Readme

Created by:

Koishore Roy

Mihika Sood

Vijay Lingam

Contents

Introduction

ClassAllot is a software which simplifies the process of allocating classrooms to courses at Ashoka (or any other) university. Just input the list of courses along with faculty preferences, the list of classrooms and the list of available time slots and let the program do its magic. Sit back and relax while the program allocates all the courses to free classrooms, or tells you about the inconsistencies for which allocation isn't possible. Professors will love it!

The software basically implements a backtracking recursion algorithm which takes in a list of courses which mentions the professor teaching them, the capacity a particular course, the professor's teaching time preference and the list of courses it cannot clash with. It then sees the number of rooms available and their capacities. Finally, the program reads through the list of faculty teaching timings, which are numbered and each number corresponds to a certain time. The program then multiplies the number of rooms with the number of slots to get the total number of teaching slots available, which is made into a matrix. The list of courses is then called by the program and the program starts assigning courses to the matrix based on faculty preferences shuffled in a random order and room capacity. If the first preference is not satisfied and the position in the matrix is already occupied, then it returns to the preference list and takes the next number into consideration. If a list of preferences is completely exhausted; then the program returns to the previous course and reallocates it to the next preference slot and retries to allocate the next course. If no such allocation can be found then it keeps on going back up until the first course in the list. If no allocation can be found despite repetitive iterations till the first course in the list, the program empties the contents of the matrix and tells the user that no allocation can be found. For other conflicting and impossible cases such as more courses than slots available, the program doesn't run altogether and warns the user about the limitations of the input file(s).

Input Files

The program requires 3 input files:

1) main.csv (args1)
2) Classrooms.csv (args2)
3) Timings.csv (args3)

The input files need to be placed in the same directory as the program (.java file), and a certain format needs to be followed while making the files, which will be explained in the next segment. The names of the files may or may not be same, but whatever the user chooses to name the file, they must explicitly mention the name of the file while running the program. the 'n' in 'args(n)' refers to the order that needs to be followed while running the file.

File Formatting

A specific format must be followed while creating the files for the program to process and read and run. Below is the specific format with examples. To make a .csv file, either open a text editor like NotePad on Windows, TextEdit on OS X, or gedit on Ubuntu (Linux) and write values in the same row separated by a comma and for multiple values inside the same cell use quotation marks (""). Otherwise, open a spreadsheet software like Excel on Windows, Numbers on OS X or Calc on Ubuntu (Linux) and save the filled file as a .csv file.

---

main.csv

This is the first file in the command line arguments list and by far the most important. It contains 5 columns namely; Course, Professor Name, Teaching Preferences, Capacity, Clashes. Fill in the respective slots with information provided and save the file in the same directory as the program. For example, if the course name is CS-205-01, which is taught by Professor Sudheendra Hangal, who has faculty preference 3, 4, 5, 8, 9, 10*(refer to Timings.csv for more details) and the class can accommodate 25 students. Another criteria for the course would be that it does not clash with CS-204-01 and CS-203-01, then write it in the following manner:

Courses	Professor Name	Teaching Preferences	Capacity	Clashes
CS-205-01	Sudheendra Hangal	3, 4, 5, 8, 9, 10	25	CS-203-01, CS-204-01

Classrooms.csv

This is the second file in the command line arguments list. It contains 3 columns namely; Serial, Classroom, Capacity. Fill in the respective slots with information provided and save the file in the same directory as the program. For example, the first two rooms to be allotted will have Serial as 1 and 2 respectively. Let their names be TR-102 and LR-103 respectively and the classes can accommodate 25 and 50 students respectively, then write it in the following manner:

Serial	Classroom	Capacity
1	TR-102	25
2	LR-103	50

Timings.csv

This is the third and final file in the command line arguments list. It contains 3 columns namely; Slot Number, Day, Duration. In this case, the number of rows will be equal to the number of slots available at Ashoka University. Fill in the respective slots with information provided and save the file in the same directory as the program. For example, write it in the following manner:

Slot Number	Day	Duration
1	Monday	9:00am-10:30am
2	Monday	10:40am-12:10pm
3	Monday	12:20pm-1:50pm
4	Monday	2:35pm-4:05pm
5	Monday	4:15pm-5:45pm
6	Tuesday	9:00am-10:30am
7	Tuesday	10:40am-12:10pm
8	Tuesday	12:20pm-1:50pm
9	Tuesday	2:35pm-4:05pm
10	Tuesday	4:15pm-5:45pm

How to Run

To run the program, open Terminal and navigate to the program directory using the cd command. Check for all files using the ls command and then execute the following commands to run the program:

```
>> javac -cp commons-csv-1.2.jar Allocator.java
>> java -cp .:\* Allocator arg1 arg2 arg3
```

A few key points:

- Here, Allocator is the name of the main .java file, which would need to be compiled using javac and having Java SE8 or Java EE installed on the particular computer.
- args1, args2 and args3 are the respective names of the aforementioned files, so be careful and name them properly.
- For getting a result, have as many faculty preferences slots as possible.
- To get at least an answer, if possible, DO NOT have more courses that the product of the number of slots and the number of rooms.