Gabriel Ikpaetuk

CS 165

Brother Alvey

February 10th, 2021

06 PREPARE: CHECKPOINT B - MOON LANDER DESIGN

1. Problem Description

- *Briefly describe the what the project is.*
    - The project, moon lander, is a which object is to safely navigate a lunar module to land on a flat portion of the moon's surface. In achieving this the user pilots uses the horizontal thrusters to maneuver the ship left and right, and the vertical thrusters can be fired to slow the ship descent. Lunar Lander was one of many influential early video games released by Atari in 1979.

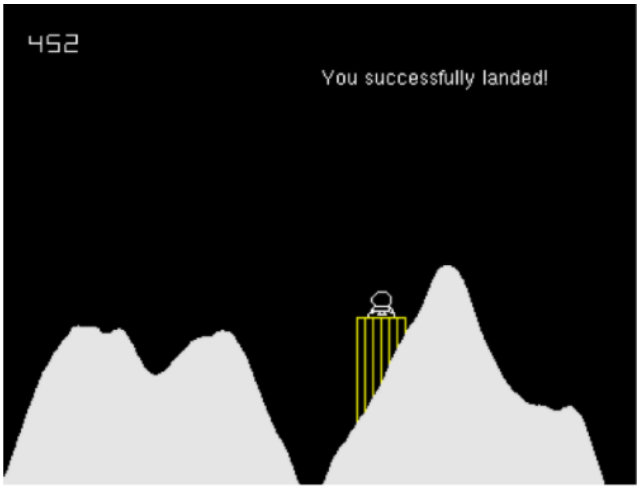
2. Design Overview

- *Describe the major components of your system and how they will interact. This should include a description of the game loop.*
    - The game loop is a set of execution steps that occur over and over again as long as the game is running. The important elements of this code area that it starts up the game, and then continually loops through the following functions:
        - advance – Handles objects movement. In the advance function of our game loop, we handle the passage of time. Things that should happen as time passes, regardless of what buttons are pressed by the user.

- handleInput – Checks for user input and take the corresponding actions. In the handleInput function, we see what keys the user is pressing, and modifies the game objects to respond to those keys.
- Draw – calls draw on every object that should be on the screen. Every time the draw function is executed, we're starting with a blank slate. This means that everything must be drawn on the screen every time we pass through the game loop

3. Interface Design

- *Describe what will show on the screen.*



- *Describe what input the user can provide and what each input command should do.*
    - The user can fire left, right, and bottom thrusters to guide the craft safely to the landing platform before the fuel is exhausted.

4. Algorithms

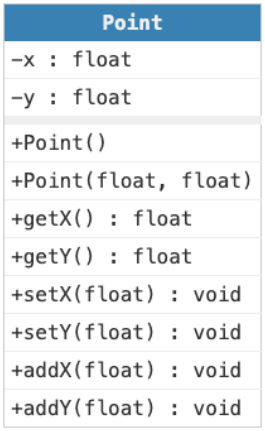
- *For this section you should think about how inertia should work. In pseudocode, define the way the lander's position should be updated.*

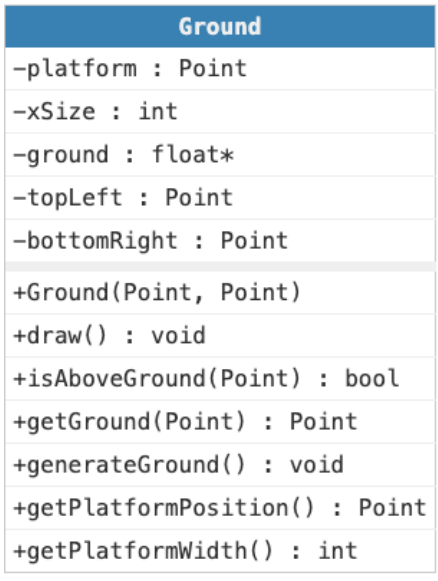
- *Advance()*

  x += speed

5. Data-structures

- *Provide the UML of the classes in your program.*





6. File Format

- *You can list that this is not applicable to this project.*
  - This not applicable to this project.

7. Error Handling

- *Identify anything that could go wrong and how your program should react.*
    - The game classes will be robust to any type of user or file input. Extensive error handling will be built into each class to ensure that clients of the class will use them correctly. Similarly, there will be no way the user can cause the program to malfunction due to incorrectly formed input.