CPSC 314 Project 4 Writeup

Team: Joey Lee, Geneva Qitong Sun, Ajang Bul

# Overview

This is a platform game. The game is consisted of a number of suspended platforms of different heights and different movements. Beginning from the ground, the player jumps from platform to platform. The score of the game is tallied using the highest height achieved by the player. The view of the gameplay is in the third-person perspective of the player. The player can adjust the view direction and zoom. The player has the following types of control over the player: move left, forward, right, backward and jump, as well as camera controls with mouse and arrow keys. The player can “pick” the rainbow balls to makes it disappear and adjust the amount of snow. The game can restarts with the same highest score.

# Features

Particle Systems: A series of rainbow balls are scattered on the scene at the beginning of the game. Every time the player touches a rainbow ball, the ball gets kicked.

Procedural Modeling and Motion: Higher platforms are each shown in response to the highest achieved height of the avatar. The position and movement of these platforms are procedurally generated.

Collision Detection: Every time the player makes a successful jump, the point of contact between the avatar and the platform is a collision and the platform stops the avatar from falling downward. The particles in the game also have collision detection.

Level of Details Control: Player controls the number of snow particles using the “Toggle Details” button.

Animation: Snowflakes are displayed on screen in animation.

# Algorithms and Data Structures:

Our unfamiliarity with JavaScript and tight timeline led us to only use arrays to store our data. A hash-table would have been more ideal for particles since any of them can be removed. However, realistically speaking this would not matter due to our relatively small number of particles in the scene.  
  
The main algorithms/models we used were ray-casting and kinematics/dynamics.

In ray-casting, which extends rays from the center of each object to detect intersection with other objects. If the point of intersection is less than the size from which the ray originated from, then we have intrusion (i.e.: collision). Ray casting is also used to detect a picked object by casting from the point of mouse-click instead.

Kinematics and dynamics is used to model the movement of particles in the scene, including what happens after the collision occurs. Factors taken into consideration include gravity, coefficient of restitution, and coefficient of drag/friction.

# How to Play:

After the game is launched, like any other video game, the WASD keys are implemented and each perform the following:

1. Key ‘W’ moves the ball forward

2. Key ‘S’ moves the ball backward

3. Key ‘D’ moves it to the right and

4. Key ‘A’ moves it to the left.

5. Pressing the spacebar make the ball jumps while double pressing it takes the ball extra height especially when the box you’re trying to jump to is far up. Since the boxes are procedurally generated, you can play the game to infinity! While jumping, you could see how far you’ve gone height wise from the “Current height” icon. Your best height is displayed by “Best Height” icon.

6. In some random cases, when you probably missed the box you’re climbing to or you unexpected roll the balls into space while adjusting the ball in preparation to jump, you can either land on one of the boxes below which by chance happened to have been directory below, or you are going to fall right onto the ground. This time, you can continue playing by climbing or restart it the game.

7. At any time of the game, you can use “Restart Game” icon to restart your game.

8. Toggle Detail spice up the view with fogs with intensity increasing with each click up to the max then back to clear sky once more. Also try toggling the lights.

9. Change your view by dragging anywhere in the scene.

10. What are the balls for? Can you find the cheat code?