

TTTK1143: TakeHomeTest – Question A

Eileen Tong Hui Guan (A180693)

Program proposal:

Classroom name list manager.

The application will keep a file-based database of all classrooms. Data stored would be the names and ID of students in a class, and the names, ID, and subject of the teachers who teach the class.

Users will be able to create new classrooms and save all the details related to it. Data is saved in text format. Once the data is saved, it can be viewed through the application. Saved classroom data may be modified via addition/deletion of data.

Changes are highlighted in red.

[Link to changelog.](#)

Contents

Input, Storage, Output, Modifications	2
Input:	2
Storage:	2
Output:	2
Modification:	3
UML Diagram and Class Definition	4
UML Diagram	4
Class Definition	5
GUI prototype (Input and Output screen design)	8
Main menu	8
Input	10
Output	12
Modify	14
Input and Output file format	16
Input file format	16
Output file format	18
Classroom.txt	18
Students.txt	18
Classname.txt	19
Assumptions/Limitations:	20
Changes	20

Input, Storage, Output, Modifications

Input:

- User submits input through the application, either by filling a form, or by providing a formatted text file (.txt).
- Required details:
 - New Classroom:
 - Classroom:
 - name (String)
 - capacity (int)
 - location (String)
 - Assign classroom members:
 - Classroom teacher:
 - id (String)
 - name (String)
 - subject (String)
 - Students:
 - id (String)
 - name (String)
 - Regular teachers:
 - id (String)
 - name (String)
 - subject (String)

Storage:

- Format: Text files. (.txt)
- Classrooms.txt
 - Stores the full list of classroom names and respective class teachers' names.
 - Classroom names used for file fetching.
 - Both classroom name and class teacher's name used for display in the application's main menu.
- Students.txt
 - Stores the full list of registered students IDs to check for duplicate entries when adding a new student.
- *ClassName.txt*
 - One created for each new classroom. *ClassName* is the assigned name of a class.
 - Stores the ids, names, and details of each person in the class. (Students and teachers.)

Output:

- Details are displayed in the application.
- The following details are displayed on request.
 - List of classroom names/class teachers.
 - Classroom details
 - Classroom name
 - Classroom maximum capacity

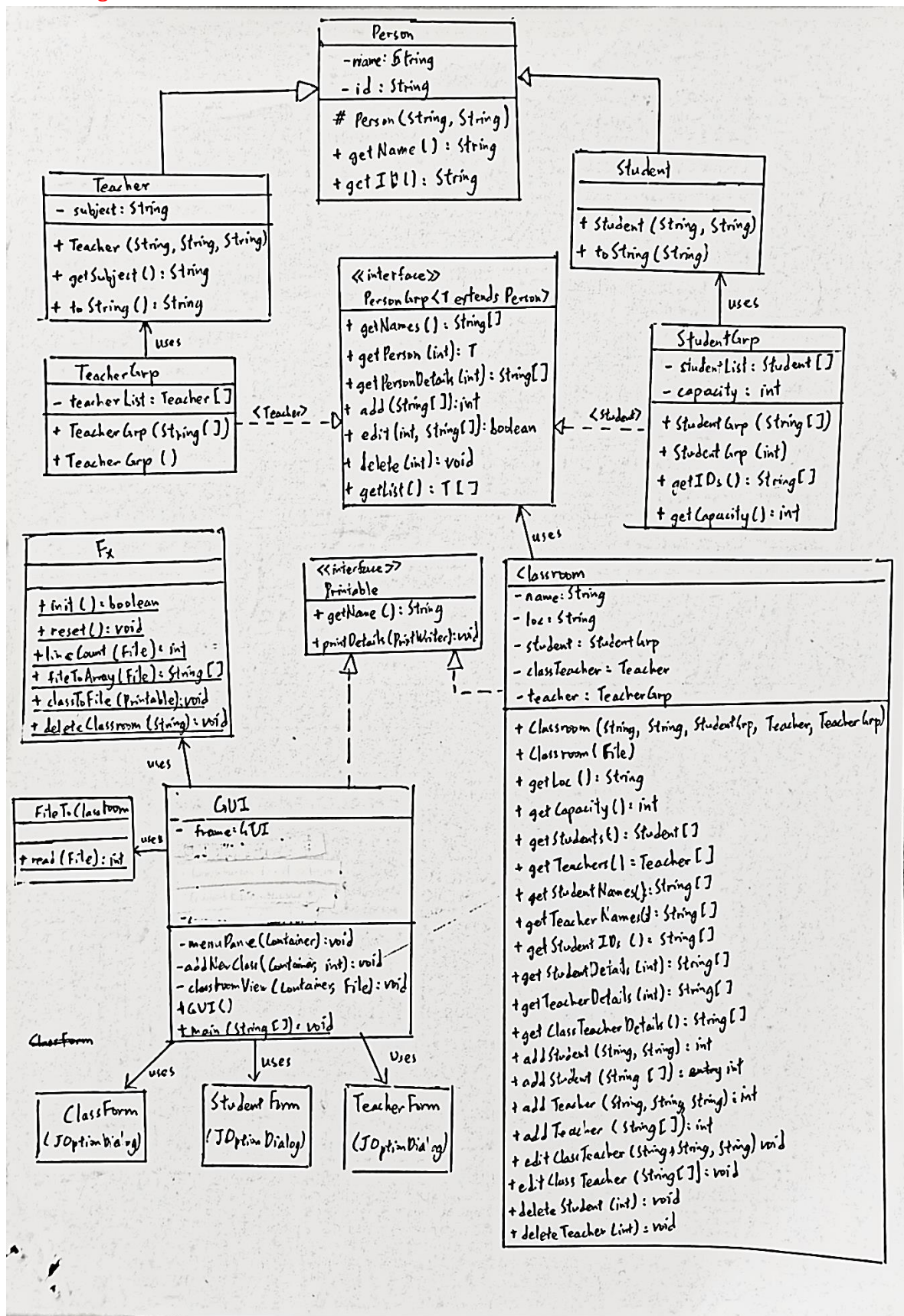
- Classroom's class teacher details
 - Class teacher's name
 - Class teacher's ID
 - Class teacher's subject
- List of students' names
 - Student's details
 - Student's name
 - Student's ID
- List of regular teachers' names
 - Teacher's details
 - Teacher's name
 - Teacher's ID
 - Teacher's subject

