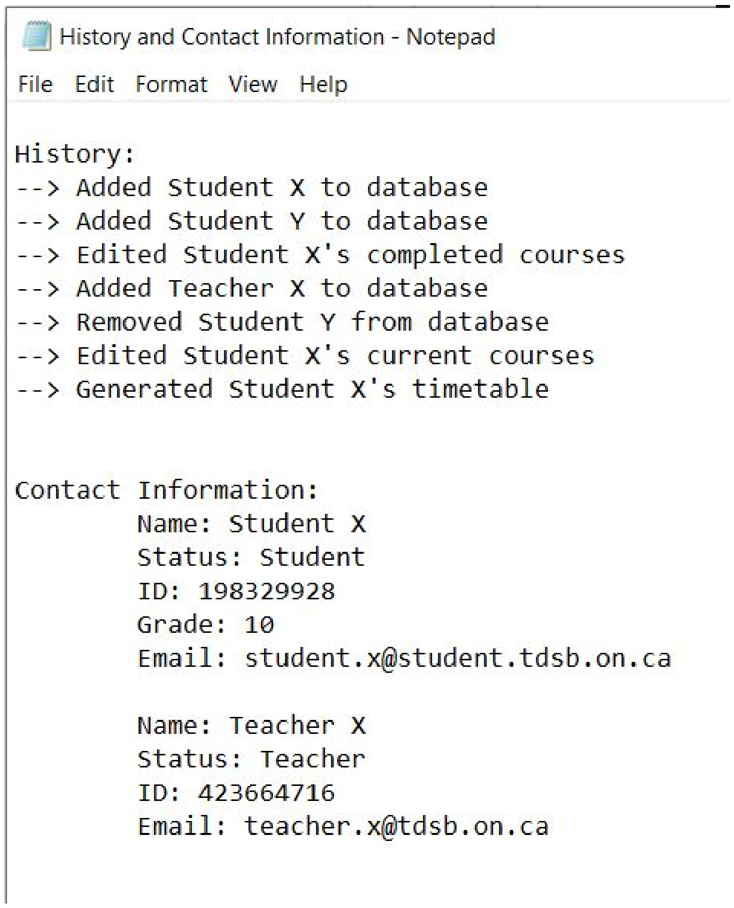
The screen for adding a student or teacher


The names can be any string, but it must consist of a first name and a last name.
Add a student
Add a teacher
The grade must be a number from 9 to 12 because this student timetable program is for my high school
Name:
[Insert student name here]
Grade:
[Insert student grade here]
Is this student currently in the IB or pre-IB program?
Yes
No
Add Student
Name:
[Insert teacher name here]
Add Teacher

