Space Shooter Game

OOP PROJECT

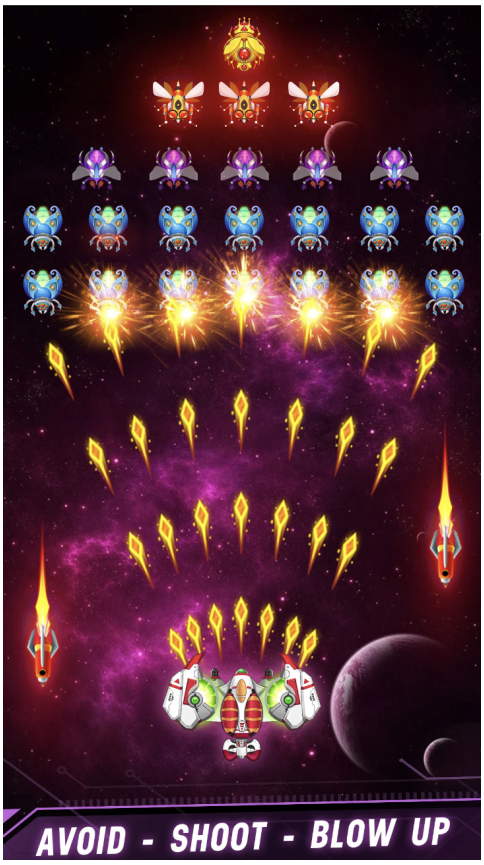
Object Oriented Programming (CS1004) – SPRING 2023

Sher Bano, Bushra Fatima, Amna Hassan

INSTRUCTIONS:

1. **Plagiarism in the course project will result in an F grade in the course. Plagiarism is strongly forbidden and will be very strongly punished. If we find that you have copied from someone else or someone else has copied from you (with or without your knowledge) both of you will be punished. You will be awarded (straight zero in the project — which can eventually result in your failure) and appropriate action as recommended by the Disciplinary Committee (DC can even award a straight F in the subject) will be taken.**
2. This is an individual project and **NOT** a group project. Therefore, each student is required to work on the project individually.
3. Divide and conquer: you are recommended to divide the task in manageable subtasks. We recommend completing the drawing and design (i.e., number of classes and their relationships) phase as quickly as possible and then focus on the intelligence phase.
4. Before writing even one line of code, you must design your final project. This process will require you to break down and outline your program into classes, design your data structure(s), clarify the major functionality of your program, and pseudocode important methods. After designing your program, you will find that writing the program is a much simpler process.
5. No Marks will be given if you do not submit your class diagram and if you do not use the object oriented design principles you have learned during the course.
6. Imagination Powers: Use your imaginative powers to make this as interesting and appealing as you can think of. An excellent solution can get you bonus marks.
7. Use proper naming convention to name the file containing source code. Failure to submit according to the above format would result in the deduction of 10% marks. E.g., *i22xxxx_project.cpp*, replace i22xxxx with your roll number.
8. Follow the given instructions to the letter, failing to do so will result in a zero.

Space Shooter Game



Space Shooter Storyline:

Our galaxy is attacked by enemies. The enemy invaders beat our squad, they destroyed all of galactica. Space team is waiting for your order. Please command the ship to protect the galaxy and the surrounding asteroids.

Space Shooter Gameplay:

The player of the game controls a spaceship in an enemy filled space field. The goal is to destroy the enemy ships ,which include (i) invaders, (ii) monsters and (iii)dragons, avoiding a collision with any of them. The invaders are further divided into three categories: (i)alpha invader, (ii)beta invader and (iii)gamma invader. The spaceship can fire bullets to destroy the objects in the space field. The spaceship is destroyed in case of a collision with an object (enemy ships) or impact with a bomb fired by the enemy ship. Similarly, the dragon appears after a random interval whereby its position is fixed, however, it could fire in 3 directions (downwards, right-downwards and left-downwards). The direction of the fire would also be decided on the basis of the position of the spaceship i.e. the dragon would fire in the zone where the spaceship

is currently located. On the other hand, the monster's vertical position is fixed however it can move in left and right directions. The monster will throw a beam of lightning on the spaceship after regular intervals.

The player can move the spaceship in any direction (right, left, up, down and diagonally). The spaceship moves in all possible directions. As an advanced move, a player can boost the spaceship into powerup mode, i.e. a spaceship cannot be destroyed and fire a continuous beam of bullets instead of periodic intervals. The galaxy is a wrap-around environment for the spaceship i.e., a spaceship disappears at one corner of the screen and then reappears from the exact opposite corner. At the start of the game, the shape of the enemies in the space field is randomly selected. The number of enemies will be selected on the basis of that shape. Next level is achieved once all enemies in the current field are destroyed. The next level starts with a different shape and an increased number of enemies in the space field. The player is awarded three lives initially. A life is decreased if the spaceship is destroyed. The player gets points for hitting the objects. The number of remaining lives, and the score is displayed on the top of the screen.

