# DATASTRUCTURES AND ALGORITHMS ASSIGNMENT

## 23/05754

## Assignment 1: Exploring Stacks and Queues

1. **Explain the Last-In-First-Out (LIFO) principle in the context of stacks. How does this principle differentiate stacks from other data structures? (5 Marks)**

The LIFO principle dictates that the ultimate detail added to a stack is the number One to be eliminated. This way that factors are added and removed from the top of the stack, following a "ultimate in, first out" order.

**How Stacks Differ from Other Data Structures**

Arrays - While arrays can be used to place into effect stacks, they will have overall performance obstacles for massive stacks because of the want to shift factors even as which include or putting off elements from the top.

Queues - Unlike stacks, queues have a look at a First-In-First-Out (FIFO) concept, where the number one detail delivered is the number one to be eliminated.

**Stack's Characteristics**

Push - Adds a detail to the top of the stack.

Pop - Removes the top element from the stack.

Peek - Returns the top element without disposing it.

LIFO principle and its specific characteristics, you could efficiently use stacks in programming programs to:

Function call stacks - To manage feature calls and their move returned values.

Expression assessment - To examine expressions and calculate results.

2. **Describe the push, pop, and peek operations of a stack. Provide a real-world example where a stack would be an appropriate data structure to use. (5 Marks)**

Push - Adds an element to the top of the stack.

Pop - Removes the top detail from the stack and returns its value. If the stack is empty, it returns an errors or null.

Peek - Returns the price of the top element without casting off it. If the stack is empty, it returns a mistakes or null.

**Real-life example**

Stack can be used in systems where memory is allocated in a static way to manage memory allocation or track which memory blocks are in use. It can also be used to keep track of function calls in a program and ensure that the program returns to the correct location when a function is complete. For example, the Java Virtual Machine (JVM) uses this concept in keeping track of the state of a running Java program.

3. **Describe the enqueue, dequeue, and front operations of a queue. Provide a real-world example where a queue would be an appropriate data structure to use. (5Marks)**

Enqueue - Adds a detail to the rear of the queue.

Dequeue - Removes the front element from the queue and returns its value. If the queue is empty, it returns an errors or null.

Front - Returns the rate of the front element without doing away with it. If the queue is empty, it returns an errors or null.

**Real-life example**

Say you've got some of files to be discovered straight away. Your OS locations all of these scientific doctors in a queue and sends them to the printer. The printer takes and prints every record inside the order the clinical docs are put inside the queue, ie, First In, First Out.

In the situation in which there are a couple of customers or a networked computer gadget, you possibly percent a printer with unique customers. When you request to print a record, your request is introduced to the print queue. When your request reaches the front of the print queue, your record is printed. This guarantees that great one individual at a time has get proper of entry to the printer and that this get proper of access to is given on a primary-come, first-served basis.

**For the above example you can visit the following:**

https://stackoverflow.com/questions/2392824/what-are-practical-applications-of-queues