Editing current student(s) and teacher(s) page

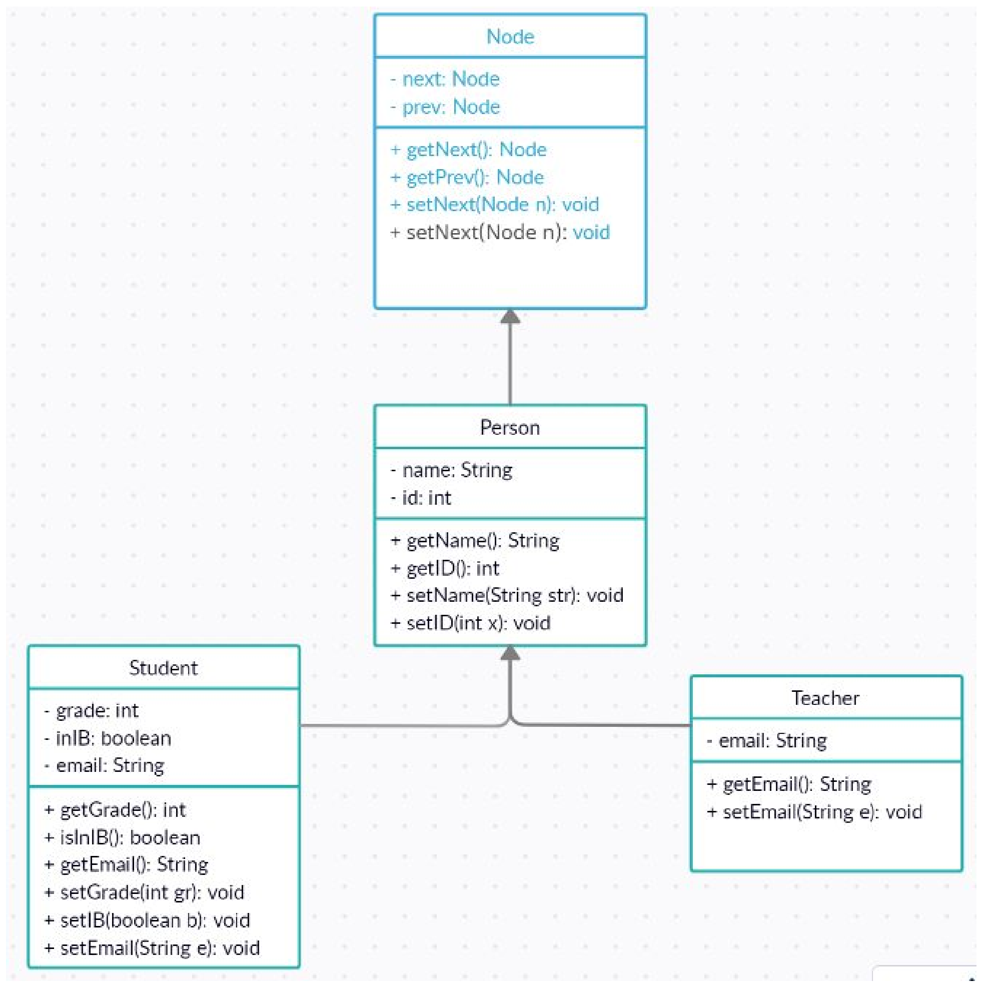

Edit current student(s) and teacher(s)
Students
Teachers
This symbol is for sorting the list of students/teachers alphabetically
S
Student X
Student Y
Student Z
X
S
Teacher X
Teacher Y
X
A scrollbar is used to allow the user to select a specific student or teacher to edit
This symbol is for deleting a student/teacher
Edit [Student X]'s completed and current courses
Generate [Student X]'s timetable

Storing the history and contact information of students and teachers


History and Contact Information - Notepad
File Edit Format View Help

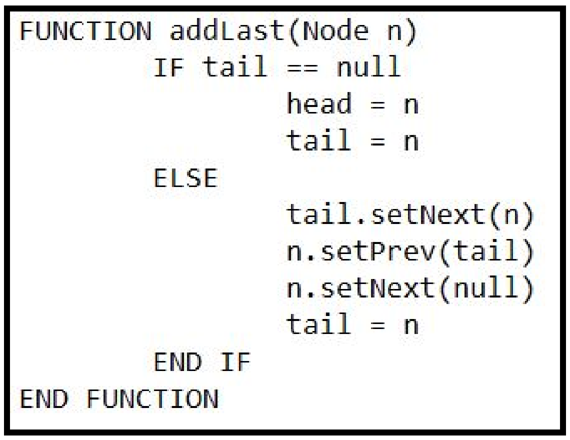
```
History:
--> Added Student X to database
--> Added Student Y to database
--> Edited Student X's completed courses
--> Added Teacher X to database
--> Removed Student Y from database
--> Edited Student X's current courses
--> Generated Student X's timetable


Contact Information:
        Name: Student X
        Status: Student
        ID: 198329928
        Grade: 10
        Email: student.x@student.tdsb.on.ca

        Name: Teacher X
        Status: Teacher
        ID: 423664716
        Email: teacher.x@tdsb.on.ca
```

In my program, I will use inheritance with the “Node”, “Person”, “Student”, and “Teacher” classes. The UML class diagram for this inheritance hierarchy is displayed below.

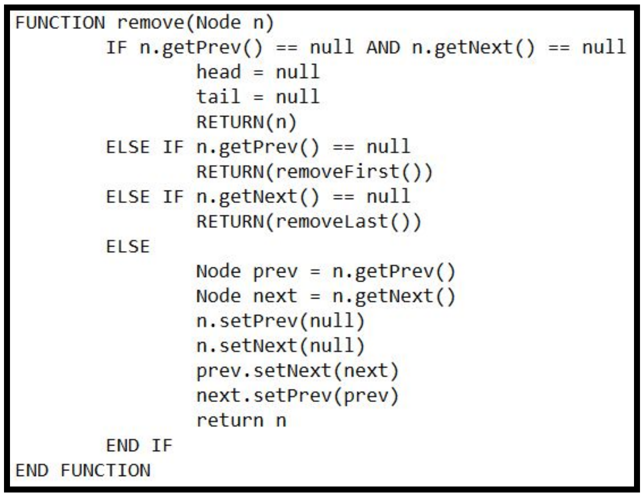


I will create my own linked list class for my program and include important methods in it, such as “addLast()”, “remove()”, and “selectionSort()”.

