Soomiya Pobutska

Midterm Exam

April 22, 2020

CIS 3130 Section EWQ6

I.

```
 public void separateLists() {
        LinkList l1 = new LinkList();
        LinkList l3 = new LinkList();

        Link prev = null;
        Link current = head; // start at beginning of list
        while(current != null) {

           if(current.iData < 0 || current.iData > 100) {
                if(prev == null) {
                   head = current.next;
                   prev = null;
               } else {
                   prev.next = current.next;
                   prev = current;
               }
               if(current.iData < 0)
                   l1.insertLast(current.iData);
               else
                   l3.insertLast(current.iData);
           } else {
               prev = current;
           }
           current = current.next;
        }
```

I.

A. One of the advantages of implementing a queue as a CLL is time saving when you need to go from the first node to the last node, in this way you don't have to traverse in between nodes, it can be done only in one step.

B. Some of the advantages of implementing a sorted list of numbers as a DLL are a way of quicker node insertion before a given node, remove operation is more efficient if a pointer to the node to be deleted is given, so you can get it directly from the deleting node, and also the DLL can traverse in both directions forward and backward.

C. The Queue will be the data structure used in this case because it has a FIFO principle, which is first in first out, so the new patients will be added to the back and old patients will be taken out from the front out of the queue.

D. The Stack will be used in this case because of its LIFO principle which is last in first out, so people who were recently hired will be removed and people who were hired first will be kept.

E. In this case Trees can be used, backtracking to the earlier position when end is reached means that you can get back to the initial parent position if the child position is null.

F. Advantages Array vs Linked List:

- Array can access any memory directly.
- Binary Search can be used in arrays because we know positions (indexes) of all elements.

Advantages of Linked List vs Array

- Linked list does not have a specified size, it can have anysize. Therefore you can add new nodes as many as you want.

Disadvantages of Array vs Linked List

- The size of the array needs to be specified.
- The size cannot be increased, otherwise ARRAY OVERFLOW EXCEPTION will occur.

Disadvantages of Linked List vs Array

- Nodes do not have their own address
- As all Nodes don't have their particular address, BINARY SEARCH cannot be used.

III.

A.

1) 5 recursive calls

   Ulam(5) -> Ulam(32) -> Ulam(8) -> Ulam(2) -> Ulam(1)

2) 10 recursive calls

   Ulam(7) -> Ulam(38) -> Ulam(19) -> Ulam(74) -> Ulam(37) -> Ulam(128) -> Ulam(32) -> Ulam(8) -> Ulam(2) -> Ulam(1)

3) 3 recursive calls

   Ulam(16) -> Ulam(4) -> Ulam(1)

4) 8 recursive calls

   Ulam(320) -> Ulam(80) -> Ulam(20) -> Ulam(5) -> Ulam(32) -> Ulam(8) -> Ulam(2) -> Ulam(1)

B. Recursive chain or simply Recursion in data structures is a function called by itself, which has been called by itself before, which also has been called by itself and so on. Very famous example of recursion is the Fibonacci Sequence. Recursive chains are important to recognize, because they are hard to keep track of and can be tricky to write and debug.

C. To simulate recursion in a program you need to make a function call itself when certain conditions are fulfilled and return the value to the function that called it when the condition isn't fulfilled. Recursion is used when you need to repeat a code without rewriting it many times using the same conditional statements.

IV.

1.Counting Number of nodes in a binary tree

```
    int numberOfNodes(Node root) {
          static int count = 1;
            if (root.left != null) {
              count += numberOfNodes(root.left);
             }
           if (root.right != null) {
             count += numberOfNodes(root.right);
            }
         return count;
      }
```

2.Counting nodes which have no sons:

```
  int countLeafNodes(Node root) {
         if (root == null)
               return 0;
         if (root.left == null && root.right == null)
               return 1;
         else
                return countLeafNodes(root.left) + countLeafNodes(root.right);
     }
```

V. DONE ON PAPER. DIFFERENT ATTACHMENT

VI. Write a routine, which will accept a pointer to a linked list and return a pointer to the same list reversed. (I do not want you to use stacks in the process. In addition I do not want you to use getnode. I want you to use the current nodes in the list to do the job)

```
Node reverse(Node node)  {
  Node previousNode = null;
  Node currentNode = node;
  Node nextNode = null;
  while (currentNode != null) {
```

```
            nextNode = currentNode.next;
            currentNode.next = previousNode;
            previousNode = currentNode;
            currentNode = nextNode;
        }
    node = previousNode;
    return node;
    }
```