Graphics
Assignment 1

Deadline : 11:55 P.M. - 26th January 2019

Problem :

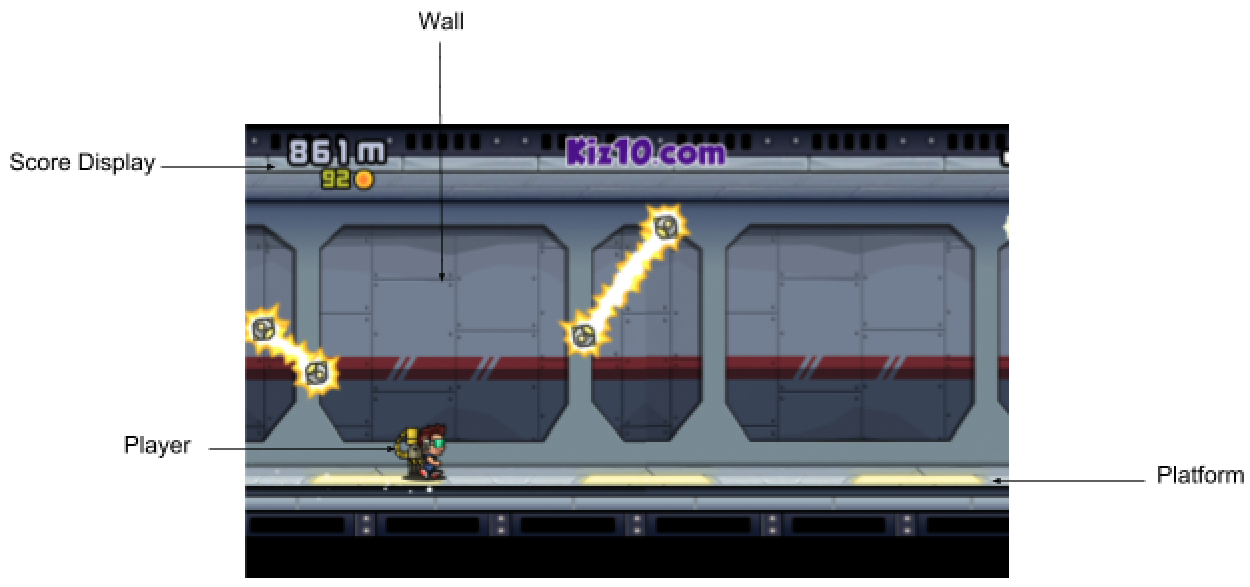
The goal of this assignment is to get yourselves familiarized with OpenGL 3.0 and to brush up your linear algebra and physics basics. This assignment will let you develop your own 2D game with keyboard and mouse controllers.

This will be an arcade game (inspired by Jetpack Joyride) where the player controls the object (for simplicity, it can be a ball too) and move it up, forward and backward while collecting coins. The player should also dodge the obstacles/enemies as they have penalties.

Objective of the game is to maximize score by collecting coins. The player will level up when certain scores are attained and that will cause some additional features to get unlocked increasing the difficulty of the game. You are free to design whole game based on your creativity and define your own rules for points and stages.

In the following sections, the minimum requirements are mentioned. Use your imagination and enhance the game as per your liking. It is your game.
You should provide a single page quick start guide to those who want to play the game (aka TAs), describing the additional controls (basic controls should be as described below), and additional features. Keep track of the scores and levels. They can be displayed on the screen too for bonus marks.

World and Objects :



1. The world consists of a wall,score display,platform and the player.The player will have a Jetpack attached to him with which he can move up. Note : we do not expect the player to look same as the player in the picture, make use of primitives to construct your player in the best creative way possible.
2. Coins of different colours (each corresponding to certain points) should appear above the ground at various heights (random).
3. Magnets,should appear and disappear randomly which will cause the player's motion to be influenced.
4. Special flying objects(moving in a projectile) at least 2, should appear randomly and should give player certain bonus(Like speed ups, more coins,etc) .You are free to use or own creativity and imagination.
5. Circular ring: A semi circular horizontal ring should appear after a certain distance, if the player goes inside this ring he should follow the semi circular path and will be protected from the enemies.
6. The game should have certain enemies(on collision with them,the player should lose points,you are free to decide this on your own):
    a. Enemy 1(**Fire Lines)**:Two circular or small cylindrical objects should appear and a "Fire" like line should be created between them(See the above figure).The Lines can be at any angle with the platform.
    b. Enemy 2(**Fire Beams):** Two pairs of circular or small cylindrical objects should appear and a "Fire" like beam should be created between each pair.The beams should be parallel to platform and can move along Y axis.
    c. Enemy 3(**Boomerangs):** A boomerang should appear randomly and move somewhat in a horizontal 'U' direction or a flattened 'C' direction ,i.e. the boomerang should start ahead of the player, go behind it and then come back and cross the player.If the player comes in contact with the boomerang ,a penalty should be incurred.
    d. Enemy 4(**Viserion**) (**BONUS):** A flying dragon should appear randomly ,adjust its position according to player(movement along Y axis) and should throw Ice balls (Need not look like an exact dragon :p , but certainly better representations would fetch you extra bonus marks)
7. There should be a provision to extinguish fire by throwing water balloons on a key press.
8. Proper collision detection is **more important than the appearance of objects**.You would not want good looking objects, bouncing off without touching each other.

