Names: Jules Maurice UWIZEYIMANA

Reg Number: 224014947

Depart: Business Information Technology

Date: 24/09/2025

Course: Data structures and Argolithims , Assignment Q122

1) STACK QUESTIONS

A stack is Last-In-First-Out (LIFO). push adds to the top; pop removes from the top. We'll use a Python list where append() = push and pop() = pop.

1.A Practical (UR)

Task: UR pushes ["Assignment1", "Assignment2", "Assignment3"]. Pop one. Which is top?

Algorithm (short)

1. Start with an empty stack [].

2. Push "Assignment1" → stack ["Assignment1"].

3. Push "Assignment2" → stack ["Assignment1","Assignment2"].

4. Push "Assignment3" → stack ["Assignment1","Assignment2","Assignment3"].

5. Pop one → removes the top item ("Assignment3").

6. The new top is the last remaining element.

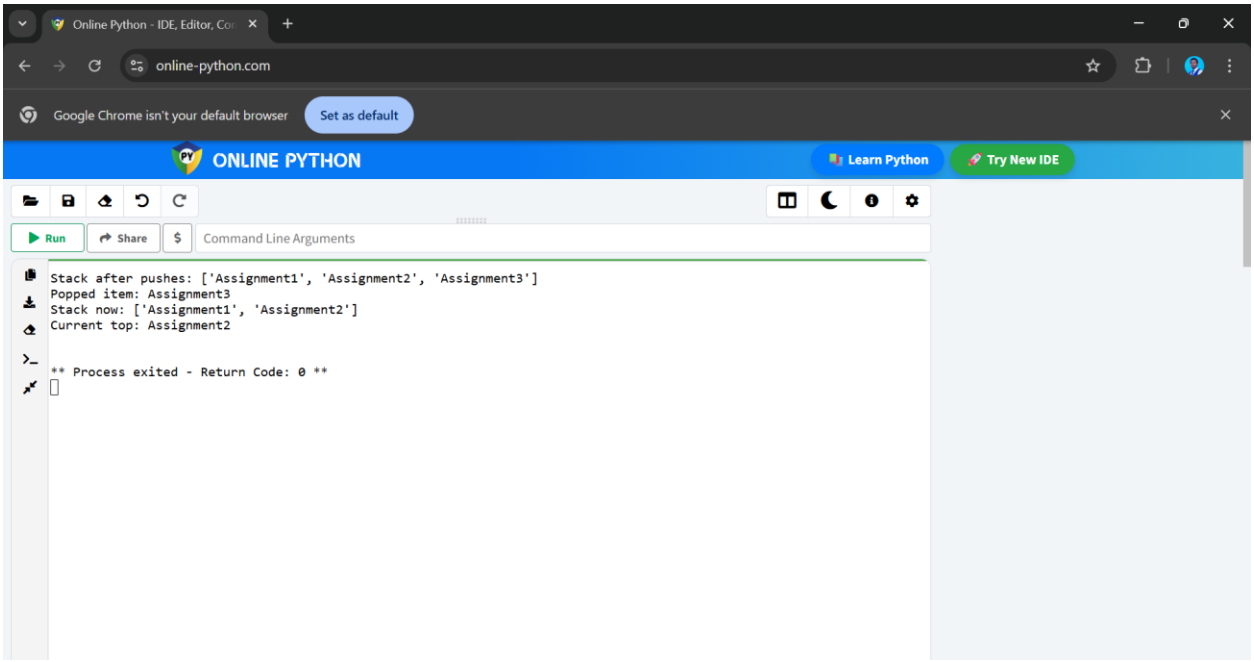
Python code (practical 1)

Stack practical UR

```
stack = []

 pushes

stack.append("Assignment1")

stack.append("Assignment2")
```

```
stack.append("Assignment3")
print("Stack after pushes:", stack)
 pop one
popped = stack.pop()
print("Popped item:", popped)
print("Stack now:", stack)


 top (if any)
top = stack[-1] if stack else None
print("Current top:", top)
```

Output (screenshot-style)



Stack after pushes: ['Assignment1', 'Assignment2', 'Assignment3']

Popped item: Assignment3

Stack now: ['Assignment1', 'Assignment2']

Current top: Assignment2

Answer: After popping one, "Assignment2" is on top.

