CS 2336 PROJECT 1 – GotG2 Ticket Reservation System (Maintenance Project 1)

Pseudocode Due: 1/20 by 11:59 PM

Project Due: 2/4 by 11:59 PM

Submission and Grading:

- All project deliverables are to be submitted in eLearning.
- The pseudocode should be submitted as a Word or PDF document and is not accepted late.
- Projects submitted after the due date are subject to the late penalties described in the syllabus.
- Programs must compile and run with JDK 8.
- Each submitted program will be graded with the rubric provided in eLearning as well as a set of test cases. These test cases will be posted in eLearning after the due date. Each student is responsible for developing sample test cases to ensure the program works as expected.
- **Type your name and netID in the comments at the top of all files submitted.**

Objectives: Create a Java program using programming fundamentals (file I/O, loops, conditional statements, arrays, functions)

Problem: In preparation for the release of Guardians of the Galaxy 2, you have been hired by the owner of a small movie theater to develop the backend for an online ticket reservation system. Patrons will be able to reserve seats in one of three auditoriums. Once the patron has selected an auditorium, the program should display the current seating arrangement and allow the patron to select seats. A report should be generated at the end of the program to specify for each individual auditorium and overall for all auditoriums how many seats were sold/unsold and how much money was earned.

Details

- The seating arrangement for each auditorium will be stored in separate files. These files will be named *A1.txt, A2.txt* and *A3.txt* for auditorium 1, 2 and 3 respectively.
- Each line in the file will represent a row in the auditorium. The number of rows in each auditorium is unknown to you.
- The number of seats in each row of a **single** auditorium will be the same. For example, if the first line of the file has 15 seats, then every subsequent row in the theater will also have 15 seats. This does not mean that each auditorium has the same number of seats in each row. One auditorium may have 15 seats per row and another may have 20 seats.
- Each auditorium will be held in a two-dimensional array.
- Empty seats are represented by a pound sign (#).
- Reserved seats are represented by a period (.).
- Tickets can only be reserved the day of the screening and all screenings are at 7 PM that night. There is no need to worry about multiple screenings or reserving seats on a specific day.
- All tickets are $7 regardless of patron age or auditorium.

User Interface and Input: Present a user-friendly menu system for the user to select the auditorium. First ask for the auditorium:

```
1. Auditorium 1
2. Auditorium 2
3. Auditorium 3
4. Exit
```

Although in reality the user would likely only make one purchase, for testing purposes, assume the user will repeat the ticket buying process until they decide to quit.

Once the auditorium has been selected, display the current seating availability for that auditorium. An example seating chart is provided below for an auditorium with 5 rows and 20 seats per row.

```
  12345678901234567890
1 ...##..#####........
2 ########....####..##
3 .........##.........
4 #.#.#.#.#.#.#.#.#.#.
5 ########.#####.#####
```

The seats are numbered sequentially from left to right and only the ones digit is displayed above each column to make it easier to display the chart. It is understood that the second set of digits from 1-0 are for the numbers 11-20 in the above example.

After the user has selected the auditorium and the seating chart has been displayed, prompt the user for the following information in the order below:

- Row number
- Starting seat number
- Number of tickets

Assume that the user wants to reserve sequential seats to the right of the first seat entered.

If the desired seats are not available, offer the user the best available seats that meet their criteria **on that row only**. The best available seats are the seats closest to the middle of the row. Prompt the user to enter a **Y** to reserve the best available or **N** to refuse the best available. Once the selection has been processed, return to the main menu.

All input will be of the valid data type. You do not have to worry about the user entering a letter when a number is expected or a floating point number when an integer is expected. You are responsible for validating that the data falls within the proper range and that a user does not try to reserve a seat that is already reserved.

Output: At the end of the program, write the current status of each auditorium to the respective file. Also, display a formatted report to the console. The report should consist of 4 columns:

- Column 1 – labels
  - Auditorium 1
  - Auditorium 2
  - Auditorium 3
  - Total

- Column 2 - number of seats reserved for each label
- Column 3 - the number of open seats for each label
- Column 4 - the total of the ticket sales for each label