1st Deliverable Submission Date: 29th May 2022 (11:55 PM)
2nd Deliverable Submission Date: 6th June 2022 till 11:55 PM

FINAL PROJECT

Object Oriented Programming (CS1004) – Spring 2022

INSTRUCTIONS

1. **Plagiarism in course project will result in F grade in the course.**
2. This is not a group project and each person will be working on the project individually.
3. The project has two deliverables
    - Deliverable 1: Class Diagram, clearly showing relationships between classes (Due 29th May)
    - Deliverable 2: Complete project (Due 6th June)
4. Combine all your work in one folder and compress it into a zip file. The folder must contain .cpp files and .h files (no binaries, no exe files etc.).
5. Each file that you submit must contain your name and student-id on top of the file in comments
6. Submit the solutions via google classroom. Submissions via email will not be accepted.
7. Use proper naming convention to name the file containing source code.
   E.g. ***P_i21xxxx_project.cpp*** , replace i21xxxx with your roll number and P with your Section.
8. Make sure you submit your project before the submission time. Late submissions won't be accepted even if they are late by just one minute.
9. You can earn bonus marks by implementing extra features in the project.
10. Use good programming practices (well commented and indented code; meaningful variable names, readable code etc.).
11. **Follow the given instructions to the letter, failing to do to so will result in a zero.**

COMBAT version we are going to develop will be slightly different than the original Atari game. You will develop a one player and two player version of the game. For reference you can always watch the videos of the original game on YouTube.

YouTube Link: https://www.youtube.com/watch?v=2LxPEdUZOkE

In this project, you are required to implement the following features:

<u>Game Start Menu:</u> Menu should be displayed as soon as the game begins, it should have following options:

a) High scores – List of top 10 high scores to be displayed in order with the player names. This will be done by maintaining a list of scores in a .txt file.
b) Settings – Choose controls for both players, select type of vehicle (tanks, helicopter, fighter plan)
c) Credits
d) 1 Player or 2 Player – Depending on the selection game will start in one of the modes.
    - Enter Name - Once the game mode is selected, player(s) will enter name(s).
    - Start – Game will start

<u>Game Levels</u>: Game will be played in levels such as given below.

a) Each game level will have obstacles, player(s)' or bot(s)' vehicles.
b) Level generation should be random. Each time a player starts the game at level 1 he/she will get a level with different alignments of obstacles.
c) Display scores at the top.
d) Randomly choose spawn points of player(s) and/or bot(s).
e) Levels have to be playable, e.g. level where player(s) cannot reach the opponent is unplayable.
f) Level should be generated with gaps in the boundaries which should allow player vehicle to leave the level and reappear from the opposite side.

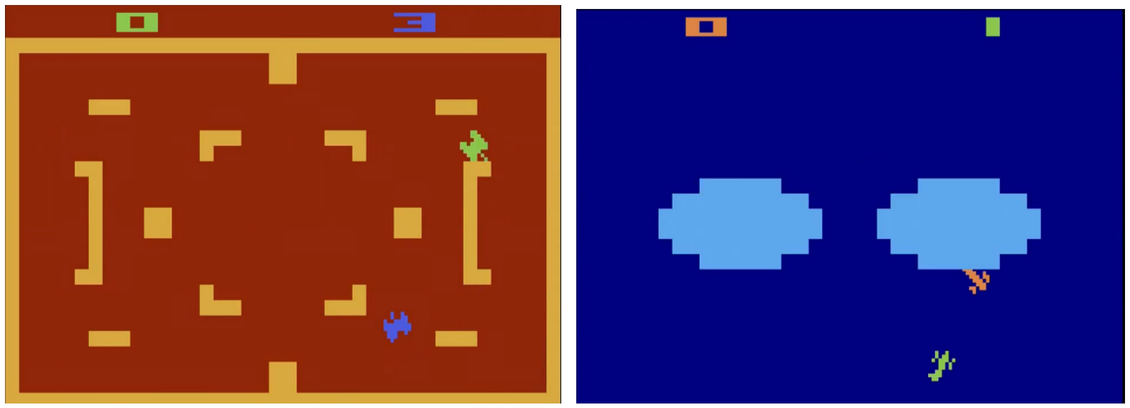

Figure: Sample levels

<u>Game play</u>: Objective of the game is to hit the opponent until the level is clear or game is won.

