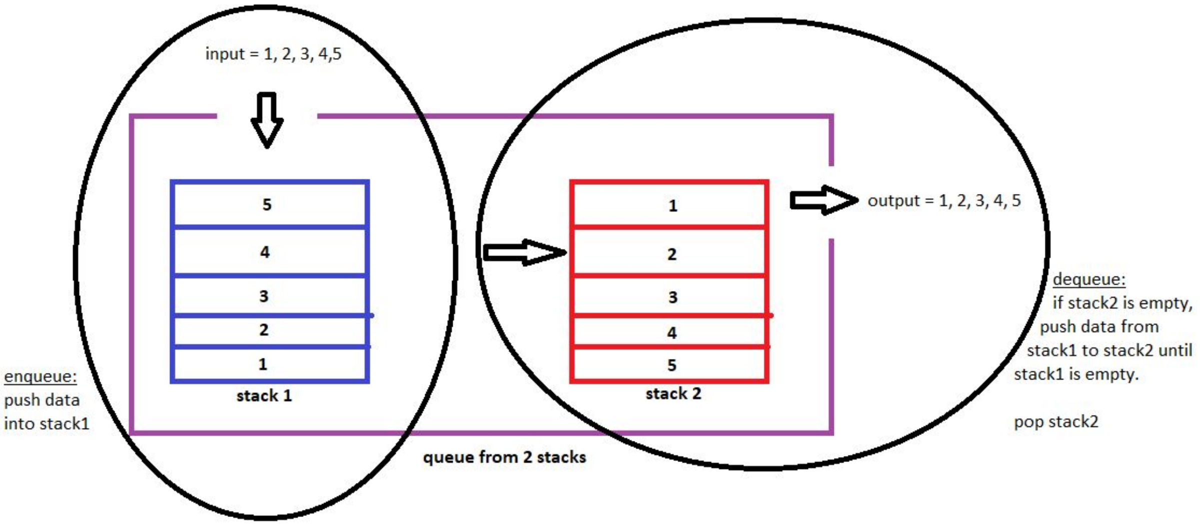
# Assignment 2

*Linus Å & Hjalmar*
*2019-03-05*

## Stacks and Queues

### 1. Queue out of 2 stacks



Psuedo code:

```
var stack1;
var stack2;

enqueue(var x)
        stack1.push(x)

dequeue()
        if (stack2.isEmpy)
                while (!stack1.isEmpty)
                        stack2.push(stack1.pop)
                endWhile
        endIf
        return stack2.pop

isEmpty()
        return stack1.isEmpty && stack2.isEmpty

size()
        return stack1 + stack2
```

Complexity for enqueue: **O(1)**
Complexity for dequeue: **O(n)** *best case: O(1)*

### 2. RPN calculator

see RPN.java file.

## Linked Lists

### 3. get

see SinglyLinkedList.java file.
Complexity for get: **O(n)** *best case: O(1)*

### 4. insertAt

see SinglyLinkedList.java file.
Complexity for insertAt: **O(n)** *best case: O(1)*

### 5. removeAt

see SinglyLinkedList.java file.
Complexity for removeAt: **O(n)** *best case: O(1)*

### 6. reverse

see SinglyLinkedList.java file.
Complexity for removeAt: **O(n)** *best case: O(1)*

## Trees

### 7. nthBFS

see Tree.java file.

### 8. DFSToString

seeTree.java file.
Complexity forDFSToString: **O(n)** *best case: O(1)*

### 7. nthBFS

see Tree.java file.