Modification:

- The following values can be modified through the application.
 - System
 - Add/delete classrooms.
 - Classroom
 - Change class teacher.
 - Add/delete students.
 - Add/delete regular teachers.
 - Classroom teacher
 - **Change class teacher details.**
 - Students
 - None
 - Regular teachers
 - None

UML Diagram and Class Definition

UML Diagram



Class Definition

Class **Person**

- variable to store the person's unique ID.
- variable to store the person's name.
- constructor to set the person's ID and name.
- method `getID()` to get the person's ID.
- method `getName()` to get the person's name.

Class **Teacher** extends **Person**

- variable to store the subject taught by the teacher.
- constructor to set the teacher's ID, name, and subject.
- method `getSubject()` to get the teacher's subject.
- **method `toString()` to get all their details in String.**

Class **Student** extends **Person**

- constructor to set the student's ID and name.
- **method `toString()` to get all their details in String**

Interface **PersonGrp<T>**

- method `getNames()`
- method `getPerson()` with one parameter for int.
- method `getPersonDetails()` with one parameter for int and returns a `String[]`.
- method `add()` consist of one parameter for `String[]` array and returns an integer for status. (Standard: 0 = no error, 1 = parameter error, 2 = duplicate entry for non-duplicate groups, 3 and above = other error)
- method `edit()` consist of one parameter for integer to specify the entry to edit, and one parameter for `String[]` array to supply arguments for the creation of a `Person` to replace the old entry in `Person[]`. Returns boolean for status. (Standard: true = no error, false = error)
- method `delete()` consist of one parameter for `int[]`.
- **method `getList()` which returns `T[]`**

Class **TeacherGrp** implements **PersonGrp<Teacher>**

- variable `teacherList` to store the list of teachers, in the form of `Teacher[]` array.
- constructor to set the `teacherList`.
- **constructor to set empty `teacherList`**

- the definition method `getNames()` that overrides from Interface `PersonGrp<T>` to get a `String[]` array containing the names of all the people in the group.
- the definition method `getPerson()` that overrides from Interface `PersonGrp<T>` to get a `Teacher()` object.
- the definition method `getPersonDetails()` that overrides from Interface `PersonGrp<T>` to get a `Teacher()` object's details, in the form of `String[]`. Elements are arranged as [*id*, *name*, *subject*]
- the definition method `add()` that overrides from Interface `PersonGrp<T>` to create a new `Teacher()` object, and then replaces the old `Teacher[]` array with a new `Teacher[]` array to add the new `Teacher()` object.
- the definition method `edit()` that overrides from Interface `PersonGrp<T>` to replace an element in `Teacher[]` array with another `Teacher()` object.
- the definition method `delete()` that overrides from Interface `PersonGrp<T>` to remove specified element from `Teacher[]` array.

