Project 3: Frogger

Due: 11 November 2019

For your third project, you will implement (a version of) the 1980s arcade game Frogger.

Basics of the game

We talked about this a bit in class, but the basic goal of the game is for the player to navigate frogs from the bottom of the screen to one of the "home" locations on the top. Along the way, the frog must cross a road, where it can step only on the road itself and must avoid the moving objects (vehicles), and the river, where it can step only on the moving objects (floating things) and must avoid the water itself.

For our version of the game, you don't need to perfectly match the original, and I'm especially not concerned whether the graphics are identical. For instance, logs could just be solid brown rectangles, and the frog could be a green square. There should, however, be multiple rows of road, with at least two kinds of vehicle and vehicles travelling in both directions; and multiple rows of river, with at least two kinds of floating thing and things floating in both directions.

The player wins the game when they successfully fill the five home locations with frogs (after each one is filled, another frog shows up at the bottom of the screen). The player loses when their frog dies three times. A frog dies when it: is hit by a vehicle; falls in the water; is on a floating thing that carries it off the screen; or jumps into the top row in a spot other than an open home location.

Interface

Running your program should open a window with the game in it; at a minimum, it should have a Pause button (clicking once pauses the game, and clicking again unpauses it), a Start button, and a Quit button.[1] While the game is running, the four arrow keys should control the frog.

[1] Pause and Start may actually be the same physical button (pause/unpause is only needed while the game is running, and start when it isn't), but this is not required.

Design homework

For your initial pass at this, *by Wednesday* you should come to class with some plans for how you'll design this. It should be on paper (so we can talk about it) and should include at least:

- All classes you will eventually implement, and the data they control
- Identify the GUI widgets you will be using for window control
- Ideas for how the various non-gui-widget classes will interact with each other (is-a relationships? has-a? what kinds of behaviours?)

Checkpoint

For the checkpoint, due next Wednesday at 4pm, you should at a minimum pop up a window with the two buttons and an open area where the game will go, and draw a "frog" on that area (again, a green square would be sufficient for that), and have the frog move back and forth in at least one dimension in response to arrow keys.

Final version

The final version should be a playable Frogger(-like) game meeting the requirements laid out above. Your first floating-thing type should undoubtedly be a log, but it's up to you whether your second is turtles or crocodiles (which each introduce a new, different way to kill the frog).

Rubric and timeline

Credit is on the following scale. Underneath the main score for each rubric line is the total number of points if you also get the checkpoint done on time (15 points) and submit appropriate documentation to tell me how/what to run (20 points), and where that number of points falls on my grading scale. In the right-hand column are some strong recommendations on where you should plan to be if you want to finish the project on time.

Most of the rubric items past the 15-point mark are independent of each other, and can be implemented in some other order—this is just the recommended order. If you get stuck on one, try a different one.

Score	Description	Pacing comments
0 (35/F)	Doesnt compile, or compiles but immediately crashes when its run.	
10 (45/D)	Compiles, runs, does no more than specified for the checkpoint.	Recommend finish by Mon 28th; must finish by Wed 30th
15 (50/D+)	Frog moves in all four directions, and at least one moving thing is onscreen, and moving.	Recommend try by Mon 28th
20 (55/C-)	Game doesn't start until Start button is pressed; Quit closes window and quits	
25 (60/C)	Moving things in both directions, that continually move offscreen and repopulate on the other side of the screen	Recommend finish by Wed 30th
30 (65/C+)	Death from being hit by vehicle or jumping in the river	Recommend try by Wed 30th
35 (70/B-)	Frog can ride on top of floating things, and dies from being carried offscreen	Recommend try by Fri 1st
40 (75/B)	Home locations: one frog enters home, new frog starts at bottom of screen	
45 (80/B+)	Win after five in home and lose after three deaths; win/loss halts game with appropriate display (and Start restarts new game)	Recommend finish by Wed 6th

50 (85/A-)	At least two types of floating thing (log plus either turtle or croc); non-log thing is visually distinct and animated (turtles submerge, crocs open/close their mouths)
55 (90/A)	Death from turtle (submerging) or croc (landing on mouth)
60 (95/A)	Pause/unpause
65 (100/A+)	Implement an additional Frogger element, e.g. crocodiles in home, snakes, fly; check with me to be sure your idea will work and be acceptable.

Handing in

For both the checkpoint and the final version (due 11 Nov at 4pm), hand it in as `proj3` using the handin script.