# Rojin And Suresh Assignment 3

## Problem 1: Pseudo-code

```
Abstract Class: Node
Properties:
- data: the data stored in the node
- next: reference to the next node
- prev: reference to the previous node

Abstract Class: Queue
Properties:
- front: reference to the front node
- rear: reference to the rear node

  Method: enqueue(data)
    Create a new node with the given
data        If rear is None (empty queue):
Set front and rear to the new node
    Else:
      Set new node's next to rear
      Set rear's prev to the new node
      Set rear to the new node

  Method: dequeue()
    If front is None (empty queue):
      Return None
    Get the data from the front node
    If front is the same as rear (single element):
      Set front and rear to None
    Else:
      Set front to front's prev
      Set front's next to None
    Return the data

  Method: isEmpty()
    Return true if front is None, else false

  Method: front()
    Return None if front is None, else return front's data
```

Method: peek()

Same as front()

<u>The running time complexity of the</u>

enqueue()

O(1), because Inserting rear of the doubly linked list is Constant.

dequeue()

O(1), because Removing front of the doubly linked list is Constant.

isEmpty()

O(1), because Checking front is None is Constant.

front()/peek()

O(1), because Checking front is None or returning data is Constant.

**Problem 2: Pseudo-code**

```
Abstract Class: Stack
Properties:
    - items: an array to store stack elements

  Method: push(item)
    Add the item to the top of the stack

  Method: pop()
    Remove and return the item from the top of the stack
    If the stack is empty, return None

  Method: isEmpty()
    Return true if the stack is empty, else false

  Method: top()
    Return None if the stack is empty, else return the item from the top of the stack

Function: isExpressionValid(expression)
  Create an instance of Stack
  For each character in the expression:
    If the character is an opening parenthesis '(', '[', or '{':
      Push the character onto the stack
    If the character is a closing parenthesis ')', ']', or '}':
      If the stack is empty, return False
      Pop the top element from the stack
      If the popped element does not match the corresponding opening parenthesis,
return False
  After processing all characters, if the stack is empty, return True; otherwise, return
False
```

**<u>Interview Q 3: Pseudo-code</u>**

```
Abstract Class: Stack
  Properties:
    - items: an array to store stack elements
  Method: push(item)
    Add the item to the top of the stack
  Method: pop()
    Remove and return the item from the top of the stack
    If the stack is empty, return None
  Method: isEmpty()
    Return true if the stack is empty, else false

Class: QueueUsingTwoStacks
  Properties:
    - stack1: the first stack for enqueue operations
    - stack2: the second stack for dequeue operations

  Method: enqueue(data)
    Push the data onto stack1

  Method: dequeue()
    If stack2 is empty:
      While stack1 is not empty:
        Pop from stack1 and push onto stack2
    Pop from stack2 and return the popped item
    If both stacks are empty, return None
  Method: isEmpty()
    Return true if both stack1 and stack2 are empty, else false
  Method: front()
    If stack2 is not empty:
      Return the top element of stack2
    If stack2 is empty but stack1 is not empty:
      While stack1 is not empty:
        Pop from stack1 and push onto stack2
      Return the top element of stack2
    If both stacks are empty, return None

  Enqueue operation time complexity: O(1)
  Dequeue operation time complexity: O(1)
```