Class **StudentGrp** implements `PersonGrp<Student>`

- variable `studentList` to store the list of students, in the form of `Student[]` array.
- variable to store student group capacity. `Student[]` may not have more elements than capacity.
- constructor to set the `studentList`.
- constructor to set empty `studentList` with fixed value as max capacity
- method to get student group capacity.
- the definition method `getNames()` that overrides from Interface `PersonGrp<T>` to get a `String[]` array containing the names of all the people in the group.
- the method `getIDs()` to return a list of student IDs.
- the definition method `getPerson()` that overrides from Interface `PersonGrp<T>` to get a `Student()` object.
- the definition method `getPersonDetails()` that overrides from Interface `PersonGrp<T>` to get a `Student()` object's details, in the form of `String[]`. Elements are arranged as [*id*, *name*]
- the definition method `add()` that overrides from Interface `PersonGrp<T>` to add a new student into `studentList`
 - 0 = no error
 - 1 = parameter error
 - 2 = duplicate
 - 3 = exceed capacity
- the definition method `edit()` that overrides from Interface `PersonGrp<T>` to replace an element in `Student[]` array.
- the definition method `delete()` that overrides from Interface `PersonGrp<T>` to remove specified elements from `Student[]` array.
- the definition method `getList()` that overrides from Interface `PersonGrp<T>` to get the maximum capacity of this student group.

Interface **Printable**

- method `getName()` that returns a `String`

- method `printDetails()`

Class **Classroom** implements Printable

- variable `name` to store the name of the classroom.
- variable `loc` to store the location of the classroom.
- constructor with accepts `String`, `String`, `StudentGrp`, `Teacher`, `TeacherGrp` as parameter, and sets the values of all its variables.
- constructor which accepts a `File` as parameter, and sets the values of all its variables.
- method `getName()` to return the name of the classroom.
- method `getLoc()` to return the location of the classroom.
- method `getCapacity()` to return the maximum capacity of class (students)
- method `getStudents()` to return a list of `Students`
- method `getTeachers()` to return a list of `Teachers`
- method `getStudentNames()` to return a list of student's names
- method `getTeacherNames()` to return a list of teacher's names
- method `getStudentIDs()` to return a list of student ids.
- method `getStudentDetails ()` to get the details of the student based on the index given
- method `getTeacherDetails()` to get the details of the teacher based on the index given
- method `addStudent()` to add a new `Student()` into *student*. Accepts (`String name`, `String id`). Returns integer for status.
- method `addStudent` to add a new `Student()` into student group. Accepts array containing {`name`, `id`}. Returns integer for status.
- method `addTeacher()` to add a new `Teacher()` into *teacher*. Accepts (`String`, `name`, `String id`, `String subject`). Returns integer for status.
- method `addTeacher()` to add a new `Teacher()` into *teacher*. Accepts array containing {`name`, `id`, `subject`}
- method `editClassTeach()` to change the details of class teacher. Returns true if successful.
- method `deleteStudent()` to delete a student from `StudentGrp`
- method `deleteTeacher()` to delete a teacher from `StudentGrp`
- the definition method `printDetails()` that overrides from Interface `Printable` which prints the formatted details of classroom. Refer to [classname.txt](#) for format.

Class **Fx**

- static method `init()` to set up files during program startup
- static method `reset()` to reset the data files of program
- static method `lineCount()` to count the lines of a file.
- static method `fileToArray()` to parse a file into `String[]` array for further parsing.
- static method `classToFile()` which writes an object's details to a file. Only accepts objects that implement `Printable`.
- static method `deleteClassroom()` which accepts the name of class (`String`) and deletes it from the system.

Class **GUI**

- variable frame to store the frame of GUI
- variable teachForm to store the TeacherForm custom dialog box
- variable studentForm to store the StudentForm custom dialog box.
- variable classForm to store the ClassForm custom dialog box
- private method menuPane() to store the Main Menu GUI
- private method addNewClass() to store the Classroom Creation GUI
- private method classroomView() to store the Classroom View GUI
- constructor GUI() to resize custom dialog box and initialise program
- main method to create new frame for Classroom Manager and set the frame's attributes.

ClassForm, StudentForm and TeacherForm

- custom dialog boxes that extends JDialog.

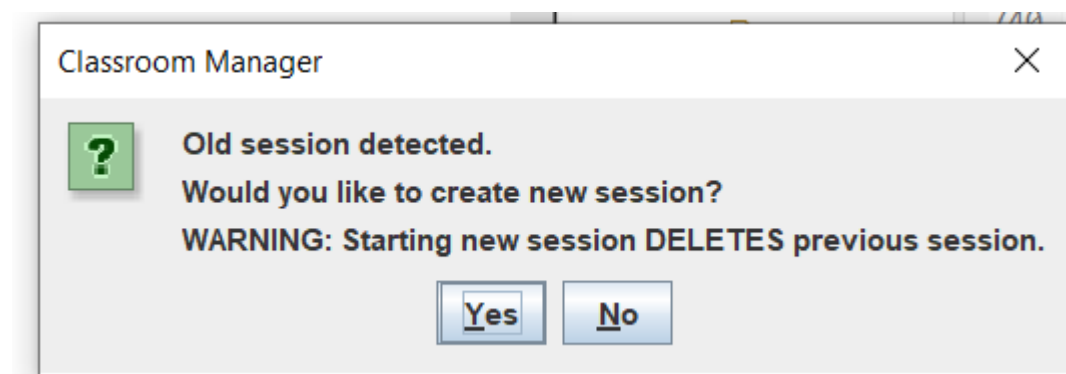
GUI (Input and Output screen design)

GUI size: 450, 592

Usage instructions in [readme.pdf](#)

Main menu

Popup if old session is detected.

