# Day 18: Queues & Stacks!

Welcome to Day 18! Review this queues and stacks video and the Java documentation for implementing stacks and queues, or just jump right into the problem.

A *palindrome* is a "word, phrase, number, or other sequence of characters which reads the same backwards and forwards." Can you determine if a given string, $s$, is a palindrome?

To solve this challenge, we must first take each character in $s$, *enqueue* it in a *queue*, and also *push* it onto a *stack*. Once that's done, we must *dequeue* the first character from the *queue* and *pop* the top character off the *stack*, then compare the two characters to see if they are the same; as long as the characters match, we continue dequeueing, popping, and comparing each character until our containers are empty (a non-match means $s$ isn't a palindrome).

Write the following four functions/methods in class *Palindrome*:

- *void pushCharacter(char ch)*: Pushes a character onto a stack.
- *void enqueueCharacter(char ch)*: Enqueues a character in a queue.
- *char popCharacter()*: Pops and returns the top character.
- *char dequeueCharacter()*: Dequeues and returns the first character.

Code handling Input/Output and determining if $s$ is palindrome is provided in the editor.

**Input Format**

A single line containing string $s$.

*Note:* $s$ will always be lowercase.

**Output Format**

If $s$ is a palindrome, print *"The word, $s$, is a palindrome."*
Otherwise, print *"The word, $s$, is not a palindrome."* without quotes

**Sample Input**

```
racecar
```

**Sample Output**

```
The word, racecar, is a palindrome.
```