Implementation:
You are required to design and implement the Asteroids game using Object Oriented Programming. Remember, this project is the perfect opportunity to impress your instructors with your understanding of the OOP concepts. Apparently, a better design of the concepts/classes in the game gets more marks as marks will be deducted for the incorrect design and implementation of the game.
A starter code is provided for reference of the sfml library in C++.

Submission
Deadline to submit the design of the concepts/classes involved in the game is **27th April, 2023 at 5:00 PM.** You can improve the design afterwards, if needed. The submission should be made on google forms as a PDF document and should contain proper figures and explanation of the design. The deadline to submit the implementation and updated design document is **12th May, 2023 at 11:55 PM.**

Features to be implemented

Spaceship
The player of the game controls a spaceship in an enemy filled space field. The spaceship can move in all possible directions (right, left, up, down and diagonally). The spaceship can only fire

bullets to destroy the objects in the space field. The spaceship can also avail the add ons options which would fall after random intervals.

Add-Ons

1. **Power Up:**
   The power up feature will fall randomly in galactic space. Once the player receives the power up add-on, the spaceship will fire a continuous beam of bullets in all seven directions as shown in figure. It will be time controlled lasting only for 5 seconds. In the power up mode, the spaceship will not be destroyed even if the enemy bombs hit the spaceship.

2. **Fire:**
   Another feature that the spaceship can avail is to catch fire. Once the spaceship avails this add-on, its bullets will change into fire destroying all the spaceships in its way. It will also last for 5 seconds only.
3. **Danger:**
   Another add-on feature is the danger sign. This danger sign cannot be destroyed by bullets or bombs. The spaceship needs to maneuver in a way as to dodge and avoid collision with the danger sign. If the spaceship comes in contact with the danger sign, it will be destroyed and the lives would be decreased. The score to dodge the danger sign is 5.
4. **Lives:**
   The lives add-on would help the spaceship by incrementing the remaining lives by 1.

Lives

There will be a total of three available lives of the spaceship. If an enemy hits a spaceship then one life will be destroyed. If a spaceship is in power up mode then no matter how many bullets hit, it will not be destroyed.

Enemy

There will be multiple types of enemies which are described below.

1. **Invader:**

   Invaders are simple enemies that will drop only one bomb at a time. Each bomb will be dropped after a certain interval and the drop time interval for each invader will be different.

   The invaders are further divided into three categories:

   a. Alpha invader: [Score: 10*level]
      The alpha invader would drop the bomb after 5 secs interval
   b. Beta invader: [Score: 20*level]
      The beta invader would drop the bomb after 3 secs interval
   c. Gamma invader [Score: 30*level]
      The alpha invader would drop the bomb after 2 secs interval

2. **Monster:**

   When the monster appears on screen then the rest of enemies will disappear. Its position is fixed however it can move in left and right directions. While moving left to right, the monster will throw a beam of continuous lightning on the spaceship and then wait 2 secs to fire another continuous beam of lightning. The score to dodge the monster is 40.

3. **Dragon:**

   When dragons appear on screen then the rest of enemies will disappear. It should remain on screen only for five seconds after a random interval. The position of the dragon is fixed, however, it could fire in 3 directions (downwards, right-downwards and left-downwards). The direction of the fire would be decided on the basis of the position of the spaceship i.e. the dragon would fire in the zone where the spaceship is currently located. It will drop continuous fire. If the spaceship comes in contact with the fire, it will be destroyed. The score to dodge the monster is 50.

Levels:

There are a total of 3 levels. Each level has a total of 3 phases with 3 different shapes of enemies that need to be destroyed. Moreover, as the user proceeds to the next levels the speed of the enemies and their bombs will increase accordingly. The user will proceed to the next level when all the three shapes in the particular level are destroyed.

1. **Level 1:**

   Level 1 includes the preliminary shapes such as rectangle, triangle and cross sign.

2. **Level 2:**

   Level 2 includes both the advanced shapes such as circle, diamond and heart.

3. **Level 3:**

   Level 3 includes the combination of both the preliminary and advanced shapes but these shapes would be filled.

Screens
Your game should have following screens:

- Game Menu screen
- Instructions screen
- Main Screen for GamePlay
- Pause Screen/Functionality
- High Score and Players Names Display
- End screen

File handling

You are required to store the scores of all the players that have played the game using file handling. You need to store the name, the badge and the highest score of a particular player in a file in a sorted format (descending order). The top three players would receive badges according to their position. The badge of the top three players of the game needs to be displayed while they are playing the game.

Bonus

1. As a bonus task, you are required to do an animation on the next level. Two enemies coming from two opposite directions collide with each other and produce an animation. You need to animate it in such a way that upon collision they destroy, producing a shape of the enemies in the next level.

2. Another bonus task is to store the state of the game at any given instance. Whenever the game is paused and closed, it should be resumed in the same state from where it was paused. The state includes the position of the spaceship, enemies, bullets fired and all other features.

3. Other than the above specified bonus tasks, you can come up with other interesting and creative ideas. The instructors will decide if an implementation is worth bonus marks or not.

References:
Visit the link and download the game to get an idea of the project features:
https://play.google.com/store/apps/details?id=com.game.space.shooter2&hl=en&gl=US