Explanation: Because the last pushed element ("Assignment3") was removed, the element pushed just before it ("Assignment2") becomes the new top. That's LIFO in action.

1.B Practical (Irembo)

Task: In Irembo, push ["Form1","Form2","Submit"]. Undo all. Which remains?

Algorithm

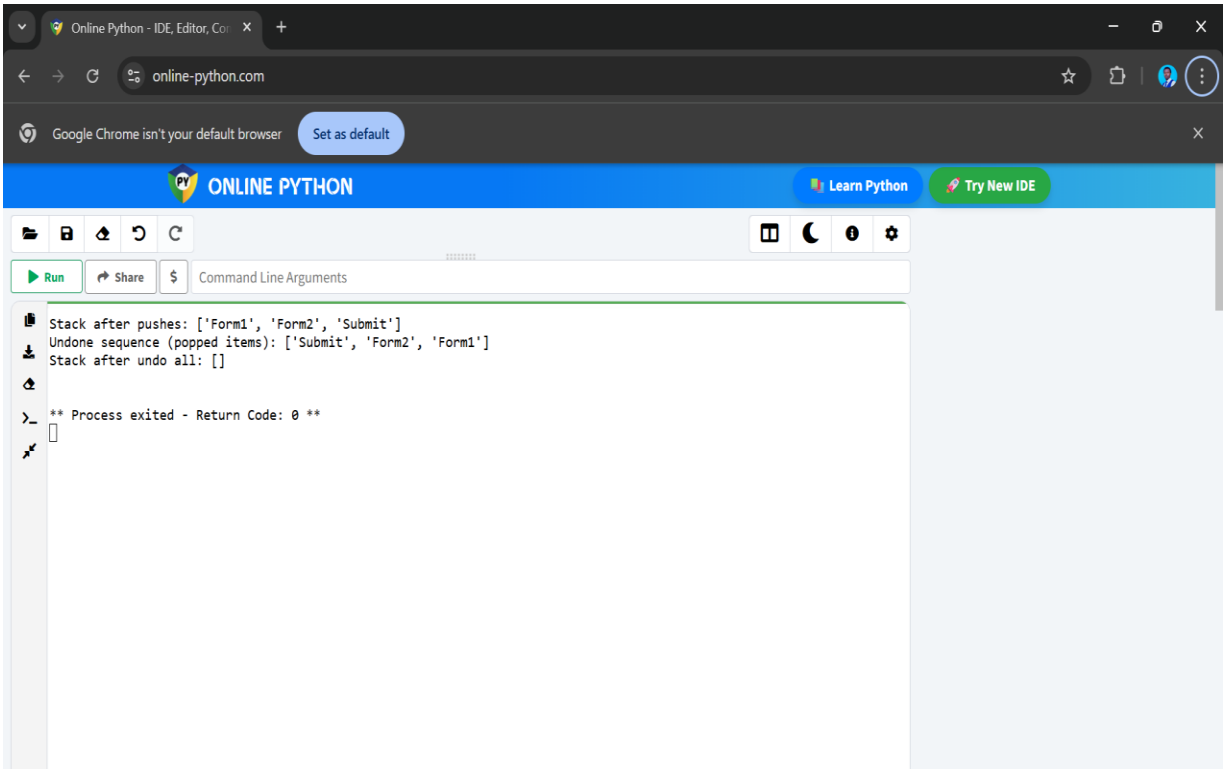
1. Push Form1, Form2, Submit (stack has 3 items).
2. "Undo all" means repeatedly pop until the stack is empty.
3. If all operations are undone, nothing remains; the stack becomes empty.

