****

**Name : Talha Usman**

**Class: SU92-BSAIM-F24-064**

**Subject: AI-Lab**

**Documentation – Python Tasks**

**Introduction:**

This document explains three Python programming tasks that demonstrate concepts of **string processing, algorithms, and validation techniques**.

* **Task 1** implements a card number validation method using the **Luhn Algorithm**.
* **Task 2** focuses on text cleaning by removing special characters.
* **Task 3** applies the **Bubble Sort algorithm** to arrange characters in a string based on their **ASCII values**.

Each task includes a description of the problem, how the solution works, and examples of input and output.

**Task 1: Card Validation (Luhn Algorithm)**

**Introduction**

Card numbers such as those used in debit and credit cards require validation to ensure they are correct before being processed. The **Luhn Algorithm** is a simple checksum formula used worldwide to validate identification numbers.

**How It Works**

1. Extract the **last digit** of the card number (this is the check digit).
2. Process all other digits from **left to right**.
3. Double the digits at **even positions**.
4. If a doubled value becomes **greater than 9**, subtract 9.
5. Add together all digits, including the last digit.
6. If the **final sum is divisible by 10**, the card number is valid; otherwise, it is invalid.

**Example**

* Input: 4532015112830366
* Output: Valid

**Task 2: Remove Special Characters**

**Introduction**

User input often contains unwanted characters such as punctuation marks, symbols, or special signs. To ensure clean data, only letters, digits, and spaces should be kept.

**How It Works**

1. Take a string as input.
2. Go through each character one by one.
3. Keep the character only if it is:
   * A letter (A–Z, a–z)
   * A digit (0–9)
   * A space " "
4. Combine these characters to form the cleaned string.

**Example**

* Input: Hello@World!123
* Output: HelloWorld123

**Task 3: ASCII Bubble Sort**

**Introduction**

Sorting is one of the most important operations in computer science. This task demonstrates how to sort characters of a string by comparing their **ASCII values** using the **Bubble Sort algorithm**.

**How It Works**

1. Convert the string into a list of characters.
2. Compare two adjacent characters using their **ASCII codes**.
3. Swap them if they are not in the correct order.
4. Repeat the process until the list is fully sorted.
5. Display the sorted characters.

**Example**

* Input: zebra
* Output: ['a', 'b', 'e', 'r', 'z']

**Conclusion**

* **Task 1** shows how to validate card numbers with a reliable algorithm.
* **Task 2** ensures data cleanliness by removing unwanted characters.
* **Task 3** demonstrates sorting logic using ASCII values and bubble sort.

Together, these tasks cover practical uses of algorithms, validation techniques, and string manipulation in Python.

**Output of code:**