Gameplay:

1. The player can move in any direction horizontally with a fixed velocity according to the key press.
2. The player can use the jet pack and move vertically upwards.
3. There should be an option of Zooming in/out using mouse scroll.
4. In some scenarios(mentioned in Controls section) there should be automatic panning (like in mario).

Physics :

You should incorporate the basic laws of motion (Newton's Laws) into the behaviour/ motion of the objects. The minimum set of factors you need to consider are:

1. Presence of Acceleration due to gravity everywhere.
2. Proper projectile nature of movement of objects according to their vertical and horizontal components of velocities.
3. Proper movement of boomerang.In a horizontal 'U' direction
4. Magnet: The player's path needs to be influenced by the magnet. Assume magnet causes a constant attractive force in its direction.

Controls :

1) For the player to move up use the space key .
   If he reaches the ceiling then he must move in that height unless the space key is released.
   After the release of the space key the player will influence gravitational force and must come down until he reaches the platform . Once the player reaches he must run on the platform.
2) To zoom in/out use the mouse scroll wheel .
3) The player can also go left/right (forward & backward for player) using left/right arrow keys if he reaches the extremes of the screen (leftmost & rightmost) then the panning should be done automatically so that the player is in screen .
4) Any key of your choice for throwing water balloons.

Bonus :

1. Enemy 4,Better representation of the Player.
2. Give special power up to the player,which will create a shield/sword in front of player ,which can cut past enemies and objects.
3. Your Powerful Imagination and Thinking Hat :) !

Useful Hints :

1. When you create objects in the world, make them as objects in the program (class,struct) with all the parameters needed for drawing, animating, motion, sound etc. built into the object along with a method to draw themselves. These parameters would include, but not limited to position, orientation, color, state, etc. This way, you will be able to replicate an object easily and enhance them later on.
2. The main control logic (or game engine) will look at the current state (time, collisions,etc.) and update each element in the world. Create a collision detection function that checks for collision between any two objects. This may be used by the game engine to find any impact and take corresponding action.
3. Start with a simple world and complete the game. You may enhance the objects and motion, once you are done with your V1.0. While creating the objects and the game engine, think of how to make each parameter that you set to be flexible.

Submission :

You submissions should include your source code, a makefile and a compiled executable. You need to include a readme file that describes any additional information that is needed in compiling/executing you code. Do not use any non-standard libraries. In addition to these, include a file named help.txt or help.pdf (no word or other proprietary formats) in the submission that gives a one page description of the game and how to play it. Details of how to submit and any modification to the above submission details will be posted by the TAs towards the submission deadline.This assignment will take time to complete. Start early. All error scenarios must be gracefully handled (Games crashing during testing will be penalised).

Plagiarism in any form shall not be tolerated (MOSS will be used) and a straight F grade for the course will be given. Not protecting your code and collaborative efforts will also be counted as Plagiarism.

Grading :

You will be graded based on the correctness and efficiency (speed) of the implementation of the elements described above. Grading would take place in several stages.

Version 1.0 : Platform,Wall,Player, Coins,Motion and Basic physics.

Version 2.0 : Enemy 1,Enemy 2,Special Flying objects,projectile motions and jet propulsion.
Version 3.0 : Display Score and Stage,Magnets, Enemy 3,Zooming ,Panning ,Circular Ring
Version 4.0 : Bonus and your own powerful imagination and creativity! :)

Marking Scheme :

- 20 marks : Basic implementation of the world.
- 15 marks : Key controls and smooth movement.
- 15 marks : Zooming Implementation.
- 20 marks : Physics
- 15 marks : Enemies and accompanied physics.
- 15 marks : Gameplay.
- 20 marks : Bonus.