BIT YEAR II - Data Structures Exercise

Part I - STACK

Q1

In the MTN MoMo app, every step you take while entering details is placed on top of the previous one, almost like stacking boxes. When you press the back button, it doesn't take away the very first thing you typed, but the last step you just added. That's basically the Last In, First Out idea. The last step is the first one to go. That's basically the Last In, First Out idea. The last step is the first one to go and Because the last thing you typed in is always the first thing to disappear when you go back. That's exactly how a stack work

Q2

Why is pressing back in UR Canvas like popping from a stack? The page or module you opened last is the first one to close when you press back, just like removing the top of a stack.

Q3

How do stacks make undo possible? Every action you take is placed on top of the stack. If you make a mistake, undo simply removes the last thing you did, without touching the older steps

Q4

How do stacks help check if forms are balanced? When you open a field or a bracket, it goes into the stack. When you close it, it gets removed. If everything matches properly, the stack will be empty at the end. If not, something went wrong .Think about Irembo forms. Every time you open a field, it's like pushing an opening bracket into a stack. If you finish or close that field properly, it's like popping the matching bracket. If one is missing or unmatched, the stack won't balance. That's how we know everything is correctly paired.

Q5

After these steps: Push(CBE notes), Push(Math revision), Push(Debate), Pop . Push(Group assignment) what's on top? The order becomes [CBE notes, Math revision, Group assignment]. So the top is Group assignment.

Q6

When you undo three times, it's like popping off the last three things you added. If it were during exams, the most recent answers would be removed and the earlier ones would stay safe in the stack. That's why undo doesn't affect everything, only the most recent actions.

Q7

In RwandAir booking, I picture it like filling a long form step by step. Each step is added on top of the stack. If I want to go back, the system will pop the last step I entered. This way, I can retrace my process one step at a time without losing the overall order.

Q8

Let’s reverse 'Umwana ni umutware' with a stack. I would push each word: Umwana, then ni, then umutware. The top is umutware, so if I pop, it comes out first, then ni, and lastly Umwana. So the proverb flips to 'umutware ni Umwana'. That’s how a stack reverses sequences neatly.

Q9

DFS, or Depth-First Search, means going as deep as possible before turning back. If I were in Kigali Public Library, I’d keep checking shelf after shelf down one path until I couldn’t go further. Then I’d backtrack. A stack is perfect here because it remembers where I came from last, letting me explore deeply instead of just level by level like a queue would.

Q10

In BK Mobile, every time I open a new transaction detail, it could be pushed into a stack. Going back would just pop the last viewed detail. A good feature would be something like undo/redo buttons—just like in browsers—so I could move forward and backward easily while checking my transactions.

Part II - QUEUE

Q1

The first person to arrive is the first to get food. New people go to the back of the line and wait their turn That’s exactly what FIFO means: First In, First Out.

Q2

In a YouTube playlist, the first video in the list plays first. When it ends, the next one starts automatically. That’s the same as dequeueing, where the front element of the queue is the one that leaves first.

Q3

At RRA offices, when people wait to pay their taxes, it looks like a textbook queue. Whoever came first is the first to be served, and the line continues until everyone is done. No jumping the lineit’s strictly FIFO.

Q4

At MTN or Airtel service centers, imagine the confusion if people didn’t wait their turn for SIM replacement. Queues make sure that everyone is served fairly in the order they arrived. This keeps things peaceful and efficient for both workers and customers.

Q5

Let’s trace it: Enqueue(Alice), Enqueue(Eric), Enqueue(Chantal). The queue is Alice → Eric → Chantal. Then Dequeue() removes Alice, so Eric moves to the front. Adding Jean makes it Eric → Chantal → Jean. So the person at the front is now Eric.

Q6

In RSSB, pension applications are handled by the order in which they arrive. That’s fair because nobody can skip ahead. A queue guarantees this fairness, and people trust the system more when it works that way.

Q7

Some real-life queue types: A Linear queue is like people waiting at a wedding buffet. One by one, they get food until the line ends. A Circular queue is like buses at Nyabugogo—after finishing a route, a bus circles back and joins the line again. A Deque is like boarding a bus where passengers can enter from either the front or the back door. Each case shows how queues can work differently.

Q8

In a Kigali restaurant, customers place their orders. These orders are queued in the kitchen. The first order placed is the first one prepared and served. When it’s ready, it’s dequeued, meaning the customer gets their food. That’s a perfect real-world queue process.

Q9

At CHUK hospital, emergencies like accidents are treated before patients who arrived earlier but have less urgent needs. This is called a priority queue because life-threatening issues are given higher importance. A normal queue would just go by arrival time, but that wouldn’t be fair in emergencies.

Q10

In moto or e-bike taxi apps, both riders and students wait for each other. A queue can ensure that the first rider available is matched with the first student request. This avoids unfair situations where someone who arrived late gets picked first. It keeps the service smooth and organized.