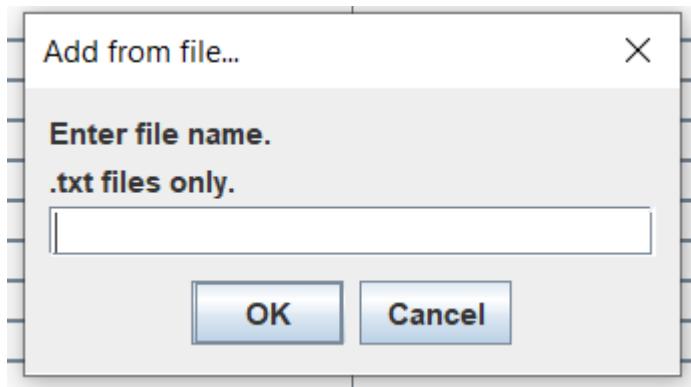


Classroom list. (Main menu)

[illegible]

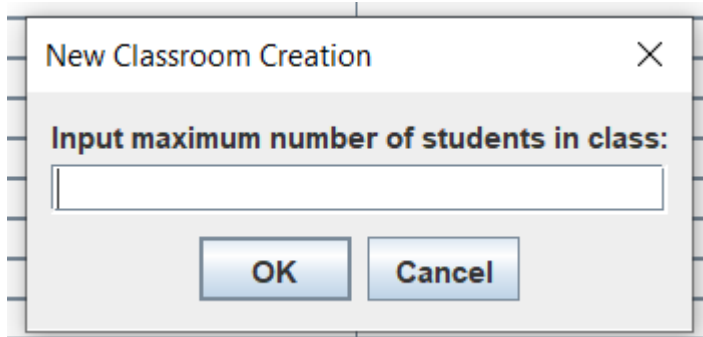
Input

Add from file (Upon pressing “Add from file” button)



A dialog box titled "Add from file..." with a close button (X) in the top right corner. The main text inside the dialog reads "Enter file name." followed by ".txt files only." Below this text is a single-line text input field. At the bottom of the dialog are two buttons: "OK" and "Cancel".

Add from GUI (Upon pressing “Add new...” button)



A dialog box titled "New Classroom Creation" with a close button (X) in the top right corner. The main text inside the dialog reads "Input maximum number of students in class:" followed by a single-line text input field. At the bottom of the dialog are two buttons: "OK" and "Cancel".

Invalid input: Anything other than numbers.

Classroom Manager

Class name:
Location:
Capacity: 12

Edit class details

Class Teacher: None
ID: None
Subject: None

Edit class teacher

Students Teachers

Student name

Add new... Delete

Cancel Save

*Above shows incomplete form. Only class name and location are mandatory.

Input rules:

- No two students may share the same ID.
- Students have a maximum registration of capacity.
- Teachers have a maximum registration limit of 100 people.
- All values in Student and Teacher must not have commas.

Output

Shown after selecting a class from the main menu.

The screenshot shows a window titled "Classroom Manager" with standard Windows window controls (minimize, maximize, close). The window displays the following information:

- Class name:** 1 Adil
- Location:** Block A, Ground Floor
- Capacity:** 20
- Class Teacher:** Pn Masyura
- ID:** K001
- Subject:** English

Below this information is a button labeled "Edit class teacher".

There are two tabs: "Students" (which is selected) and "Teachers".


Under the "Students" tab, there is a list box titled "Student name" containing the following names:

- Imran Hakimi
- Kathryne Wong
- Daniel

At the bottom of the window, there are three buttons: "Add new...", "Delete", and a wide "Save and return" button.

System Library

4

 Classroom Manager

—

□

×

Class name: 1 Adil

Location: Block A, Ground Floor

Capacity: 20

Class Teacher: Pn Masyura

ID: K001

Subject: English

Edit class teacher

Students

Teachers

Teacher name

Pn Aidawati

Add new...

Delete

Save and return

Problems

@ javadoc

Declaration

Console

Modify

Data can be edited from viewing output.