Python code (practical 2)

```
 Stack practical Irembo
stack = []
stack.append("Form1")
stack.append("Form2")
stack.append("Submit")

print("Stack after pushes:", stack)

Undo all: pop until empty
```

```
undone = []
while stack:
    undone.append(stack.pop())

print("Undone sequence (popped items):", undone)
print("Stack after undo all:", stack)
```

Output (screenshot-style)



Stack after pushes: ['Form1', 'Form2', 'Submit']

Undone sequence (popped items): ['Submit', 'Form2', 'Form1']

Stack after undo all: []

Answer: Nothing remains , the stack is empty.

Explanation: Undoing all reverses each action in LIFO order: first "Submit" is undone, then "Form2", then "Form1". After undoing every push, the stack has zero items.

1.C Challenge

Task: Push ["X","Y","Z"], pop 2, push "W". Which is top?

Step-by-step

1. Start empty [].

2. Push X → ["X"].

3. Push Y → ["X","Y"].

4. Push Z → ["X","Y","Z"].

5. Pop 2 → removes Z then Y, remaining ["X"].

6. Push W → ["X","W"].

7. Top is last element: "W".

Answer: "W" is on top.

Why: After removing the two most recent pushes (Z and Y), X remains, then pushing W puts W above X.

1.D Reflection

Question: Why stack undoes last edits first?

Explanation :

. A stack models a chain of actions where each new action is placed on top.

. Undoing must revert the most recent action before older actions , otherwise you could leave later actions depending on earlier ones inconsistent.

. LIFO guarantees consistency: the last change (top of stack) is reversed first, then the previous one, etc.

Example: if you type characters “A”, “B”, “C” then undo twice, you expect “A” to stay , undoing LIFO keeps edits in correct temporal order.

2) QUEUE QUESTIONS

A queue is First-In-First-Out (FIFO). enqueue adds at the back, dequeue removes from the front.

2.A Practical (Airtel)

Task: At Airtel, 7 clients queue. After 2 served, who is in front?

Interpretation & Assumption

Clients are in order: Client1 (front), Client2, Client3, ..., Client7 (back). Serving removes from the front.

Step-by-step

1. Queue initially: Client1, Client2, Client3, Client4, Client5, Client6, Client7.

2. After serving (dequeue) 1 → Client2 is new front.

3. After serving (dequeue) 2 → Client3 is new front.

Answer: Client3 is in front.

Explanation: Removing two customers from the front moves the head forward by two positions.

2.B Practical (CHUK)

Task: At CHUK, 9 patients queue. Who is served third?

Interpretation & Assumption

Queue order: Patient1, Patient2, Patient3, ..., Patient9. Serving order is front → back.

Answer: Patient3 is served third.

Explanation: First served is Patient1, second is Patient2, third is Patient3 , FIFO order.

2.C Challenge

Question: Queue vs stack for student registration. Which is correct?

Answer & Explanation:

Queue (FIFO) is correct for student registration where fairness and order of arrival matter. Students who come earlier should be served earlier.

Stack (LIFO) would let the most recent arrival get served first — that's unfair and not suitable for registration.

Use a queue for arrival-based services (registration, ticketing), use a stack for "undo" operations or nested-call structures.

2.D Reflection

Question: Why FIFO ensures fairness in universities?

Explanation:

. FIFO serves people in the order they arrived; no one who arrived later cuts ahead.

. This predictable ordering is easy to explain to users and enforce, avoiding disputes.

. For resource-limited services (registrations, counseling), FIFO matches normal expectations of fairness.

In a nutshell

Stack (LIFO)

Use for: undo history, recursion, backtracking, browser back button.

Push = add to top; Pop = remove top.

Example: push A,B,C → pop → top is B.

Queue (FIFO)

Use for: customer service lines, task scheduling, breadth-first search.

Enqueue = add to back; Dequeue = remove from front.

Example: enqueue 1..7 → dequeue twice → new front is 3.