CMPS005J, Fall 17, Section 01
INTRODUCTION TO PROGRAMMING IN JAVA

Navigation

Programs

Program 6

Objective

The objective of this assignment is to get some experience with a larger more complex program invloving multiple functions and some experience working with a class.

Program Specification

For this assignment you need to create a version of the classic 2D video game "Lunar Lander." See http://en.wikipedia.org/wiki/Lunar_Lander_%28video_game%29 for some history of the game. Your program must be built using a modification of the Rocket.pde class found in Canvas/Files/examples/program6.

For full credit, your program must meet the following requirements:

- The rocket must have a limited fuel supply that is consumed at a rate proportional to the thrust. If the fuel supply is exhausted the rocket will continue without power until it crashes, lands, or the occupants die due to lack of power for life support systems at some predetermined number of seconds after the fuel is exhausted.
- A safe landing must require that both the horizontal and vertical velocities at the time of touch down be below a threshold that you specify.
- The landing surface must be roughly a horizontal surface that extends across the width of the display (e.g. it is not ok to have your rocket land on a planet represented as a circle totally - or mostly - within the display).
- There must be a visual indication that clearly indicates that you have detected a safe landing, a crash, or died due to life support failure (e.g. an image of the destroyed rocket on a crash).
- You must not remove the "private" modifier from the instance variables of the Rocket class (such as x, y, xVel, yVel). You should instead add code and/or methods to the Rocket class for anything that requires access to these variables.

Getting Started

- Create a new folder named LunarLander.
- Download the starter code from Canvas/Files/examples/program6 into that folder (LunarLander.pde and Rocket.pde).

- Add code to limit the fuel supply so the thrust goes to zero when the fuel is exhausted.
- Add code to display the amount of fuel remaining.
- Add some basic scenery.
- Add code to detect a landing (don't worry at first about it being safe or not).
- Add code to detect a safe landing.
- Add code to report everyone died if you have run out of fuel and some amount of time has elapsed.
- Add code to display a safe versus a crash landing.
- Work on one of the bonus features (see below under grading).

What To Turn In

This program will NOT be submitted to CrowdGrader. When you and your partner have completed your sketch (program):

- ONE of you should submit a zip file of your sketch folder in Canvas. Make sure that it contains ONLY your sketch. I recommend you move the zip file you create to an empty folder and unzip it to make sure it contains all of your sketch and nothing more.
- BOTH of you should paste a programming log into the text area on the submission page. Use one of the sample logs as a template. Do NOT put the log in the comments section of Canvas.

Grading

A full credit submission must include:

1. Have a limited fuel supply and shut down the rocket when exhausted.
2. Visually indicate a crash when the horizontal velocity is too high on touch down.
3. Visually indicate a crash when the vertical velocity is too high on touch down.
4. Visually indicate "everyone died" when the life support fails.
5. Provide at least a simple scene showing the landing surface.
6. Display the percentage of fuel remaining.
7. Include at least one new method/function for the Rocket class, not part of the starter code.
8. Minimize the use of magic numbers.
9. Include an opening block comment describing your program that includes all authors' names and inline comments describing the major sections of the program.
10. All functions should be no more than 40 lines long and most should be 25 lines long or less.

We will use the following to help us determine where your program belongs. The following is only a guide. The programs will be graded as a whole. Your program will start with 10 points and then we will deduct up to one point for deficiencies in each of the 10 items above. The following list shows what the various scores are intended to reflect:

9-10. Excellent program meeting all requirements.
7-8. Very good program with some minor deficiencies.
5-6. Satisfactory program possibly including one major deficiency.
3-4. Unsatisfactory but showing some understanding of the key concepts for this project.
1-2. Totally deficient but a non-trivial submission.

Bonus points:

To encourage creativity and to provide insurance against small deficiencies in items 1-10 above, a program with the following can receive 0.5-1 point for each bonus feature.

- A more visually "interesting" rocket than the simple triangle provided.
- A particularly creative/complex visual indication of the various end conditions (e.g. in the original video version the astronaut climbed out of the rocket, walked to a McDonald's, ordered a burger and fries, and then flew off again).
- Provide a special successful ending if the rocket lands "near" some surface feature (e.g. a crater or restaurant).

Two other classic 2D video games can also be created using the starter rocket class for this assignment:

- Space Wars - You have two rockets each trying to shoot down the other one.
- Asteroids - You have one rocket trying to blast away asteroids before they smash into your rocket.

This site is maintained by: mcdowell@ucsc.edu
UC Santa Cruz, 1156 High Street, Santa Cruz, CA 95064

Log In