Edit class teacher

The screenshot displays a web application interface for managing class teachers. On the left, a sidebar contains a list of class teachers, with 'Class Teacher: Pn Masyura' selected. The main content area shows the details for this teacher: ID: K001, Subject: English, and a button labeled 'Edit class teacher'. Below this, there are tabs for 'Students' and 'Teachers', with 'Teachers' currently active. The 'Teacher name' field shows 'Pn Aidawati'. An 'Add Teacher' modal dialog is open in the center, allowing the user to add a new teacher. The dialog contains three input fields: 'Teacher' with the value 'Nagapan a/p Harikrish', 'ID' with the value 'T1001', and 'Subject' with the value 'Pendidikan Moral'. At the bottom of the dialog are 'Enter' and 'Cancel' buttons.

Class Teacher: Pn Masyura
ID: K001
Subject: English
Edit class teacher

Students Teachers

Teacher name
Pn Aidawati

Add Teacher

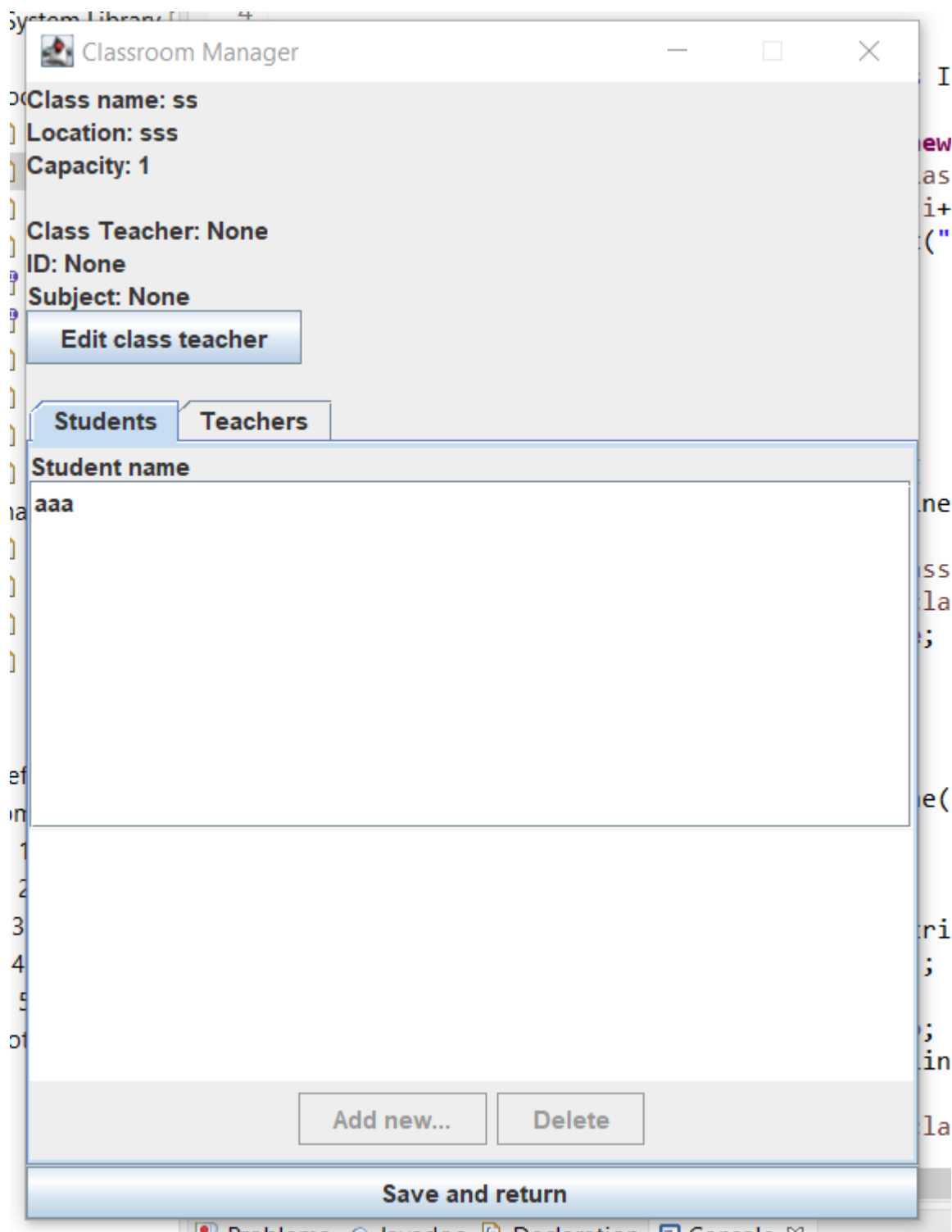
Teacher: Nagapan a/p Harikrish
ID: T1001
Subject: Pendidikan Moral

Enter Cancel

Attempt to add students when maximum capacity reached.

