Drew Pulliam – DTP180003

CS3345.0U1

Assignment 3

1. Inserting into an array is slow compared to a linked list because array insertion is O(N), and linked list insertion is 0(1) (obviously much faster).

------------------------------------------------------------------------------------------------------------------------------------------

1. A. the initial for loop runs N times, and inside of it each element is inserted into the front and back of another list. Inserting into the front of an array is O(N), and inserting into the end of an array is O(1). O(1) is ignored and the total Big-O runtime is N\*N = O(N^2)

B. the initial for loop runs N times, and inside of it each element is inserted into the front and back of another linked list. Inserting into the front of a linked list is O(1), and inserting into the end of a linked list is O(N). O(1) is ignored and the total Big-O runtime is N\*N = O(N^2)

------------------------------------------------------------------------------------------------------------------------------------------

1. A. the initial while loop runs N times, and inside of it each element is checked and then possibly removed from the list. Removing from an ArrayList is O(N). The total Big-O runtime is N\*N = O(N^2)

B. the initial while loop runs N times, and inside of it each element is checked and then possibly removed from the list. Removing from a Linked List is O(1). The total Big-O runtime is N\*1 = O(N)

------------------------------------------------------------------------------------------------------------------------------------------

1. A. the initial while loop runs N times, and inside of it the second while loop runs M times. The contents of the while loop is simply checking the current elements and adding to a count variable. This results in a total Big-O runtime of N\*M = O(N\*M)

B. this particular program performs identically on ArrayList and Linked Lists, so the Big-O runtime is also O(N\*M)

------------------------------------------------------------------------------------------------------------------------------------------

1. A. the initial for loop runs N times, and inside of it program runs multiple gets. For an ArrayList, get is O(1). This results in a total Big-O runtime of N\*1 = O(N)

B. the initial for loop runs N times, and inside of it program runs multiple gets. For a Linked List, get is O(N). This results in a total Big-O runtime of N\*(N^3) (worst case is 3 get calls per iteration) = O(N^4)

------------------------------------------------------------------------------------------------------------------------------------------

1. A. Iterating over the entire list take N iterations. Removing from the beginning of an ArrayList takes N iterations. Adding and removing from the stack takes O(1) time. Iterating back over the entire stack takes N iterations, and inserting at the end of the ArrayList takes O(1) time. This results in a total Big-O runtime of N\*N\*N = O(N^3)

B. Iterating over the entire list take N iterations. Removing from the beginning of a Linked List also takes O(1) time. Adding and removing from the stack takes O(1) time. Iterating back over the entire stack takes N iterations, and inserting at the end of the Linked List takes O(1) time. This results in a total Big-O runtime of N\*N = O(N^2)

------------------------------------------------------------------------------------------------------------------------------------------

1. Convert [ (a+b)\*(c+d)-e ] to postfix
2. stack = ( , output =
3. stack = ( , output = a
4. stack = ( + , output = ab
5. stack = , output = ab+
6. stack = ( , output = ab+
7. stack = ( , output = ab+c
8. stack = ( + , output = ab+c
9. stack = ( + , output = ab+cd
10. stack = , output = ab+cd+
11. stack = \* , output = ab+cd+
12. stack = - , output = ab+cd+\*
13. stack = - , output = ab+cd+\*e
14. stack = , output = ab+cd+\*e-

Result = ab+cd+\*e-