## PROJECT 85

- ✓ ***STACK ANSWERS BOTH FOR CHALLENGES AND REFLECTIONS***

### 1. *CHALLENGE ANSWER*

**Reverse "DATAQUEUE" using stack.**

***To reverse the string "DATAQUEUE" using a stack, we push each character onto the stack and then pop them off to build the reversed string. A stack's LIFO property ensures that the last character pushed is the first popped, naturally reversing the order.***

***Algorithmic Sequence:***

*1.Initialize: Create an empty stack and a string variable for the result.*

*2.Push Characters: Iterate through each character in "DATAQUEUE" and push it onto the stack.*

*3.Pop Characters: Pop each character from the stack and append it to the result string.*

*4.Return Result: The result string is the reversed from "**DATAQUEUE" to "EUEUQATAD"***

- ***ALGORITHM CODES AND IT'S EXPLANATION***

## Step 1: Initialize an empty stack (list) and result string.

```
stack = []
result = ""
```

## Step 2: Manually push each character of "DATAQUEUE" onto the stack

```
stack.append("D") #Output Stack: ['D']
stack.append("A") # Output Stack: ['D', 'A']
stack.append("T") # Output Stack: ['D', 'A', 'T']
stack.append("A") # Output Stack: ['D', 'A', 'T', 'A']
stack.append("Q") # Output Stack: ['D', 'A', 'T', 'A', 'Q']
stack.append("U") # Output Stack: ['D', 'A', 'T', 'A', 'Q', 'U']
stack.append("E")  # Output Stack: ['D', 'A', 'T', 'A', 'Q', 'U', 'E']
stack.append("U")  # Output Stack: ['D', 'A', 'T', 'A', 'Q', 'U', 'E', 'U']
stack.append("E")  # Output Stack: ['D', 'A', 'T', 'A', 'Q', 'U', 'E', 'U', 'E']
```

## Step 3: Manually pop each character and append to result.

```
result += stack.pop()  # Pop 'E', Output result = "E"
result += stack.pop()  # Pop 'U', Output result = "EU"
```

```
result += stack.pop()  # Pop 'E', Output result = "EUE"
result += stack.pop()  # Pop 'U', Output result = "EUEU"
result += stack.pop()  # Pop 'Q', Output result = "EUEUQ"
result += stack.pop()  # Pop 'A', Output result = "EUEUQA"
result += stack.pop()  # Pop 'T', Output result = "EUEUQAT"
result += stack.pop()  # Pop 'A', Output result = "EUEUQATA"
result += stack.pop()  # Pop 'D', Output = "EUEUQATAD"
```

## Step 4: Print the original and reversed strings.

```
print("Original:", stack)
print("Reversed: ", result)
```

**Output**

**Original: DATAQUEUE**

**Reversed: EUEUQATAD**

## 2. Reflection answer

**The stack is not suitable for long queues, Because of its LIFO structure. In a long queue, such as people waiting for service, fairness requires serving the first person who arrived (FIFO, or First-In-First-Out). A stack, however, prioritizes the most recently added item, meaning the last person to join would be served first. This can lead to unfairness and inefficiency in scenarios like ticketing systems or customer service, where earlier arrivals expect priority. Additionally, stacks are memory-intensive for large datasets, as they require frequent push/pop operations without direct access to earlier items, making them impractical for managing long, ordered sequences.**

**END**