REPORT

Project Title:

Stack and Queue

Objective:

The main objective of this mini project is to implement and demonstrate the use of two essential data structures: Stacks and Queues, in a C++ program

Introduction

The "Item Categorization System" is a C++ program designed to categorize items into three different stacks (Food, Stationery, and Sports) based on user input. This project report outlines the design, implementation, and functionality of the system.

System Design

Functional Requirements

The system is designed to:

- Allow users to input items and categorize them into three categories: Food (F), Stationery (S), or Sports (X).
- Maintain a queue to store the names of items in the order they arrive.
- Maintain three stacks, one for each category, to store items based on their category.
- Display the contents of each stack along with the number of items in each category.
- Handle invalid inputs and provide appropriate feedback to the user.
- Allow the user to exit the program.

System Architecture

The system architecture includes the following components:

- A **queue<string>** (**itemQueue**) to maintain the order of item arrivals.
- Three **stack<string>** objects (**foodStack**, **stationeryStack**, and **sportsStack**) to categorize and store items.
- Input handling and processing using a **do-while** loop.

- A function (**displayStack**) to display the contents of a stack in the order they were added.
- Error handling for invalid inputs.

Implementation

Code Structure

The system is implemented in C++ and structured as follows:

1. The program includes necessary header files for input/output operations (**<iostream>**), queues (**<queue>**), stacks (**<stack>**), and string manipulation (**<string>**).
2. The program defines a template function **displayStack** to display the contents of a stack. This function can handle any data type and is utilized to print the contents of different stacks.
3. The **main** function initializes the data structures and enters a **do-while** loop to manage user input.
4. Within the loop:
    - The user is prompted to enter an item type (**F** for Food, **S** for Stationery, **X** for Sports, or **Q** to quit).
    - If the input is a valid item type (**F**, **S**, or **X**), the user is prompted to enter the name of the item, and this name is pushed onto the corresponding stack and also added to the **itemQueue**.
    - If the input is not a valid item type (neither **F**, **S**, nor **X**), an error message is displayed.
5. When the user chooses to quit (**Q**), the program displays the contents of each stack, along with the number of items in each category.
6. The **displayStack** function is used to print the contents of each stack. This function pops elements from the stack and displays them in the order they were pushed.
7. The program terminates after displaying the items in each category.

Conclusion

The "Item Categorization System" demonstrates the use of stacks and queues to categorize and manage items based on their type. It allows for the retrieval and display of items in their respective categories and provides an interactive interface for users to input and categorize items. This project serves as a simple yet effective example of how these data structures can be used for practical applications.