Function for adding a node to the end of the linked list and removing a node from the linked list (in pseudocode)

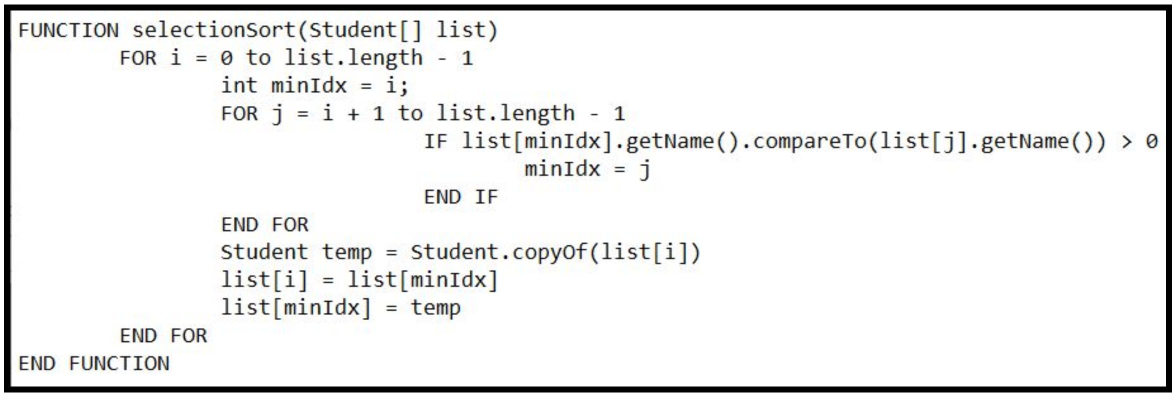


```
FUNCTION addLast(Node n)
        IF tail == null
                head = n
                tail = n
        ELSE
                tail.setNext(n)
                n.setPrev(tail)
                n.setNext(null)
                tail = n
        END IF
END FUNCTION
```



```
FUNCTION remove(Node n)
        IF n.getPrev() == null AND n.getNext() == null
                head = null
                tail = null
                RETURN(n)
        ELSE IF n.getPrev() == null
                RETURN(removeFirst())
        ELSE IF n.getNext() == null
                RETURN(removeLast())
        ELSE
                Node prev = n.getPrev()
                Node next = n.getNext()
                n.setPrev(null)
                n.setNext(null)
                prev.setNext(next)
                next.setPrev(prev)
                return n
        END IF
END FUNCTION
```

Function for sorting the list of students alphabetically (in pseudocode)



```
FUNCTION selectionSort(Student[] list)
        FOR i = 0 to list.length - 1
                int minIdx = i;
                FOR j = i + 1 to list.length - 1
                                IF list[minIdx].getName().compareTo(list[j].getName()) > 0
                                        minIdx = j
                                END IF
                END FOR
                Student temp = Student.copyOf(list[i])
                list[i] = list[minIdx]
                list[minIdx] = temp
        END FOR
END FUNCTION
```

Success Criteria

Action test	**Method of testing**
Program will have a user-friendly login screen and allow the user to create only one account	Asking friends, family, and client if my login screen is user-friendly Creating an account and then trying to create another account to test that only one account can be created
Upon rebooting the program, all previous data that has been entered for the students and teachers will be saved	Entering data for student(s)/teacher(s), exiting the program, and rebooting back the program to see if the data is saved
Adding a student to the program is done successfully	Going to the “Contact information” page or “Edit current students” page to see if the student has been added
The editing of the student’s completed and current courses are done successfully. For example, if a student does not have the prerequisites for a particular course, it is not possible for them to select that course	Testing if a student can take a course that they do not have the prerequisites for Generating a timetable for the student to see if their current courses have been successfully selected
Generation of the student’s timetable is done randomly	Clicking the “Generate student timetable” button many times to see if their timetable is truly random
Sorting the list of students alphabetically is done successfully	Clicking the “Sort students” button and seeing if the names of the students on the scrollbar are sorted alphabetically

Deletion of a student from the database is done successfully	Clicking the “Delete a student” button and seeing if the student is gone from the “Contact information” page
Contact information for students and teachers are successfully displayed	Add some students and teachers to the database and go into the “Contact information” page to see if the student’s and teacher’s contact information are correctly shown
Adding a teacher to the program is done successfully	Going to the “Contact information” page or “Edit current teachers” page to see if the teacher has been added
Sorting the list of teachers is done successfully	Clicking the “Sort teachers” button and seeing if the names of the teachers on the scrollbar are sorted alphabetically
Deletion of a teacher from the database is done successfully	Clicking the “Delete a teacher” button and seeing if the teacher is gone from the “Contact Information” page
History page correctly displays the history of actions that the user has taken	Do many actions (e.g. add two students, add a teacher, remove a student, edit a student’s courses and generate their timetable) and go into the “History” page to see if the correct set of events are displayed in the correct order
Warning messages are displayed for incorrect actions that the user has taken	Do actions that are not allowed and see if a warning message pops up. For example, adding a student without giving the student a name, and seeing if a warning message will pop up
The program should be simple to use	Sending my executable jar file to friends, family, and client to see if they find it simple to use
The program will cite all background images that are taken from the internet and used in the solution	Check if the program correctly displays the image citations in a readable font. These image citations will be provided in the “Image Citations” page

Word Count: 61