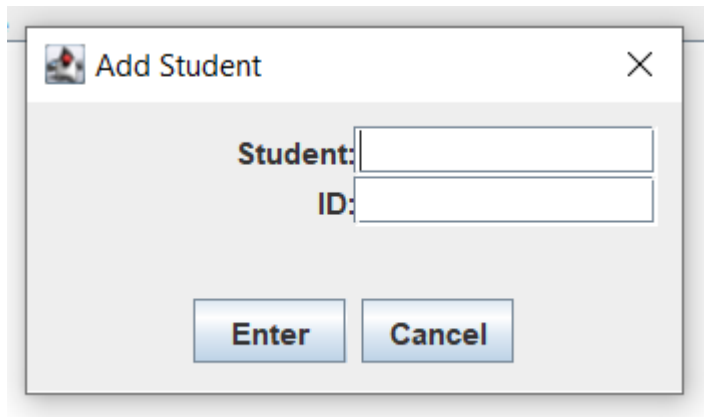
("Add new.." button will be disabled.)

Notice the capacity is 1.



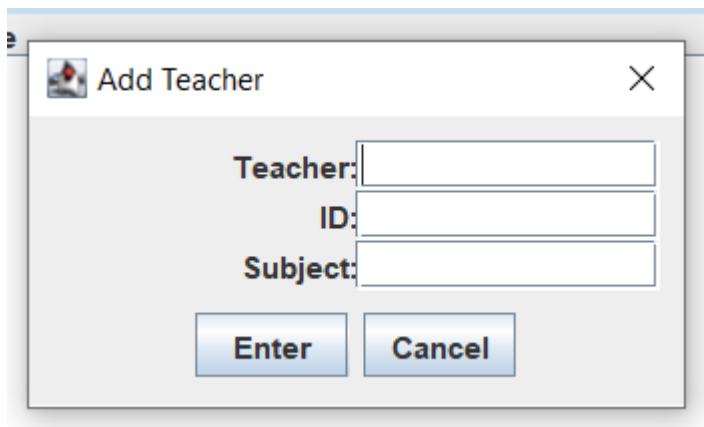
Add student dialog box

If duplicate data, dialog box will close and nothing will happen upon pressing “Enter”.



A dialog box titled "Add Student" with a close button (X) in the top right corner. It contains two text input fields: "Student:" and "ID:". Below the fields are two buttons: "Enter" and "Cancel".

Add teacher dialog box



A dialog box titled "Add Teacher" with a close button (X) in the top right corner. It contains three text input fields: "Teacher:", "ID:", and "Subject:". Below the fields are two buttons: "Enter" and "Cancel".

Input and Output file format

Input file format

format.txt	example.txt
1 Number-of-classes	1 2
2 Class-name	2 1 Adil
3 Location	3 Block A, Ground Floor
4 Maximum-capacity	4 20
5 Number-of-students	5 3
6 Student-name, Student-id	6 Imran Hakimi, A0001
7 Number-of-teachers-including-class-teacher	7 Kathyne Wong, A0002
8 Class-teacher-name, Class-teacher-id, Teacher-subject	8 Daniel, A003
9 Teacher-name, Teacher-id, Teacher-subject	9 2
9 Teacher-name, Teacher-id, Teacher-subject	10 Pn Masyura, K001, English
	11 Pn Aidawati, K002, Geografi
	12 1 Bestari
	13 Block B, Ground Floor
	14 10
	15 0
	16 1
	17 Cik Nik Ina, K003, Sains
	18
	19

Breakdown of example.txt

Line 1: Number of classes (2 classrooms to be scanned)

Line 2: Class 1 name

Line 3: Class 1 location

Line 4: Class 1 maximum capacity (20 students)

Line 5: Class 1 current number of students (3 students to be scanned)

Line 6: Name of first student, ID of first student

Line 7: Name of second student, ID of second student

Line 8: Name of third student, ID of third student

Line 9: Class 1 current number of teachers (2 teachers to be scanned)

Line 10: Name of Class 1 classroom teacher, ID of classroom teacher, subject taught

Line 11: Name of Class 1 regular teacher, ID of regular teacher, subject taught

Line 12: Class 2 name

Line 13: Class 2 location

Line 14: Class 2 maximum capacity (10 students)

Line 15: Class 2 current number of students (0 students to be scanned)

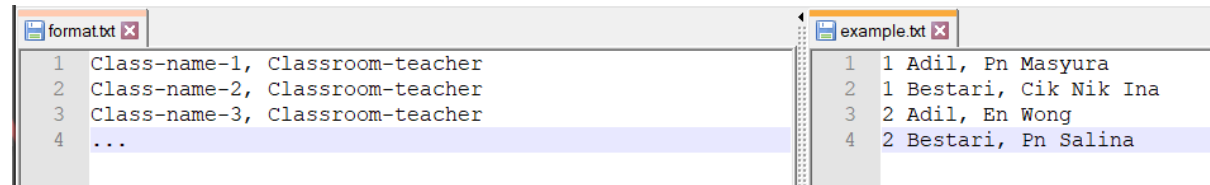
Line 16: Class 2 current number of teachers (1 teacher to be scanned)

Line 17: Name of Class 2 classroom teacher, ID of classroom teacher, subject taught

If an error occurs, the application will state at which line the input is faulty and will terminate the entire input process. No data will be saved if the input process terminates early.

