CSE 201 - Advanced Programming (Monsoon 2017)
Course Project Option-2
Chain Reaction Multiplayer Game

Instructor: Vivek Kumar

IMPORTANT Instructions:

1. **It's mandatory that you attend all the deadlines in this project as per the schedule. No request for rescheduling the demo will be entertained. In case of any unavoidable circumstances you have to take email approval from me well in advance.**
2. **Each group should choose their project latest by 23:59pm on 03/10. Absolutely no changes are allowed after this deadline. Either of the two members in any group should ensure they choose the project in case other group member is unavailable until this deadline.**
3. **Before you start working on this project, you MUST go through the slides on "Pair Programming" that was presented in Tutorial-8b.**
4. **You MUST have a PRIVATE git repository for your project and every group member should very frequently check in their code in this repository.**
5. **No extensions will ever be provided. Any submission after the deadline will not be evaluated. If you see an ambiguity or inconsistency in a question, please seek a clarification from the teaching staff.**

Plagiarism: All submitted deliverables are expected to be the result of your individual effort. You should never misrepresent someone else's work as your own. In case any plagiarism case is detected, it will be dealt as per new plagiarism policy of IIITD that was also discussed in the lecture.

Create a JavaFX application for a multiplayer game - *Chain Reaction*. It's a strategy game for 2 to 8 players.The objective of this game is to take control of the board by eliminating your opponents.

Rules of the game is mentioned below -
(Given for 2 players but can be generalized to any number of players)

1. The gameplay takes place in a m x n board.

2. For each cell in the board, we define a critical mass. The critical mass is equal to the number of orthogonally adjacent cells. That would be 4 for usual cells, 3 for cells in the edge and 2 for cells in the corner.
3. All cells are initially empty. The Red and the Green player take turns to place "orbs" of their corresponding colors. The Red player can only place an (red) orb in an empty cell or a cell which already contains one or more red orbs. When two or more orbs are placed in the same cell, they stack up.
4. When a cell is loaded with a number of orbs equal to its critical mass, the stack immediately explodes. As a result of the explosion, to each of the orthogonally adjacent cells, an orb is added and the initial cell loses as many orbs as its critical mass. The explosions might result in overloading of an adjacent cell and the chain reaction of explosion continues until every cell is stable.
5. When a red cell explodes and there are green cells around, the green cells are try to red and the other rules of explosions still follow. The same rule is applicable for other colors.
6. The winner is the one who eliminates every other player's orbs.

Download the game from the Play Store available here for visualizing the GUI components and understanding how the game works.

Basic Requirements :
1. Main page must provide:
    a. Option to choose number of players (2 to 8).
    b. Option to choose grid size, (9x6 or 15x10).
    c. Button to start the game.
    d. Resume Button - This button should only be visible on main page when the last game played has not finished and a move has been made. The game will resume from the last saved state.
    e. Settings Button - To open the settings page.
2. Settings page:
    a. It should contain a list of 8 players.
    b. Clicking on a row corresponding to a player should allow to configure the colour of the ball for that player.
    c. Ensure that colour of the balls are different for each player in the game.
3. When the game starts:
    a. A GUI similar to that of the game in playstore.

   b. Option to redo the last step.
   c. A drop down with two options:
      i. Start the game again
      ii. Exit and go to main page
4. Animation:
   a. If more than one ball in a cell, then they must rotate as a unit within the cell.
   b. Transition should be visible when balls are splitting.
5. Save the state of the game at any point of time.
6. Also save the state in abnormal conditions (user presses cross, presses home button (if any), etc.)
7. Generate very detailed documentation for each and every class/method/fields of your project by using the JavaDoc tool. You can get more information on what/how to use JavaDoc in this online tutorial.

Bonus:

Although we have specified the basic requirements, if you are able to come up with some more interesting features then you would be "eligible" for bonus marks. Although this eligibility we will decide based on factors such as how many other group has also come up with same additional functionality.

Project Deliverables -

- Deadline 1 (Due on 11th Oct) - Submit UML class diagram and use case diagram for your project
  - Demo for this component the same day
- Deadline 2 (Due on 25th Oct) - Show static GUI of your project and also some animation components
  - Demo for this component the same day
- Deadline 3 (Due on 12th Nov 23:59pm) - Submit complete project on backpack.
  - Demo in between week 17 and week 18 after your end sem exams are over