
Deadline:	Friday, March 21 2014, midnight.
Evaluation:	5% of final mark.
Late Submission:	No late assignments accepted for credit
Teams:	The assignment can be done individually or in teams of 2 – both students must be from the same lecture section.
Purpose:	The purpose of this assignment is to help you practice I/O and exception handling.

Spartan Race Version 3.0:

In this assignment, you will update your Spartan Race program to accommodate the following new specifications. Note that these new specifications are "in addition" to the previous assignment.

Your program will now be made robust by handling exceptions, and the "debugging mode" of the race will now be handled by an input text file and some output will be written to an output text file.

1. Your program will first ask the user (on the monitor) in which mode the game will be played (debug or real-time), and will read the answer from the keyboard. This is identical to assignment 2.

2. In the debugging mode, as opposed to reading player details and moves of the players from the keyboard, your program will read these values from a text file named "debug_input.txt". Sample "debug_input.txt" files are available on Moodle for you to test your program.
   - If the input file does not exist, your program will show a message to the user on the console, and provide instructions on how to create the file.
   - If the file debug_input.txt exists, your program will read its content and play the game with the information provided in the file. Any possible exceptions must be handled appropriately, and result in a gracious behavior from your program.
     The format of the input file will be follows: First, the number of players will be indicated, followed by the name of player #1, type of player #1, name of player #2, type of player #2 (…include entries for all players). Then the moves of all players will be specified in the following fashion:
     dievalue1 dievalue2 jokerchoice
     dievalue1 dievalue2 jokerchoice
     ….
     In each move line: you first specify the value of dice roll 1, next the value of dice roll 2, and then the choice value for a Joker cell (c for "Pick a card" and w for "Spin the wheel"). The Joker choice will be used only if the player lands in a Joker cell (i.e., ignored for all other cases). You can also check the provided sample files to better understand the input format.

     When reading the input file, your program must handle all types of error cases and halt its execution if the input file does not contain the expected data. In particular,
     - If the value of a specified die value is not between the 1 and 6, then a DieValueOutOfBoundsException must be thrown, and the program should display a message on the monitor and halt the program.
     - If any other data does not have a legal value (e.g., the number of players is not legal, the type of player is invalid, the Joker choice value is invalid, the Joker choice value is missing, etc.), your program should display a message on the monitor and halt the program.
     - If the format of any data is not as expected (e.g., you expect an integer but the file contains a character), then your program must use an appropriate exception handling mechanism to display a message on the monitor and halt the program.

3. In real-time mode:
   - If the user enters illegal values for any data (e.g., the number of players, types of players, etc.) the program will not halt (as it would in debugging mode), but will ask the user to try again to enter a correct value.
   - If the user enters information in the wrong format (e.g., the program expects an integer but the user enters a character), then your program must use an appropriate exception handling mechanism to display a message on the monitor and halt the program.

4. Both in debugging and real-time modes, some output of your program will be stored in a text file and some will be displayed on the monitor. Specifically:
    - On the monitor, your program will display all output (as in assignment 2), except for the grid. The grid should not be displayed on the screen.
    - In a file called "race_out.txt", your program will display all output, including the grid.

   Your program must handle all possible output file exceptions (e.g., the output file cannot be written) with proper exception handling mechanism and result in a gracious behavior from your program.

To implement the new specifications above, you can build upon your own assignment 2 or start with the solution given to you (available on Moodle).

JavaDoc Documentation:
As usual, documentation for your classes must be written in **javaDoc**.
In addition, the following information must appear at the top of each file:

```
Name(s) and ID(s)      (include full names and IDs)
COMP249
Assignment #           (include the assignment number)
Due Date               (include the due date for this assignment)
```

Evaluation Criteria for Assignment 3 (20 points)

Files and Exception Handling: 15pts

Getting input in debugging mode (see item 2 above) i.e., correct handling of file and possible I/O exceptions, correct behavior in case of invalid data, design and use of the DieValueOutOfBoundsException class	10pts
Getting input in real-time mode (see item 3 above)	2pts
Generating output (see item 4 above) i.e., correct handling of file and possible I/O exceptions.	3pts

Satisfaction of the output format specifications (i.e., required information in the correct output stream in the correct format)	5pts

Submitting Assignment 3

When you have finished the program, you must submit the assignment online.

1) Create **one** zip file, containing the necessary files (.java and .html)

Please name your file following this convention:
- If the work is done by 1 student: Your file should be called *a#_studentID*, where # is the number of the assignment *studentID* is your student ID number.
- If the work is done by 2 students: The zip file should be called *a#_studentID1_studentID2*, where # is the number of the assignment *studentID1* and *studentID2* are the student ID numbers of each student.

2) Upload your zip file at the URL: https://fis.encs.concordia.ca/eas/ as *Programming Assignment 3*

Note: Assignments not submitted in the right location and/or not in the requested format will not be graded.