Output file format

Classroom.txt



```
format.txt
1 Class-name-1, Classroom-teacher
2 Class-name-2, Classroom-teacher
3 Class-name-3, Classroom-teacher
4 ...

example.txt
1 1 Adil, Pn Masyura
2 1 Bestari, Cik Nik Ina
3 2 Adil, En Wong
4 2 Bestari, Pn Salina
```

Each line contains

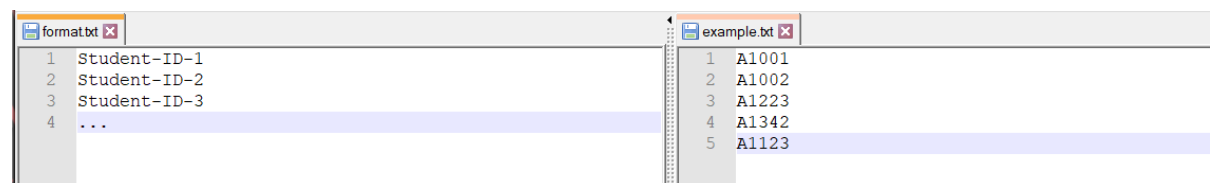
- Class name
- Classroom teacher name

Both separated by a comma.

Each entry is a new line.

Program will scan until the end of text file.

Students.txt



```
format.txt
1 Student-ID-1
2 Student-ID-2
3 Student-ID-3
4 ...

example.txt
1 A1001
2 A1002
3 A1223
4 A1342
5 A1123
```

Each line contains the unique ID of a student.

Order of entry does not matter.

Classname.txt

```
format.txt
1 Class-name
2 Class-location
3 Class-capacity
4 Student-name, Student-id
5 Student-name, Student-id
6 Student-name, Student-id
7 ,
8 ,
9 ,
10 ...
11 Class-teacher-name, Class-teacher-id, Teacher-subject
12 Teacher-name, Teacher-id, Teacher-subject
13 Teacher-name, Teacher-id, Teacher-subject

1 Adil.txt
1 1 Adil
2 Block A, Ground Floor
3 20
4 Imran Hakimi, A0001
5 Kathryne Wong, A0002
6 Daniel, A003
7 ,
8 ,
9 ,
10 ,
11 ,
12 ,
13 ,
14 ,
15 ,
16 ,
17 ,
18 ,
19 ,
20 ,
21 ,
22 ,
23 ,
24 Pn Masyura, K001, English
25 Pn Aidawati, K002, Geografi
26
```

Special guidelines:

- File name is the same as class name.
- If the number of students is less than maximum capacity, the text will leave space for the empty student slots.
- The first teacher listed in the text file is the class teacher.
- If there is no class teacher, the class teacher's details will be replaced with "None, None, None".

example.txt breakdown

Line 1: Class name

Line 2: Class location

Line 3: Class maximum capacity

Line 4: Student name, student ID

Line 5: Student name, student ID

Line 6: Student name, student ID

Line 7 - 23: No students, so left blank. Format ", "

Line 24: Class teacher name, class teacher ID, class teacher subject

*Note: if there is no class teacher, will be replaced with "None, None, None"

Line 25: Regular teacher name, regular teacher ID, regular teacher subject

Program will scan for teacher entries until the end of file.

Assumptions/Limitations:

Newly added assumptions in red.

- Only the names, id, and positions of persons shall be kept.
- Each classroom information will be recorded separately and independently from each other.
 - Classroom names must be unique.
 - General:
 - All values for students and teachers must not have commas.
 - All students and teachers' IDs must not have spaces.
 - Teachers:
 - A teacher may have duplicate entries across classroom files. This is because a teacher may teach multiple classes, and they may teach multiple subjects in the same class.
 - A teacher's entry will not be checked for duplicates.
 - Should a teacher need to be fired/removed from a system, their entries must be removed manually from each classroom.
 - A maximum of 100 teachers can teach a class. It is possible to add more teachers in during classroom creation and classroom edit/view, but entries beyond 100 teachers will not carry between sessions.
 - Students:
 - A student will be prevented from having duplicate entries by checking their student ID during the addition of new students.
 - If a student needs to be transferred to another class, they will need to be manually removed from their current class and be added to another class.
- A classroom may have no teachers, students, or class teacher if none are assigned.
- All IDs are manually assigned by user.
- User must NOT tamper with system files.
- Every change made in GUI will cause changes in the file immediately. There is no undo.

