Date : 24 /9/2025

NAME : SHUTIYIMANA Carmen

Module : Data structure and algorthim

Registration Number : 224014222

Year 2

School of Business

College of Business And Economics

PartI: STACK

1.The MTN MoMo app example shows the LIFO (Last-In, First-Out) nature of stacks in the way it handles navigation through steps when entering payment details.

Action Stack (Steps Completed)

Entered Amount [Amount]

Chose Recipient[Amount, Recipient]

Confirmed Details [Amount, Recipient, Confirmation]

Pressed Back (1st time) [Amount, Recipient]

Pressed Back (2nd time)[Amount]

The last action done Confirmation is the first to be undone — exactly how a stack works.

2.UR Canvas, when you navigate course modules, pressing back undoes the last step. it is similar to poping a stack because it flow papping operation like stack removes the top item first.

The top item here is the most recent page visited.

Pressing Back removes (or “pops”) that page, taking you to the one before it.

Pressing Back in UR Canvas works like a pop from a stack because it undoes the last step by removing the most recent item — exactly how the pop operation works in a stack data structure. for example Let's say your navigation looks like this:

Action Navigation Stack

Opened Home [Home]

Clicked Module 1 [Home, Module 1]

Clicked Lesson 3 [Home, Module 1, Lesson 3]

Pressed Back [Home, Module 1] ← Popped Lesson 3

3. in an app like BK Mobile Banking, if a user is performing a series of actions (e.g. filling a form, entering amounts, selecting accounts), each step or change can be pushed onto a stack. Each user action (e.g. entering a number, selecting a beneficiary) is pushed onto the stack.If the user makes a mistake, they can press "Undo".

The system will then pop the last action from the stack — removing the most recent change.

The user sees the screen go back to the previous state
A stack enables the undo function by keeping track of each step. When the user presses undo, the app pops the last action from the stack and reverts to the previous state — exactly how LIFO (Last-In, First-Out) works. Push lets the app save each action.

Pop lets the app undo the most recent action — just like going backward in time.

4.Every opening symbol (like (, {, or [)

has a matching closing symbol (like), }, or])

in the correct order.

This same logic applies to forms, where every input field started must also be completed or closed properly. A stack ensures form sections are opened and closed in the correct order. Just like with parentheses, if every opening action (like starting a section) is properly matched with a closing action (like finishing it), and the stack is empty at the end, the form is balanced and valid.

5.Group assignment A stack keeps tasks in the order they were added, with the most recent on top.

The student removed the last task ("Debate") and then added a new one.

So, the top task now is "Group assignment", and that's what they'll do next

6.The first item added is the first one to be processed.

Just like people lining up in a queue at a bank — whoever comes first is served first. This is called FIFO: First In, First Out.

In RSSB Pension Applications:When people submit pension applications:

Each application is added to the queue in the order it is received. How a Queue Ensures Fairness:

No one can skip the line — applications are handled in the order they arrive.Everyone gets equal treatment — no one is processed early just because they applied later.

It avoids bias or confusion about who should be served next.

The system then processes them one by one, starting with the oldest (the one that arrived first)

7.1.inear Queue

People join at the back, leave from the front (First In, First Out – FIFO).

Example: People lining up at a wedding buffet.

- One-way line — first come, first served.

2. Circular Queue

Works like a circle — after the end, it starts again.

Example: Buses at Nyabugogo terminal going in loops.

- Repeats in a cycle — comes back to the beginning.

3. Deque (Double-ended Queue)

You can add or remove from both front and back.

Example: People boarding a bus from front or back doors.

: Two-way access — enter or exit from either end.

3. Deque (Double-ended Queue)

You can add or remove from both front and back.

Example: People boarding a bus from front or back doors.

: Two-way access — enter or exit from either end.

8,Customer 1 places an order → added to the queue.

Customer 2 places an order → added behind Customer 1.

The kitchen prepares food in the same order.

When Customer 1's food is ready, they are called first (dequeued).

Then Customer 2, and so on.

queue models this restaurant process because it handles orders in the same sequence they are placed.

