

**BS Artificial Intelligence**

**Name:** Hafiz M. Muneeb Akbar

**Roll No:** SU92-BSAIM-F24-048

**Subject:** Artificial Intelligence (Lab)

**Lab Task 4**

**Step-by-Step Explanation of Code (LUHN Algorithm, Remove Punctuation, Sorting)**

This document explains three different Python tasks step by step:  
**1.** LUHN Algorithm for card number validation  
**2.** Removing punctuation from a given string  
**3.** Sorting a sentence alphabetically (by characters and by words)

**Task 1: LUHN Algorithm**

The LUHN algorithm is used to validate credit/debit card numbers. It checks whether the given card number follows a valid structure.

**Function Definition**

A function named 'luhn' is defined which accepts a card number as input.

**Extract Last Digit**

The last digit of the card number is stored in 'last\_digit'. This is the check digit.

**Convert to Digits**

All digits except the last one are converted to integers and stored in a list 'digits'.

**Reverse Digits**

The digits list is reversed so that we can start doubling from the right side.

**Double Alternate Digits**

Every second digit (at even indices after reversing) is doubled. If the doubled value is greater than 9, subtract 9 from it.

**Calculate Total Sum**

The sum of all processed digits is calculated, then the last digit is added.

**Validation Check**

If the total sum modulo 10 equals 0, the card number is valid; otherwise, it is not.

**Example Execution**

The code checks '6074227196220459' and prints whether it is valid or not.

**Output Screenshot**A screenshot of a computer program

AI-generated content may be incorrect.

Figure : showing that the number is Valid or Not.

**Task 2: Remove Punctuation from a String**

This task removes punctuation from a given string while keeping only letters, numbers, and spaces.

**Original String**

The text **'Hey, Man; this is' not over, yet'** is used as input.

**Result Variable**

An empty string 'result' is initialized to store cleaned characters.

**Loop Through Characters**

For each character in the string, check if it is alphanumeric or a space.

**Build Cleaned String**

If valid, add it to **'result'**. This removes commas, semicolons, and apostrophes.

**Output**

The original and cleaned string are both printed.

**Output Screenshot**

A computer screen shot of a computer code

AI-generated content may be incorrect.

Figure : Shows how we can remove the punctuations.

**Task 3: Sorting a Sentence Alphabetically**

This task sorts a sentence alphabetically in two ways: by characters and by words.

**Sorting by Characters**

**Input String**

The string **'I am Muneeb'** is used as input.

**Convert to Characters**

Each character in the string is stored in a list.

**Bubble Sort**

A nested loop compares ASCII values of characters and swaps them if needed **(Bubble Sort algorithm).**

**Output**

Characters are printed in sorted order as a continuous string **(IMabeemnu).**

**Output Screenshot**A screen shot of a computer code

AI-generated content may be incorrect.

Figure : Sorting by Character.

**Sorting by Words**

**Input String**

The string **'I am Muneeb'** is used again.

**Word Splitting**

The string is split into words manually by checking spaces.

**Bubble Sort on Words**

The words are compared lexicographically **(letter by letter)** and swapped if needed.

**Output**

Words are printed in alphabetical order as: **'I Muneeb am'.**

A screen shot of a computer code

AI-generated content may be incorrect.

Figure : Sorting the sentence by words with the help of Bubble sorting method.