Changes

- Assumptions:
 - all values for Students and Teachers must not have commas.
 - All Students and Teachers' IDs must not have spaces.
 - Maximum 100 teachers can teach a single class.
 - Class names must be unique.
 - Every change made in GUI will be applied to file. There is no undo.
- UML
 - Overriden toString for Teacher and Student. Simply returns all variables separated by commas. (Refer [Classname.txt](#))
 - Changed order of declaration for Person into (String name, String id). Affected areas:
 - In UML diagram, changed Person constr params order to Person(String name, String id). Also affects all children.
 - TeacherGrp, method getPersonDetails returns a String[] with format [name, id, subject]
 - StudentGrp method getPersonDetails returns a String[] with format [name, id]

- PersonGrp implementations
 - TeacherGrp implements PersonGrp<Teacher>
 - StudentGrp implements PersonGrp<Student>
 - delete() now accepts only an integer (Previously int[])
- Added getList() to PersonGrp<T>. Returns T[] list.
- Add error status for PersonGrp add(): Int value more than 2 means other error.
Refer to implementation for more details.
 - StudentGrp: int 3 means exceed capacity. Checked by checking last entry.
 - StudentGrp: int 4 means Student.txt missing. Should error occurs, prompt to reset session. (All data will be lost)
- StudentGrp add constr: get capacity, empty group with no entries
- TeacherGrp add constr: get empty group with no entries
- StudentGrp changed constr: only accepts array. Capacity derived from array length.
- StudentGrp and classroom: added getIDs()
- Interface Printable changed printDetails() to require PrintWriter as parameters.
- Classroom add constr for raw/new classroom entry. Accepts values for all its variables and initialises with received values.
- Classroom add variables: StudentGrp and TeacherGrp
- Classroom several get methods involving students and teachers are separated into two
 - getList() into getStudentNames() and getTeacherNames()
 - getPersonDetails into getStudentDetails() and getTeacherDetails()
 - deletePerson() into deleteStudent() and deleteTeacher()
 - Reasoning: having option as parameters implies any object with PersonGrp interface can get their own list, which is not intended. Only students and teachers' names can be fetched from classroom.
 - Accordingly, the PersonGrp requirement for parameters are dropped.
- Classroom: add getCapacity method.
- Classroom: add editClassTeach method and getClassTeachDetails method.
- Classroom: add getStudents and getTeachers method (returns Student[]/Teacher[])
- Overloaded several methods in Classroom.java, StudentGrp.java and TeacherGrp.java.
-
- Added static methods in Fx.java
 - init()
 - reset()
 - lineCount(File)
- New class: FileToClassroom.java, class creation from file. Accepts a file.
 - returns integer as status.
 - 0 = no errors
 - 1 = File not found
 - 2 = Other exception. (Input format/value error)
- Added new class files in package main.
 - ClassForm.java (Custom JOptionPane for entering className and classLocation)
 - StudentForm.java (Custom JOptionPane for entering student details.)
 - TeacherForm.java (Custom JOptionPane for entering teacher details.)
- New fields in GUI.java

- GUI frame
 - int extraWindowWidth
 - Teacherform teachForm
 - StudentForm studentForm
 - ClassForm classForm
- New methods in GUI.java
 - void menuPane(Container pane)
 - Contains the Main Menu GUI
 - void addNewClass(Container pane, int cap)
 - Contains the Add New Class GUI
 - void classroomView(Container pane, File classFile)
 - Contains the Classroom Detail View
 - Constructor GUI()
 - packs the custom dialog boxes
 - Runs the Fx.init() method
 - Initialises the first GUI pane. (menuPane())
- File format
 - *Classname.txt*, blank student entries changed from “ , “ to “ , “. (Removed space)
- GUI
 - Changed size to 450, 592
 - Changed “Create new classroom from GUI”
 - User must enter max capacity first before entering data.
 - Interface changed to resemble classroom detail view.
 - Program reset question changed to “Would you like to start a new session?”
 - Add “View class” to classroom main menu.
 - Add popup dialog when “Add from file” button is pressed. Prompts for file name.
 - Removed current number of student from Class Capacity in Classroom Details view.
 - Student and teacher list in Classroom Details view: can only select one at a time.
 - Classrom Details view: Changed “delete selection” button to “delete”
 - Certain buttons automatically disable when a function is not allowed
 - Main menu: no classroom selected, view/delete disabled
 - Edit student: add button disabled when max capacity
 - Edit student & teacher: delete button disabled when nothing selected
 - Create new class from GUI: prompts the maximum number of students in a class first.
 - Removed “Delete confirmation dialog box”. Entries are immediately deleted upon pressing the delete button.