NAMES: ISHIMWE Philbert

REG NO:224016016

MODULE: data structure and algorithm

Part I – STACK

Q1. How does this show the LIFO nature of stacks?

In the MoMo app, when you go back, the last step you entered (like amount or PIN) is removed first. This is exactly LIFO: *Last In, First Out*.

Q2. Why is this action similar to popping from a stack?

Pressing "back" in Canvas undoes the last page or module you opened. Just like pop(), it removes the *topmost/most recent* action.

Q3. How could a stack enable the undo function when correcting mistakes?

Each action (like typing or a transaction) is pushed onto a stack. When undo is pressed, the system pops the last action, restoring the state before the mistake.

Q4. How can stacks ensure forms are correctly balanced?

Each opening bracket (or field) is pushed onto the stack. When a closing bracket appears, the stack checks if it matches the top. If all match correctly, the form is valid. If not, it's flagged as unbalanced.

Q5. Which task is next (top of stack)?

Steps:

- 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 = Group assignment

Q6. Which answers remain in the stack after undoing?

If a student undoes 3 recent actions, the last three pushed answers are popped. The earlier ones remain in the stack.

Q7. How does a stack enable this retracing process?

In RwandAir booking, each new step is pushed. Going back pops the latest, showing the previous step. This allows step-by-step retracing.

Q8. Show how a stack algorithm reverses the proverb “Umwana ni umutware.”

- Push “Umwana” → Push “ni” → Push “umutware”
- Pop → “umutware”
- Pop → “ni”
- Pop → “Umwana”

Reversed = **“umutware ni Umwana”**

Q9. Why does a stack suit this case better than a queue?

DFS goes deep before backtracking. A stack naturally supports this: last path explored is the first to backtrack. A queue (FIFO) would explore breadth-first, which is not suitable for DFS.

Q10. Suggest a feature using stacks for transaction navigation.

A “Back to last transaction” feature: each viewed transaction is pushed, and popping lets users retrace transactions in reverse order.

Part II – QUEUE

Q1. How does this show FIFO behavior?

In a restaurant, the first customer to arrive is the first to be served. That is **First In, First Out**.

Q2. Why is this like a dequeue operation?

In YouTube playlists, the first video in line plays, then is removed, and the next one moves forward—just like dequeue() removes from the front.

Q3. How is this a real-life queue?

At RRA, the first taxpayer to line up is the first to be served. Each new arrival joins at the rear.

Q4. How do queues improve customer service?
They ensure fairness—no one jumps the line. Customers are handled in the exact order they arrived, reducing conflicts.

Q5. Who is at the front now?
Steps:

- Enqueue Alice → [Alice]
- Enqueue Eric → [Alice, Eric]
- Enqueue Chantal → [Alice, Eric, Chantal]
- Dequeue() removes Alice → [Eric, Chantal]
- Enqueue Jean → [Eric, Chantal, Jean]

Front = Eric

Q6. Explain how a queue ensures fairness.
By FIFO rule, whoever submits first is processed first. Later arrivals must wait their turn, ensuring no favoritism.

Q7. Explain how each maps to real Rwandan life.

- **Linear queue** → people lining at a buffet; once food is finished, the line doesn't loop.
- **Circular queue** → buses at Nyabugogo return and re-enter the queue in a loop.
- **Deque** → school bus student enter from both front and rear doors; people can enter/exit from either side.

Q8. How can queues model this process?
Each food order is enqueued. When ready, the order is dequeued and delivered to customer whose on front of queue.

Q9. Why is this a priority queue, not a normal queue?
it's because at CHUK hospital, emergencies are treated before regular patients regardless of arrival time. That's **priority-based**,

while FIFO ensure that, who reaches first is one who is going to be served first.

Q10. How would queues fairly match drivers and students?
Passengers join a queue; drivers also wait in a queue. The system dequeues one from each queue, pairing them in fair arrival order.