Player Movement

a) Player can move in all four directions (up, down, right and left using the four arrow keys or other keys configure in settings.
b) Obstacle detection – player cannot go through obstacles generated in the level.

Bot Movement

c) Bots can also move in all four directions
d) Bots movement should be random not fixed
e) Intelligent bot movements – so that they find the shortest path to the player

Canon, their types, and movement

f) Vehicle can shoot canons (of three types – small, medium and large) at their opponents. Although the shape of each canon is circular but they will be different in size, damage and availability. For each level, unlimited small, three (3) medium, and one (1) large canon will be available.
g) Damage of small will be 1 point, medium 2 points and large 4 points.
h) During each level large canon can popup at any point in the level. Player can collect it once it close enough by a key press.
i) The canon can bounce three times against an obstacle before self-destructing.

Scoring Points

j) Each time a player hits another player a point is scored.
k) Each time a player hits the other player front on a point is scored.

1 Player mode – against computer

l) Each level will clear once you have scored 10 points
m) If bots are able to hit the player 10 times in a level computer wins and game ends.
n) You are required to implement three levels, once the player clears all three levels he/she wins.
o) In level 1 there will be one bot, in level 2 there will be two, and in level 3 there will be three bots.
p) Speed of bots and intelligence should increase per level.
q) In 1 player mode you will add power ups which for a certain time will increase your speed, this will help you catch or kill bots easily.
r) Save the progress of player in a separate text file.

2 Player mode – against another player

s) A player will win if he/she hits his opponent 10 times, and game will end.
t) When game ends A screen should popup to ask for re-match.

Bonus Features:

a) Adding sound
b) Adding simple animations when a player is being hit
c) Saving player progress, so that next time player with a same name plays 1 Player mode he restarts from the same level.
d) Intelligent bots
e) Canons fired can also reflect of the walls at angles other than 90

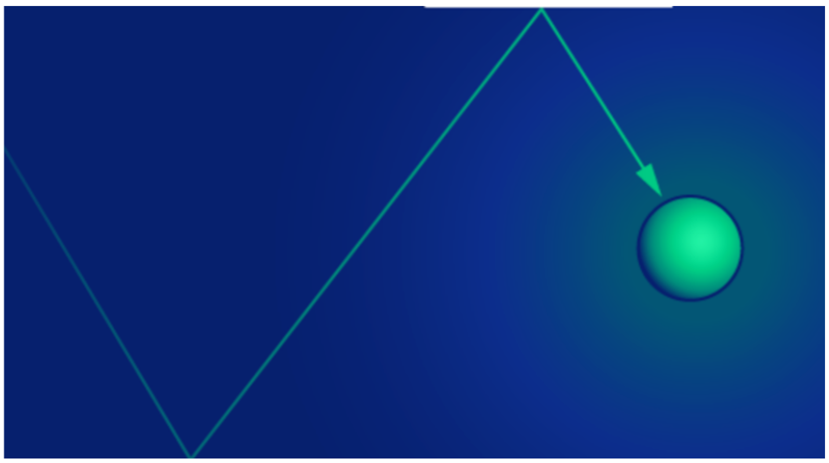

Figure: Canon bouncing off walls

You may think of other interesting (and programmatically challenging) features to implement to get bonus marks.

Starter Code:

We have provided you the starter code in C++ to draw basic shapes which can be used for level generation and generating other game objects. You will change it and add functionality according to the project statement.

To execute the starter code on Ubuntu, you need to do the following:

- Extract the attached zip file.
- Open the terminal and navigate to the path of extracted directory
- Install the required libraries by executing the command below:
    - *bash install-libraries.sh*
- Compile the project by writing the command
    - *make*
- Run the main file

```
./game
```

Instruction to run the project in windows are in the separate file.

Important Note:

You must use OOP concepts of association, inheritance and polymorphism.