Assignment: GitHub Repo + Recursion (No Two Consecutive Activities)

Course: Problem Solving / Level 3
Release Date: 2025-08-12
Due: *2025-08-12*

Goal

1. Create a GitHub account and a repository named `gsg_level_3_codes`.
2. Implement a program that lists **all valid activity schedules** for `n` days (where `1 ≤ n ≤ 10`) using **Swimming**, **Running**, and **Football**, with the rule: **no two consecutive days can have the same activity**.

Part A — GitHub Setup (10%)

1. Go to https://github.com and create an account (if you don't have one).
2. Create a **public** repo named exactly: `gsg_level_3_codes`.
3. Add a top-level `README.md` with your name, ID, and a short description.
4. Create a folder: `activity_schedules/` for this assignment.

Deliverables in repo:

- `activity_schedules/solution.cpp`

Part B — Programming Task (80%)

Problem Statement

Given a number of days `n` (**1 ≤ n ≤ 10**), print **all** possible sequences of daily activities using the set: `{"Swimming", "Running", "Football"}`
subject to the constraint that **no two consecutive days** have the **same** activity.

Input

- A single integer `n` (1 ≤ n ≤ 10).

Output

- Print **each valid schedule** on its own line, with activities separated by a single space.
- After listing all schedules, print a final line: `COUNT: X` where `X` is the total number of schedules generated.

Example

Input

```
2
```

Valid schedules

```
Football Running
Football Swimming
Running Football
Running Swimming
Swimming Football
Swimming Running
COUNT: 6
```

Hints

- This is a **recursion** problem.
- At each day, try the 3 activities but **skip** the one equal to the **previous day's** choice.

Part C — Repo Quality & README (10%)

Your `activity_schedules/README.md` should include:

- Problem statement (copied briefly)
- Example run with `n = 2`
- Expected `COUNT` formula and value for your chosen `n` tests
- Your name and ID

Submission

- Share your repo URL (e.g., `https://github.com/<username>/gsg_level_3_codes`) on the classroom.
- Ensure the repo stays public until grading is finished.

Bonus (Optional)

- Write an **iterative** version (no recursion).

May Allah Grant you success!