Q.1:

Answer to Question 1:

```
def permutations(n):

    if n == 1:
        return [[1]]

    previous_permutations = permutations(n-1):
    new_permutations = []

    for perm in prev_permutations:
        for in range(len(perm)+1):
            new_permutations.append(perm[:i] + [n] + perm[i:])

    return new-permutations
```

Justification to Question 1:

The function works because it recursively generates permutations for $n-1$, and THEN inserts n at every possible position in each of those permutations. This guarantees all arrangements of $[1, \ldots, n]$ are produced.

Q.2:

```python
class StackUsingQueue:

    def __init__(self):
        self.queue = Queue()

    def push(self, element):
        self.queue.enqueue(element)

        for i in range(self.size() - 1):
            self.queue.enqueue(self.queue.pop())

    def pop(self):
        return self.queue.pop()

    def top(self):
        return self.queue.top()

    def is_empty(self):
        return self.queue.is_empty()

    def size(self):
        return len(self.queue.queue)
```