**COSC 3304 – Algorithms Design and Analysis**

**Assignment 6**

**Due: 23:59:00pm, 02/27/2024**

1. Please describe how to insert a new element into an array that is already full (10 points)
   1. Without deletion, this is impossible
   2. By replacing a specified element in the array, an element will be lost but an insertion will take place.
2. Please describe how to remove a node from a doubly linked list (10 points)
   1. A double linked list has the characteristic of an element (not first or last) pointing to both the element before and behind it. The first and last elements point to null before and behind, respectively. This allows traversal of the data in both ways.
   2. To take away a node, one must first shift the pointer (for the .next) of the element prior to it to the element preceding it. Then one must shift the pointer (for the .previous) of the element preceding it to prior of it. Then we can delete the element.
   3. An example of this can be shown as such:
      1. We start off with this:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Element | I | J | K | M |
| Next pointer | J | K | M | Null |
| Previous pointer | Null | I | J | K |

* + 1. We want to delete Element J. Therefore, we will have to do the following:
       - * I.next = J.next;

If the next element is unknown, then the actual code would be:

I.next = I.next.next;

* + - * + K.previous = J.previous

If the next element is unknown, then the actual code would be:

K.previous = K.previous.previous;

* + 1. This will then result in the structure as shown:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Element | I | J | K | M |
| Next pointer | K | K | M | Null |
| Previous pointer | Null | I | I | K |

* + 1. We are then allowed to delete the element without losing access to parts of the data structure (shown below)

|  |  |  |  |
| --- | --- | --- | --- |
| Element | I | K | M |
| Next pointer | K | M | Null |
| Previous pointer | Null | I | K |

1. What is the worst time complexity (big O) when inserting an element into an array? Why? (10 points)
   1. The worst time would be inserting an element into the beginning of an array.
   2. Insertion of an element into an array causes the rest of elements to shift. Shifting an element requires the space behind it be open. If the space is open, then the element can move. Otherwise, if the array has available locations, but there are elements behind it, it must wait till the elements behind it move first.
   3. By inserting an element at the beginning of the array, all the elements must first individually shift to make space at the beginning of the array.

1. Let us consider an empty hash table with 10 positions indexed from 0 to 9. Please illustrate the content of the hash table after inserting the elements 69, 18, 49, 89, 93, 0, and 60.
   1. Linear probing (20 points)

A grid of numbers with black text

Description automatically generated

* 1. Quadratic probing (25 points)

A grid with numbers and a number on it

Description automatically generated

* 1. Double hashing with

*h(k, i)=(h(k)+i\*h’(k))%10*, *h(k) = k%10,* and *h’(k)= 7-k%7*

(25 points)

A white grid with numbers and symbols

Description automatically generated