Project Report: Phone User Generation and Filtering System

1. Introduction

This project focuses on creating a Python program that generates a dataset of phone users in the United States. Each user is assigned a random full name, a mobile operator (Verizon, AT&T, or T-Mobile), a phone number from predefined lists, and a subscription year. The program also allows the user to filter the displayed data by operator or view all users at once.

The objective was to simulate a small-scale user database and practice working with lists, loops, conditionals, and formatted console output.

---

2. Project Description

The system works by:

1. **Generating random user information**
    - First name and last name are selected randomly.
    - Subscription year is randomly generated between 2018 and 2025.
2. **Assigning a phone operator**
    - Three operators are available: Verizon, AT&T, and T-Mobile.
    - Each operator has a predefined list of 10 numbers, assigned sequentially.
3. **Storing user details**
    - Each generated user is stored in a list as a dictionary containing:
        - Name
        - Operator
        - Subscription year
        - Phone number
4. **Displaying data based on user choice**
    - The program displays a menu:
        1. Verizon
        2. AT&T
        3. T-Mobile
        4. All
    - Based on the selection, the program filters and prints formatted data.

---

3. Code Summary

The code uses:

- `random` module for randomized user generation and importing the random liberory
- Lists for storing names, operators, and phone numbers

- Loops to generate 30 total users (10 for each operator)
- Conditionals to filter users based on input
- String formatting to display output in a clean table format

---

4. Difficulties Faced

Throughout the development of this project, several challenges were encountered:

1. Ensuring Phone Number Distribution

Each operator had exactly 10 phone numbers available. Careful indexing was needed (`v_i`, `a_i`, `t_i`) to ensure no number was repeated and no index went out of range.

It was hard to collect avaibale data (names,numbers)

2. Maintaining Even User Generation

Since all operators needed exactly 10 users, loops had to be structured carefully. Initially, errors occurred when the loop structure caused uneven distribution or repeated numbers.

3. Filtering Logic

When filtering users based on the operator selection, we had to:

- Create new lists dynamically
- Ensure operators were compared correctly using strings
- Handle unexpected inputs (e.g., anything outside 1–3 defaults to “All Users”)

4. Formatting Output

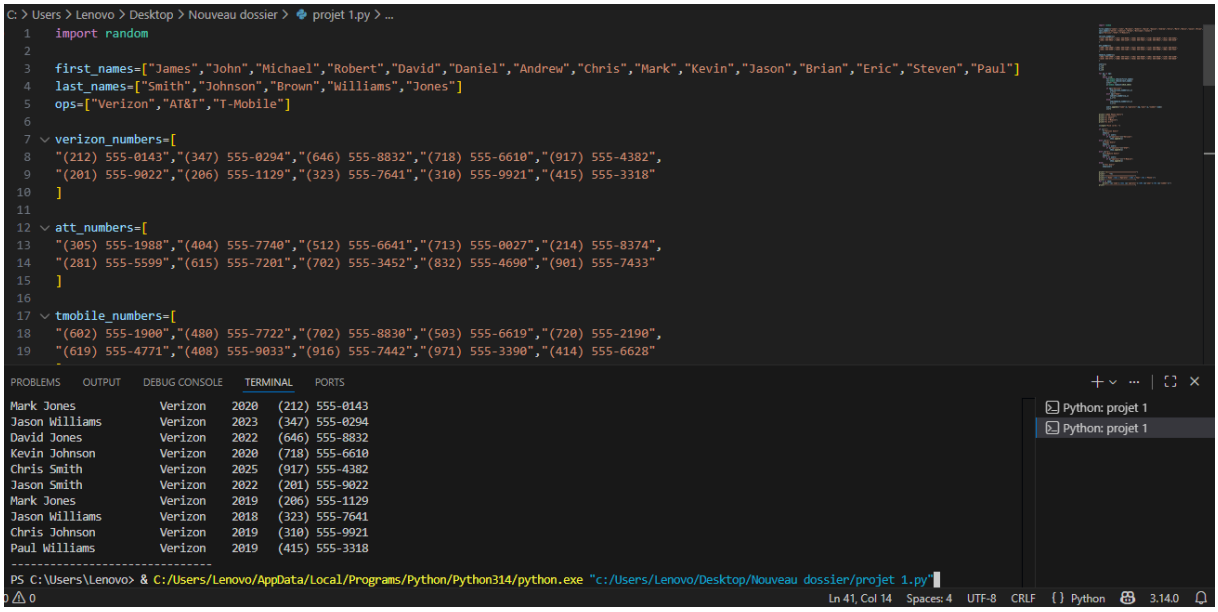
Printing user information in a readable, aligned table required:

- Mastery of formatted f-strings organized outputs
- Adjusting spacing for long names or operator names
- Ensuring the table remained aligned regardless of values

5. Conclusion

The project successfully simulates a phone user database and demonstrates essential programming concepts such as lists, dictionaries, loops, and user interaction. Despite some early difficulties with data distribution, filtering, and formatting and the logical thinking, the final system works efficiently and displays information in a clean, user-friendly manner.

This project provided valuable practice in Python programming, data handling, and console-based presentation of structured information.



Members :

- *Adollah Larbi cherif*
- *Khali ishak*
- *Mouas ayoub*
- *Bouzebra med Amine*