Bansharee Ireen

COSC10

QuadtreeTest

For QuadtreeTest, I created 6 dots, most of which lie in the 2nd quadrant of my first dot, A. I made the test2() method to create my own tree and run tests there. I also created a hitTest() method to use instead of the provided testFind(); this method tests my findInCircle() method in PointQuadtree by creating a list of (x, y) coordinates of the dots that are hit by the query circle and comparing it to an expected (x, y) coordinate list taken from the parameter. To test the other methods in PointQuadtree, I used the provided testSize() method in QuadtreeTest.

I placed 2 dots, B and C, at the same point (0, 0). I placed two more, D and F, at places between A and B/C, and 1 dot, A.5, above A (they have the same x coordinate). I used testSize() in test2() to check if all 6 were accounted for, especially to see if both B and C were recognized by the code; they were. I then used hitTest () to compare the (x, y) coordinates of dots with an expected (integer) list of x, y coordinates as parameter. I tried checking to see if a search radius that barely touches D would hit D; it did. I then redid this last test with the search radius reduced by 1 pixel in order to make sure it would not hit D—this also worked.

I also used my toString() method in test2() to see whether or not dot A.5 would be recognized as a child despite sharing the same x coordinate value as A, hoping there would not be any issues in identifying the quadrant it was in. There were no issues as A.5 was recognized as a child in the first quadrant and this can be seen in the statement returned by my toString() method.

CollisionGUI

I took screenshots test1, test2, and test3 with 2-3 wandering blobs. In test1, there were 2 such blobs placed close enough to each other that they had the possibility of colliding if they moved closer towards one another; they did not, so they remained black, signaling that they had not collided. In test2, I placed a third wanderer between the 2 to increase the nearly non-existent rate of collision and it soon collided with one of the original blobs and both the newly added blob and its victim turned red. After a while, this third blob collided with the remaining black blob, turning them all red in test3.

In the next 3 screenshots, ‘w’ stands for wanderer and ‘b’ stands for bouncer and some arrows are added to indicate the direction bouncers are moving in. In test4, there are 6 wandering blobs in a sort of pattern while there is 1 bouncer moving towards one of the wanderers. In test5, you can see that the fast bouncer collided with that wandering blob, turning both of them red, and has already bounced off of 3 walls in the time it took me to take and save the last 2 screenshots—it can be seen aiming for a second wanderer in test5. In test6, the bouncing blob successfully hits its second wander.