CSE 201 Advanced Programming (Monsoon Semester2022)
Course Project
Tank Stars Game

IMPORTANT Instructions:

1. **You must stick to 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 well in advance.**
2. **You MUST have a PRIVATE git repository for your project and every group member should frequently check in their code in this repository.**
3. **If you see any ambiguity or inconsistency in a question, please seek 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 with as per the plagiarism policy of IIITD.

Create an application for the game - **Tank Stars** using either LibGDX or JavaFX. Using LibGDX is encouraged because of the following.

- JavaFX is not used much in real-world projects. On the other hand, LibGDX is a game development framework that provides a lot of support for developing games.
- It might turn out that the UI and other game features might just "look better" when LibGDX is used. If there is a subjective assessment component, there is a chance that the project using LibGDX might end up getting slightly higher points than an equivalent JavaFX project.

We will be implementing the **two-player (1 vs 1) game mode** where players face off on hilly terrain, taking turns firing bombs at one another. Players get a limited fuel ration per turn and are allowed to position themselves for a better shot. They can gain weapon upgrades by random airdrops, and must shoot at one another until one of their tanks is destroyed.

It is important that you play the game and make yourself well-familiar with it before reading this document any further. A youtube video of the game can be found at this link.

Android link: https://play.google.com/store/apps/details?id=com.playgendary.tanks

iOS link: https://apps.apple.com/us/app/tank-stars/id1347123739

***Note:** Kindly make sure that you are using all the OOPs principles taught throughout the course (inheritance, polymorphism, interfaces, etc.), including at least 2 design patterns*

(which would be covered in the coming weeks), along with following best coding practices (naming conventions, access modifiers for class, fields, comments, etc.) and JUnit tests.

Some of the rules for the gameplay are mentioned below. You will get to know all the rules after you play the game and read the rules associated with the game. In your implementation, you should follow all those rules.

Basic features:

1) There are many modes in this game. We are only concerned with the 1 vs. 1 game mode, where tanks face each other and take turns to shoot each other until one of them runs out of health.
2) The player is allowed to select the power and angle of the trajectory and then fire the shot.
3) The effect of the hit (impact on the health) on the tank should be based on how close it has been hit.
4) The affected tank should move in the direction it has been hit. Also, the amount of movement should be decided on how accurate the hit was.
5) Allow players to choose from at least 3 tanks before starting the game. All the types of tanks available can be found here and also can be found by playing the game.
6) There must be a pause menu that allows players to save or resume or exit to the main menu at any point in the game.

Features that are not present in the game but are required to be implemented by you:

1) Note that the game doesn't have the option to save the state of the game at any given point in time. However, your implementation should be able to save the game state and save the following:
    a) Health of the 2 players
    b) Store the exact position of the tanks
    c) Store the orientation of the ground.
2) A player must be able to save as well as load any saved game.
3) The game must allow multiple save/load games.
4) It is not required to show the destruction/mutilation of the ground upon projectile impact.

Basic Requirements:

1) Main page:
    a) New game button
    b) Resume game button
    c) Exit game button

2) Resume game button: It should lead to a screen showing a list of saved games.

3) In-game options
    a) (on losing) Restart game, exit to main menu
    b) During gameplay - a pause input, save the state of the game

4) GUI should be designed using LibGDX or JavaFX and should be similar to the gameplay video referred above.

5) Minimum number of different types of tanks to be included should be 3 [*Refer types of tanks available in Tank Stars*].

6) Show the current health at an appropriate position on the screen.

7) Command line output will not be considered part of the game. GUI should be the sole interface for interaction.

8) It is not mandatory to have a fluid-like animation as shown in the game.

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 will be decided based on factors such as how many other groups have also come up with the same additional functionality and how significant that functionality is.

Some possible bonus features

** Note that not all of these have to be implemented to get a bonus. You can come up with other features for bonus, but please discuss them with the instructors/TAs beforehand.*

1) Make use of drops between a match to make the game interesting and allow players to take advantage over one another. They need to be placed appropriately so that both the tanks have fair chances of picking up the drops. Add at least 1 type of special attack that is available from the air drops (e.g. Air strike, Big one, etc.)
2) Add destruction/mutilation to the ground upon projectile hitting, as in the actual game with animations.
3) Come up with interesting modes of your own, for example, add Player vs. Computer mode to the game, where the computer's actions have to be automated with some amount of intelligence and not completely randomized.

Project Deliverables (20%):

Only one member from each group needs to upload the files.

1. **Deadline 1** (Due on 22nd November, 11:59 PM)

   Submit detailed ***UML class diagrams (3%)*** and ***UML use case diagrams (2%)*** for your project along with ***static GUI (5%)*** with ***some basic animation components***. The static GUI should be made in accordance with the UML diagrams you will submit.

   The **UML class diagram [3%]** should include all the basic classes and interfaces. They should have the necessary attributes and methods. Class relationships should be correctly identified with class multiplicities. Make sure that the game is serializable.

   The **UML use case diagram [2%]** should include use cases where the user interacts with

the game with meaningful demonstrations of relationships and multiplicities between them.

The **static GUI component [5%]:**
Static GUI means you should show the events but need not implement the corresponding event handlers. This deadline will simply test "how your project looks" to support all the project requirements. For example, for a project that implements the email login page, the static GUI components are a) login page with the name of email service, b) text box for username, c) text box for password and d) submit button. How the submit button will actually work is not part of static GUI components.
For Tank Stars, this may include Home Page, Game Page, In-Game menus with basic elements (like Tank, navigation buttons, etc.)

Submit pictures of your diagrams on google classroom. For the static GUI, submit this code on google classroom.

2. **Deadline 2** (Due on 20th December, 12 noon)
   Submit the complete project **(10%)** on google classroom, along with the GitHub repository link (which should be made public after the deadline). Also, record a demo of the game showcasing the functionalities and bonus features (if implemented). The video demo can help us resolve any discrepancy in grading and bonus components that may arrive later. Make sure to follow the OOPs concepts taught in class and JUnit tests, along with at least 2 design patterns.

Project demos will begin **around 3 pm on 20th December 2022**.

No further extensions will be provided since the deadlines have already been designed in a way wherein you get maximum time to deliver your project.