First In = First Out — just like people waiting to be served food.

9.a normal queue uses FIFO (First In, First Out):

The first person in line is the first to be served, no matter the situation.

But in a priority queue, each item (or person) has a priority level.

Higher priority = served earlier, even if they came late

CHUK Hospital Example:

A patient with a broken arm comes first — normal case.

A patient with a heart attack arrives later — emergency.

Even though the first patient came earlier, the emergency patient is seen immediately

10.Enqueue passengers:

When a student requests a ride, they are added to the passenger queue.

2.Enqueue drivers:

Available moto/e-bike drivers are added to a driver queue.

Match them (Dequeue both):

When a match is made, the first driver in the driver queue is assigned to the first student in the passenger queue.

Both are then dequeued (removed from the front of their queues).

PART TWO QUEQUE

1. This scenario shows FIFO (First-In, First-Out) behavior because customers are served in the exact order in which they arrive.

- Enqueue represents a customer arriving and being added to the rear of the line (queue).
- Dequeue represents serving a customer from the front of the line.

Since the first customer to arrive is the first to be served, and new customers must wait their turn behind those who arrived earlier, it perfectly demonstrates the FIFO principle — the first person in the line is the first one out.

2. This action is similar to popping from a stack because it undoes the most recent action first, just like a stack removes the top (last added) item.

In UR Canvas (or similar apps), as you navigate through course modules:

- Each step (like opening a module or page) is pushed onto a stack.

- When you press back, the system pops the last step off the stack — essentially reversing your most recent

This follows the LIFO (Last-In, First-Out) principle of a stack:

- The last action you performed is the first one to be undone.

So, pressing "back" works just like a pop operation on a stack.

This follows the LIFO (Last-In, First-Out) principle of a stack:

- The last action you performed is the first one to be undone.

So, pressing "back" works just like a pop operation on a stack.

This follows the LIFO (Last-In, First-Out) principle of a stack:

- The last action you performed is the first one to be undone.

3.A stack can enable the undo function by storing each user action (like transactions or steps) as it happens — using the push operation.

Here's how it works:

- When a user performs an action (e.g., entering an amount, confirming a transfer), that action is pushed onto the top of the stack.
- If the user makes a mistake and wants to undo it, the system performs a pop — removing the most recent action first.

4. • When a user starts filling a section (like opening a group of fields), the system pushes a marker (like an opening bracket) onto the stack.

• When the user completes that section (like closing the group or submitting), the system checks if there’s a matching opening marker on the stack to pop.

• If every opening has a matching closing, the stack will be empty at the end, meaning the form is balanced and correctly filled.

• If the stack is not empty or a closing appears without a matching opening, it indicates an error or mismatch in the form structure.

Using a stack helps the system track nested or grouped fields and ensures that all opened sections are properly closed — maintaining a balanced and error-free form structure.

5. Group assignment is at the top of the stack, so it's the next task.

6. Remaining in the stack:

- Answer Q1
- Answer Q2

7. A stack enables backtracking in the RwandAir booking form by recording each step the passenger takes and allowing them to go back one step at a time, using pop operations.

- Each new step in the booking process (e.g., selecting flight, entering passenger info, choosing seats) is pushed onto the stack.

- When the passenger clicks "Back", the system performs a pop to remove the most recent step and return to the previous one.

- This retraces the user's path in reverse order — the same way a stack works (LIFO: Last-In, First-Out).

8.- Push each word onto the stack:
Stack: `["Umwana", "ni", "umutware"]`

-Pop each word to reverse:
Popped: `"umutware", "ni", "Umwana"`

✓ Reversed proverb: "umutware ni Umwana"

9. A stack suits DFS because it supports deep, backtracking search, which matches how a student explores shelves in depth — one path at a time.

- 10. Every transaction the user views or performs is pushed onto a stack.
- When the user taps “Back” or “Undo”, the app pops the latest transaction from the stack to return to the previous transaction or state.
- This lets users easily revisit or undo recent transactions step-by-step in reverse order.

You could also add a “Redo” feature by pushing popped transactions onto a second stack, enabling forward navigation again.