Project 1

Ticket Reservation System

KEY ITEMS: Key items are marked in red. Failure to include or complete key items will incur additional deductions as noted beside the item.

Submission and Grading:

- The pseudocode will be submitted in eLearning as a PDF document and is not accepted late.
- All project source code will be submitted in Zybooks.
- Projects submitted after the due date are subject to the late penalties described in the syllabus.

• Type your name and netID in the comments at the top of all files submitted. (-5 points)

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

Specification

Problem: The owner of a small theater in an isolated town in Montana has hired you to develop the backend for an online ticket reservation system to handle the customer demand. Your program will allow attendees to select seats and display the current seating arrangement. At the end of the program, a report will be generated for the owner indicating how many seats were sold/unsold and how much money was earned.

Details

To avoid potential errors when grading, do not create multiple Scanner objects for System.in. If the Scanner object is needed in multiple functions, please pass the Scanner object into the function.

Do not create a package.

The seating arrangement for each auditorium will be stored in a file.

Prompt the user for the filename at the beginning of the program

Each line in the file will represent a row in the auditorium. The number of rows in the auditorium is unknown to you. There will be a newline character after each line in the file except for the last line which may or may not have a newline character.

The maximum number of rows is 10.

The number of seats in each row of the auditorium will be the same.

No row will have more than 26 seats.

There will not be more than 10 rows.

Each auditorium will be held in a two-dimensional array, 10 rows x 26 columns. (-10 points if not)

Empty seats are represented by a period (.).

Reserved seats are represented by a letter (A, C or S) **in the file**
This will be used to create reports

- A = adult
- C = child
- S = senior

Reserved seats will be represented by a hashtag (#) **on the screen**
The user does not need to know what type of ticket was sold, just that a seat is reserved.

There is no need to worry about multiple screenings or reserving seats on a specific day.
Ticket prices are as follows:

- Adult - $10
- Child - $5
- Senior - $7.50

User Interface and Input: Present a user-friendly menu system for the user.

```
   1. Reserve Seats
   2.  Exit
```

Loop the menu until the user decides to quit. Imagine this is for a ticket kiosk in the lobby of the theater.
If the user wants to reserve seats, 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.

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

The rows are numbered, and the seats are identified in each row by a letter of the alphabet

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

- Row number
- Starting seat letter
- Number of adult tickets
- Number of child tickets
- Number of senior tickets

Assume that the user wants to reserve sequential seats to the right of the first seat entered. Adult tickets will be reserved first, followed by child and then senior. All seats must be open for a reservation to be processed.

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 measured by the distance from the center of the selection to the middle of the row. In case of a tie pick the seats with lower seat number. 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. If there are no alternate seats available, display **`no seats available`** to the user instead of a prompt and 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. If the user's selection extends past the end of a row, consider this invalid.

User Interface Workflow: Please do not add extra prompts since this will cause a mismatch in the input which will either force the program to throw an exception or cause the program to perform an unintended operation.

- `Prompt for filename`
- `Display main menu`
- `Prompt for input`

If user is reserving tickets

- `Prompt for row`
    - Validate – loop until valid
    - Valid row = row number listed in auditorium display

- `Prompt for starting seat`
    - Validate – loop until valid
    - Valid seat = seat letter listed in auditorium display

- `Prompt for number of adult tickets`
  - Validate – loop until valid
  - Valid ticket number = number >= 0
- `Prompt for number of child tickets`
  - Validate – loop until valid
  - Valid ticket number = number >= 0
- `Prompt for number of senior tickets`
  - Validate – loop until valid
  - Valid ticket number = number >= 0

If selected seats available complete the booking. If user selected seats unavailable:

Display best available

- `Prompt user to reserve (Y/N)`

If options Y is selected complete the booking

- Return to main menu

Loop to top of workflow until user selects exit

Output: Display the best available seats in the following format:

```
<row>< starting seat> - <row><ending seat>
```

Example: `3D – 3F`

At the end of the program, write the current status of the auditorium to `A1.txt`. Remember to write the auditorium reservations using A, C and S to identify the type of ticket sold. Also, display a report to the console using the following format:

```
Total Seats: <value>
Total Tickets: <value>
Adult Tickets: <value>
Child Tickets: <value>
Senior Tickets: <value>
Total Sales: <value>
```

Total amount of money collected for tickets in the whole auditorium. Not just the ones purchased using the program.

All values except total ticket sales will be an integer value. Total ticket sales will be a floating-point value rounded to 2 decimal places and formatted with a dollar sign before the first digit of the number.