UNIVERSITY OF RWANDA ON24th/September/2025

HUYE CAMPUS

COLLEGE CBE

DEPARTMENT BIT

ASSIGNMENT OF DATA STRUCTURE AND ALGORITHMS

DUSENGIMANA Ferdinand

Reg:224010854

Part I – STACK

A. Basics

*Q1: **MTN MoMo app and LIFO behavior***

When filling payment details step-by-step, each new detail is pushed onto the stack. Pressing "back" removes the most recent step — the last one added. This is classic LIFO (Last In, First Out): the last action is undone first.

*Q2: **UR Canvas and Pop (Undo)***

Navigating course modules and pressing "back" undoes the last navigation step. This is like popping the top item from a stack — removing the most recent action to return to the previous state.

*B. **Application***

*Q3**: Stack for Undo in BK Mobile Banking***

Each transaction is pushed onto a history stack. To undo a mistake, the app can pop the last transaction — removing the most recent action without affecting earlier ones.

*Q4**: Balanced Parentheses in Irembo Forms***

Stacks help match opening and closing brackets. For forms, each opened section (e.g., a dropdown or nested field) is pushed. When a section is closed, it's popped. If all pairs match correctly, the form is balanced — just like checking parentheses.

*C**. Logical***

*Q5**: Stack sequence — What's on top?***

Sequence:

- Push("CBE notes")

- Push("Math revision")

- Push("Debate")

- Pop() → removes "Debate"

- Push("Group assignment")

Top of stack: "Group assignment"

*Q6**: Undo 3 actions in ICT exam***

Assuming the stack had 5 answers: A1, A2, A3, A4, A5

Undoing 3 Pops removes A5, A4, A3

Remaining: A1, A2

*D. **Advanced Thinking***

Q7: RwandAir booking and backtracking

Each form step is pushed onto a stack. Pressing "back" pops the last step, allowing the user to retrace their path step-by-step — just like backtracking in a stack.

Q8: Reversing "umukiriya ni umwami" using a stack

Push each word:

- Push("umukiriya")

- Push("ni")

- Push("umwami")

Pop each word:

- Pop() → "umwami"

- Pop() → "ni"

- Pop() → "Umukiriya"

Reversed: “umwami ni Umukiriya”

*Q9**: DFS in Kigali Library***

Depth-First Search (DFS) explores one path deeply before backtracking. A stack suits this because it remembers the last shelf visited and allows deep exploration. A queue would explore broadly, not deeply.

*Q10**: BK Mobile navigation with stacks***

Feature idea: “Jump Back” — each transaction viewed is pushed onto a stack. Pressing “Back” pops the last viewed transaction, letting users retrace their navigation history.

Part II – QUEUE

A. Basics

*Q1: **Restaurant in Kigali and FIFO***

Customers are served in the order they arrive. First In, First Out: the first customer to enter is the first to be served — just like dequeueing from a queue.

*Q2**: YouTube playlist and Dequeue***

Videos play in order. The next video (at the front of the queue) is played and removed — just like a dequeue operation.

*B**. Application***

Q3: RRA tax line as a queue

People arrive and are added to the end of the line (enqueue). They’re served in order — the first person is dequeued first. This models a real-life queue.

*Q4: **MTN/Airtel SIM replacement***

Requests are handled in order of arrival. Queues ensure no one skips ahead, improving fairness and efficiency in customer service.

*C**. Logical***

*Q5**: Equity Bank sequence***

- Enqueue("Alice")

- Enqueue("Eric")

- Enqueue("Chantal")

- Dequeue() → removes "Alice"

- Enqueue("Jean")

Front of queue: Eric

*Q6: **RSSB pension fairness***

Applications are processed in arrival order. A queue ensures fairness by serving the earliest applicant first — no jumping ahead.

*D**. Advanced Thinking***

*Q7: **operation;different queue types***

- Linear queue: Wedding buffet — people line up and move forward as served.

- Circular queue: Nyabugogo buses loop and reuse the same route — the queue wraps around.

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

*Q8**: Enqueue orders, Dequeue when ready***

In a restaurant, orders are placed (enqueue) and prepared in order. When ready, they're served (dequeue). This ensures timely and fair service.

Enqueue Orders, Dequeue When Ready — Kigali Restaurant

How it works:

- Enqueue: When a customer places an order, the kitchen adds it to the end of the task list — this is the queue.

- Processing: The kitchen prepares orders in the same order they were received.

- Dequeue: Once an order is ready, it's removed from the front of the queue and served.

Why this models a queue:

- It follows FIFO (First In, First Out): the first customer to order is the first to be served.

- It ensures fairness and predictability — no one skips ahead unless there's a special rule (like priority).

Real-life benefits:

- Reduces confusion and complaints.

- Helps staff manage workload efficiently.

- Can be automated in digital systems (e.g., POS apps or food delivery platforms).

*Q9: **Priority Queue — CHUK Hospital Emergencies***

How it works:

- Patients arrive and are assessed.

- Emergencies (e.g., heart attack, severe bleeding) are given higher priority.

- These patients are treated before others, even if they arrived later.

Why this is a priority queue:

- Unlike a normal queue, where order is based on arrival time, a priority queue uses urgency as the deciding factor.

- Each item (patient) has a priority level — higher priority items are dequeued first.

Real-life benefits:

- Saves lives by treating critical cases quickly.

- Ensures medical resources are used effectively.

- Can be managed using triage systems or digital health platforms.

*Q10**: **Enqueue/Dequeue Matching System** — Moto/E-bike Taxi App*

How it works:

- Students request rides — they're enqueued in a waiting list.

- Drivers become available — they're also enqueued in a separate list.

- The system matches one student with one driver by dequeuing both.

***Why this models a queue**:*

- It ensures fair matching — the first student to request gets the first available driver.

- If multiple students and drivers are waiting, the system processes them in order.

Real-life benefits:

- Prevents long wait times or skipped requests.

- Can be enhanced with location-based priority (e.g., nearest driver).

- Supports scalability — works well even with hundreds of users.