BIT Data structure Exercise

A <u>stack</u>

A stack is a fundamental data structure that follows the Last In, First Out (LIFO) principle. In simpler terms, the last element added to the stack is the first one to be removed. Think of it like a stack of plates, we can only take the top plate off the stack, and to add a new plate, we place it on top.

A. Basics

Q1:

In the MoMo app, each step you complete (like entering account, amount, confirmation) is pushed on the stack. When you press back, it doesn't skip to the first step — it removes the last entered step first.

This is exactly LIFO (Last In, First Out**).**

Q2:

In UR Canvas, navigation works like a stack of visited pages. Each page you open is pushed. When you press back, the app pops the most recent page and shows the one just before it. Same as pop(), which removes the top item.

B. Application

Q3:

In BK Mobile Banking, each transaction is pushed onto history. If you need to **undo**, the system can simply pop the latest transaction(s) off the stack.

This means errors can be corrected step by step, in reverse order.

Q4:

Balanced forms are like balanced parentheses. Each "open field" is pushed onto a stack. When you finish or close it, it is popped.

If at the end the stack is empty, all fields are matched correctly; if not, some fields are left unmatched.

C. Logical

Q5: (Sequence)

- Push("CBE notes") → [CBE notes]
- Push ("Math revision") → [CBE notes, Math revision]
- Push ("Debate") → [CBE notes, Math revision, Debate]
- Pop () → removes Debate → [CBE notes, Math revision]
- Push ("Group assignment") → [CBE notes, Math revision, Group assignment]

Top of stack = Group assignment (next task).

Q6**:** (Undo)
If a student undoes 3 actions, that means pop() is called 3 times.

- The last 3 actions are removed.
- Only the earlier answers (pushed before those 3) remain.

Remaining = all actions before the last 3.

Q7: RwandAir booking backtracking
In booking forms, each completed step (flight → passenger → payment) is pushed onto a stack. When the passenger presses back, the app pops the last step and shows the previous one. A stack fits because it naturally retraces steps in reverse order (LIFO).

Q8: Reversing “Umwana ni umutware” using a stack
Algorithm:
1. Push each word: `["Umwana", "ni", "umutware"]`
2. Pop words one by one → `"umutware" "ni" "Umwana"`
Because popping gives the last inserted word first, the proverb is reversed.

Q9: DFS in Kigali Public Library
In Depth-First Search (DFS), you go as deep as possible along one shelf/row before backtracking.
A stack remembers where you came from, letting you backtrack step by step.
A queue (FIFO) would spread search broadly (BFS), but a stack is better for deep exploratio

Q10: Stacks in BK Mobile app navigation
When moving through transaction history:
Each opened transaction is pushed onto the stack.
Pressing back pops the last transaction and shows the previous one.
🔎 Suggested feature: Quick Undo Transaction Review Allow users to backtrack multiple transactions step-by-step using stack pops.
That’s a compact set of answers with practical Rwandan examples.

On queue
1. Basics of Queue
Definition: A queue is a linear data structure that follows **FIFO (First In, First Out)
Operations:

Enqueue(x): Add an element `x` at the rear
Dequeue(): Remove an element from the front.
Analogy: Like people standing in line → first person in line is the first served.

Q1: How does a restaurant line show FIFO behavior?
Customers arrive and join at the back. The first customer to arrive is the first to be served, which is exactly FIFO.

Q2: Why is YouTube autoplay like Dequeue?
The next video in the list (front of queue)* is automatically removed (played), then the following one becomes next. This mimics `Dequeue()`.

B. Application

Q3: How is the RRA tax office line a real-life queue?
People wait in a first-come, first-served line. Whoever arrives first pays first.

Q4: How do queues improve customer service in MTN/Airtel centers?
By serving people in order, queues avoid confusion, ensure fairness, and reduce disputes about who should be served next.

C. Logical

Q5:Equity Bank sequence:

Start: `[]`
Enqueue("Alice") → `[Alice]`
Enqueue("Eric") → `[Alice, Eric]`
Enqueue("Chantal") → `[Alice, Eric, Chantal]`
Dequeue() → removes "Alice" → `[Eric, Chantal]`
Enqueue("Jean") → `[Eric, Chantal, Jean]`

Front now: `"Eric"`.

Q6:How do queues ensure fairness in RSSB applications?
Applications are processed in **arrival order**, so no one can skip ahead (FIFO ensures equal treatment).
D. Advanced Thinking

Q7: Different queue types in Rwanda:

Linear queue: Wedding buffet line ,first person eats first.
Circular queue:Buses looping at Nyabugog after reaching the end, they return to start and repeat.

Deque (double-ended queue): Boarding a bus from front or rear people can enter/exit from both ends.

Q8:Restaurant orders modeled by queue:
Orders are enqueued when customers place them, and dequeued when ready, ensuring meals are served in order of request.

Q9: Why emergencies at CHUK hospital = priority queue?
Unlike normal FIFO, emergencies are given higher priority and jump ahead of others. That's priority queue behavior.

Q10. Moto/e-bike app matching system:
Riders (drivers) wait in a queue. When a student requests, the front rider is matched first. This ensures fairness so drivers are served in order of waiting.

References

Website: [GeeksforGeeks Queue Data Structure] (https://www.geeksforgeeks.org/queue-data-structure/)