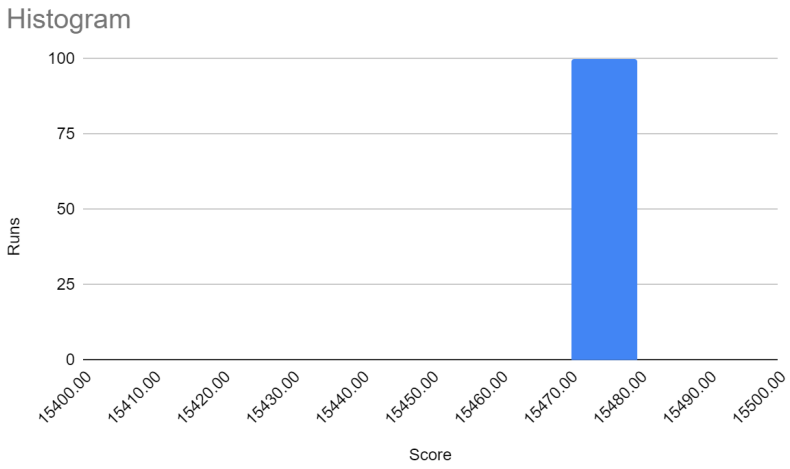
Qbert AI

The approach we took to this project was to first model the whole environment, since we are able to fully observe the environment it made sense to fully model the environment. To do this we first found all the points of the tile Qbert stands on. We then created a data structure to represent each tile holding a pixel cord so we can see what color it is, if it's color is on or off and who is on that tile if anyone. To do this we look at a couple pixels above the cord for the tile pixel (stored for each tile) and see if any of those pixels are the colors of a character. This way each frame we know the entire state of the game(except if Qbert is jumping between two tiles, but that does not need to be known since we can't act till he lands). We then took this objects and put them in a 2d array that represented our environment part of the array was filled with NONE, to indicate empty space where Qbert should not jump, however we did not necessarily need to do this as we ended up with a map of what move Qbert can make from a given tile, so we could have just used a simple 1d array of tile objects. Despite modeling the whole environment we did not use all of the information, we completely ignored the location of the green characters as we do not need to avoid them and we were more focused on other parts of the game.



Our Histogram shows that we only got one score 15475 every time we run the game, this makes sense because the game doesn't introduce any randomness and our character never introduces any randomness so this makes sense. The mean and median are both 15475 and our std deviation is 0. Where we end is on the level where you have to jump on every box twice, something we can not detect and leads us to getting stuck.

submitted by: Rhythm K C, Adam Bock