**Assignment 3**

**Patrick El-Hage**

**250650822**

1. **Give an O(1)-time algorithm for concatenating two doubly linked lists, each having head and tail pointers, such that the obtained doubly linked list has also head and tail. Explain why your algorithm runs in O(1)-time.**

    To Concatenate Two Doubly Linked Lists:

    You have two Doubly Linked Lists, List 1 and List 2, and you will add List 2 to List 1.

    First, you will find the node that List 2's Head points to (let's call this node H2)

    Next, you will find the node that List 1's tail references (T1), and link the node to H2

    Now that we have linked the tail Node of List 1 to the Head Node of List 2, we must update List 1's Tail reference. We will do this by setting it equal to the node that List 2's tail reference points to.

2. **Give a linear-time recursive algorithm that finds the minimum and maximum values in an array of integers without using any loops.**

   Have a function that takes three parameters, min, max, and a list

   The recursion will break if the list is empty, and then return the min and max variable values.

   We initialize the recursion by initially passing in a list, and arbitrarily passing in min and max values equal to the first element of the array. The recursive algorithm body then will compare the array[0] value (the first value in the array) to the min and max values. If the value is less than *min* then it becomes the new min value to be passed in the next recursion. The same follows for *max*, that is, if the array[0] value is greater than max, then it becomes the new max value. We then shift the array to remove the first element and pass in the array to the next recursion.

3. **Give a recursive algorithm, without using any loops, that checks whether a string is a palin- drome, that is, it is equal to its reverse, e.g., racecar.**

    To recursively check if a string is a palindrome, you will first start with a string.

You will compare the first letter of the string, to the last letter of the string. If they are the same, then we recursively call the same function on the same string, only we splice it so that the passed string to be checked again is one character shorter from each side. If the two characters are not the same, the function should return False.

The recursive function will have an if statement and return true if the String length is less than 1 (meaning that it has checked the entire string recursively to the point where there are no characters left to compare)

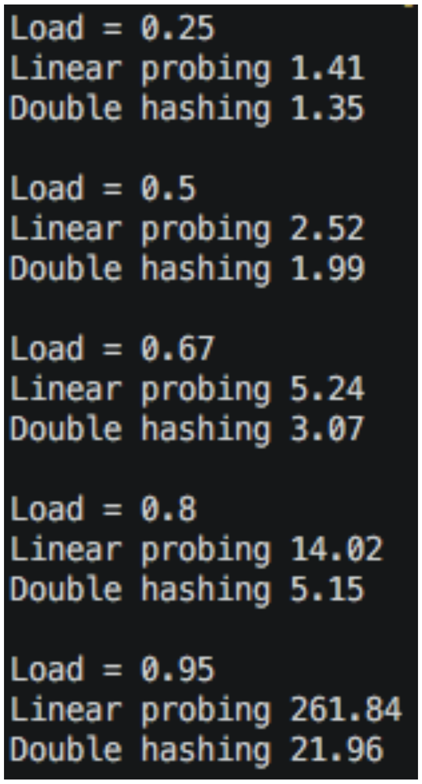
4. **Give a concrete example of a hash table that is not full yet an insertion cannot be performed. You need to specify your example in all details:**

The situation in which a hash table is not full yet can not have a new element inserted is essentially one where the hashing function does not

- Hash table size is not a prime number
- All the table values are 'EMPTY' as opposed to 'UNUSED', therefore the algorithm skips over all the empty values as it searches for an unused value
- Hash function is a linear hash function that does not have a prime modulo, so therefore looks at the same slots over and over again, instead of seeking new slots

**5. ANSWERS**

This is the output of my algorithms



```
Load = 0.25
Linear probing 1.41
Double hashing 1.35

Load = 0.5
Linear probing 2.52
Double hashing 1.99

Load = 0.67
Linear probing 5.24
Double hashing 3.07

Load = 0.8
Linear probing 14.02
Double hashing 5.15

Load = 0.95
Linear probing 261.84
Double hashing 21.96
```