Code Challenge

Implement a Queue using two Stacks.

Specifications

- Read all of these instructions carefully. Name things exactly as described.
- Do all your work in a public repository called `data-structures-and-algorithms`, with a well-formatted, detailed top-level README.md.
- Create a new branch in your repo called `queue_with_stacks`.
- Your top-level readme should contain a "Table of Contents" navigation to all of your challenges and implementations so far. (Don't forget to update it!)
- This assignment should be completed within the `challenges` subdirectory of the repository.
- On your branch, create...
    - *C#*: a new .NET Core console project named `QueueWithStacks`. Within your `Program.cs` create a new static method outside of `Main()` following the naming conventions below. Call your newly created method in `Main()` once complete.
    - *JavaScript*: a folder named `queueWithStacks` which contains a file called `queue-with-stacks.js`
    - *Python*: a folder named `queue_with_stacks` which contains a file called `queue_with_stacks.py`
    - *Java*: a folder named `queueWithStacks` which contains a file called `PseudoQueue.java`

- Include any language-specific configuration files required for this challenge to become an individual component, module, library, etc.
    - *NOTE: You can find an example of this configuration for your course in your class lecture repository.*

Feature Tasks

Create a brand new `PseudoQueue` class. Do not use an existing Queue. Instead, this PseudoQueue class will implement our standard queue interface (the two methods listed below), but will internally only utilize 2 `Stack` objects. Ensure that you create your class with the following methods:

- `enqueue(value)` which inserts `value` into the PseudoQueue, using a *first-in, first-out* approach.

- `dequeue()` which extracts a value from the PseudoQueue, using a *first-in, first-out* approach.

The `Stack` instances have only `push`, `pop`, and `peek` methods. You should use your own Stack implementation. Instantiate these Stack objects in your PseudoQueue constructor.

Example

`enqueue(value)`

Input	Args	Output
`[10]->[15]->[20]`	`5`	`[5]->[10]->[15]->[20]`
	`5`	`[5]`

`dequeue()`

Input	Output	Internal State
`[5]->[10]->[15]->[20]`	`20`	`[5]->[10]->[15])`
`[5]->[10]->[15]`	`15`	`[5]->[10]`

Requirements

Ensure your complete solution follows the standard requirements.

1. Write unit tests
2. Follow the template for a well-formatted README
3. Submit the assignment following these instructions