CURIOUS HUNGRY ROBOT

BY

ARRAH ENAW

AND

NAVID HARIRI

REPORT

Before doing the simulation our hypothesis was that a stack would perform better than a queue for a memory structure. The reasoning behind this hypothesis was that when the robot gets hungry he would go back to where he saw an energy source most recently instead of the energy source he saw at the start of the simulation. The energy location taken from the top of the stack should be relatively closer to where he currently is located, which would allow him to get to an energy location and recharge faster. All of this would ultimately keep him running for longer which would increase the amount he travels in the simulation.

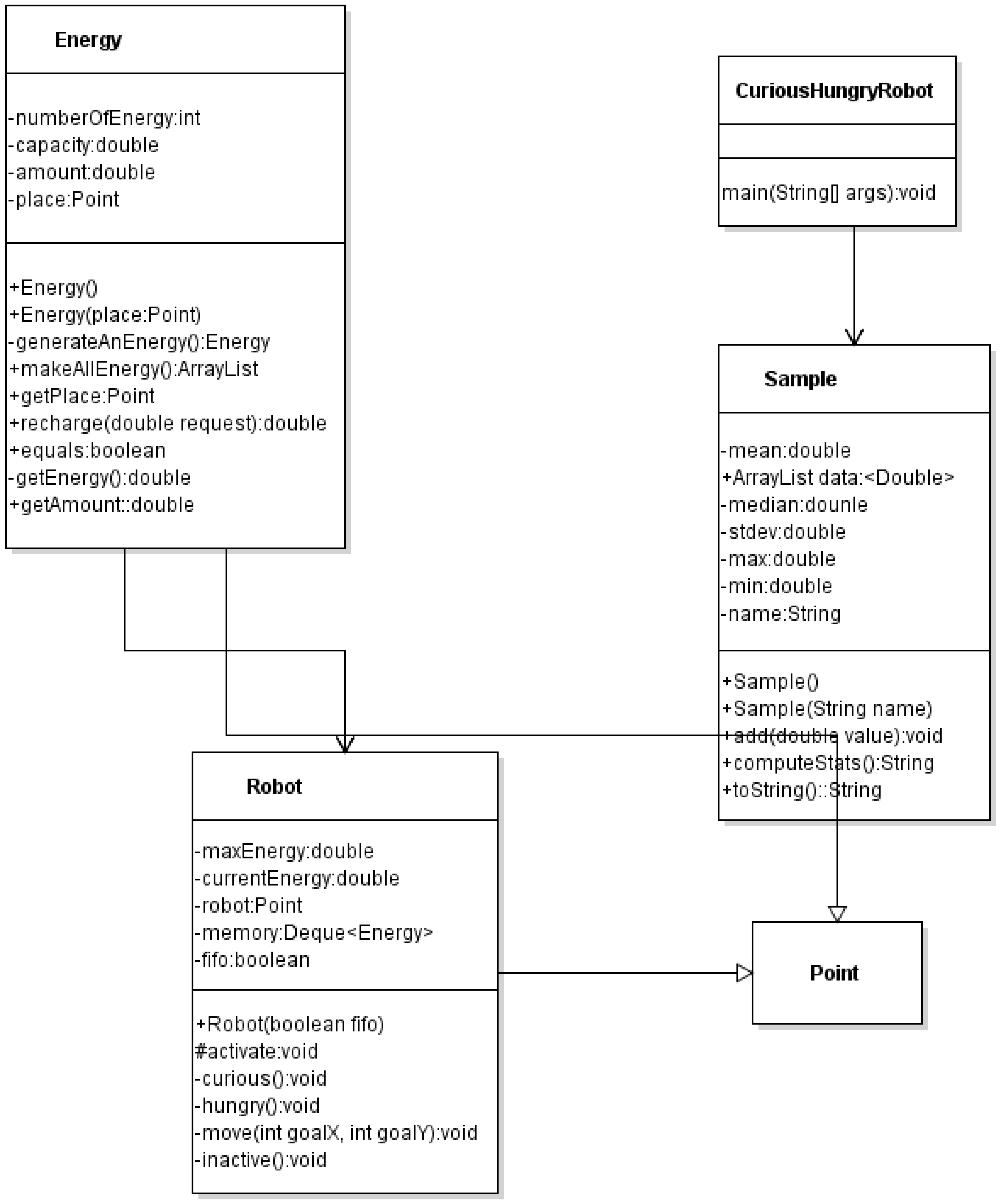
However, the argument for queue being a better memory structure is that as he travels longer to get to his memory goal he is able to explore more. This way he is can find more energy locations, and never be in a situation where he is hungry and does not have any memory goals. In a way he stays alive by learning about his environment. However, this memory structure would be inferior to a stack memory structure because the chances of him running out of energy before he can ever reach his memory goal and becoming inactive is greater. This would ultimately make him inactive sooner and decreasing the amount he is able to travel in a simulation=\].

Upon initially finishing the project the results we obtained showed that both memory structures performed very similarly, with the statistics many times being almost identical. This did not make much sense to us. We thought either we were incorrect with our original reasoning or there was a logical error in the code that we were unaware of and giving us these results. We decided that the best choice would be to check the code as closely as possible and if we could not find any errors then we would have to accept results and try to find an explanation for why they could behave.

Upon further inspection we realized that we were inserting it according to a stack or a queue, but in both cases we were removing memory goals for a queue and never using it as a stack. After fixing this error the results we got were different and there was a clear difference between the performances of the two memory structures for the robot. The mean and median for the stack memory structure were consistently around 1,000 and with a standard deviation of approximately 640 for several 1,000 trails. The mean for the queue memory structure was around 800 and the median was about 700 with a standard deviation of around 450 for several 1,000 trails.

We decided we could use either the mean or median with the addition of the standard deviation to judge which memory structure allowed him to travel the longest distance. However, we decided to use both for greater accuracy and better reliability. In both cases of the mean and the median the stack memory structure always proved to be better for allowing him to travel a longer distance. This confirmed our original hypothesis that with a stack memory structure, the robot is kept alive for longer and is able to travel a larger distance each simulation. Concluding that the stack memory structure is more advantageous for the robot.

DESIGN AND UML DIAGRAM



RESULT

Name	Size	Mean	Median	Standard Deviation	Minimum	Maximum
FIFO (Queue)	1000	884.55	699.59	479.04	49.77	3911.38
LIFO (Stack)	1000	1051.74	944.65	636.45	36.77	5047.21

Evaluation:

FIFO (Queue) Mean: 884.55 + 479.04 = 1,363.59
FIFO (Queue) Median: 699.59 + 479.04 = 1,178.63

LIFO (Stack) Mean: 1051.74 + 636.45 = 1,688.19
LIFO (Stack) Median: 944.65 + 636.45 = 1,581.10

LIFO (Stack) Mean: 1,688.19 > FIFO (Queue) Mean: 1,363.59
LIFO (Stack) Median: 1,581.10 > FIFO (Queue) Median: 1,178.63