COMP3506 Assignment 1 Report

## Michael Ponticello (42878601)

# Question 2(b)

The operation to place the two new variables in an array always runs in constant time. Adding an entry (the array) to the end of a deque also runs in constant time. Therefore, the buy operation runs in O(1) time.

The worst case time complexity of the sell operation is based on the number of times the while loop needs to execute, based on how large q is. If q is a high enough number (the worst case), the while loop will have to run a number of times equal to the number of elements in the deque. All the operations within the while loop are simple comparing, assignment and array operations. These run in O(1) time, thus the worst case time complexity of sell is O(N) where N is the size of the stocks ArrayDeque.

The worst-case space complexity after n buy and m sell operations will be O(n), since the stocks ArrayDeque will have length n if the sell operations didn’t have a large enough quantity to completely remove the oldest element in stocks (and thus all the shares purchased are still present).

# Question 3(b)

The worst-case time complexity of the reward method is O(log n), where n is the number of nodes in the given binary tree. In the worst-case, the loop will run a number of times equal to the height of the tree (if the answers are such that the node with the highest depth is reached). The height h is proportional to the number of nodes (n): h ≥ log2(n + 1) – 1; therefore the worst-case running time is proportional to O(log n).

# Question 4(b)

The worst case time complexity of the printDecisionTree method is O(n2), though in an extremely rare case. The second function which is recursively called usually runs in O(log n) time, but in O(n) time if the tree is structured similarly to the following diagram.

The simple print and comparison operations run in constant time, but the for loop which builds the tab string runs is O(n) time in the worst case, since the number of tabs is based on the depth of the elements (see diagram). The recursive calls essentially perform a pre-order traversal of the tree, meaning each node is visited once (n times). Therefore, printDecisionTree runs in O(n2) time.

# Question 5(b)

The worst-case time complexity of revisedAnswers is O(qn). The function performs a pre-order traversal in a recursive fashion of the given tree’s n nodes, visiting each node once. The recursive function runs in O(q) time – while the simple comparisons, array operations and function calls run in constant time, the cloning of the answers array (which is performed once per node, unless it is an external node with a smaller total reward than the current largest) takes time based on its size. Thus, since it is an O(q) operation running n times, the worst-case time complexity is O(qn). The worst-case space complexity is O(qn), since the function makes up to n copies of an array of size q.