**Manual of Code**

**Lab 4.3**

**Explanation of the Code**

This code takes a sentence as input from the user, splits it into individual words, sorts the words in **alphabetical order**, and then joins them back into a single string. The sorting is done using the **Bubble Sort** algorithm, which compares adjacent words and swaps them if they are in the wrong order.

**How the Code Works**

**1. The sort\_word Function**

The sort\_word function takes a sentence (as a string) and sorts the words in alphabetical order. Here’s how it works:



**2. Split the Sentence into Words**

The input sentence is split into a list of words using the split() method. By default, split() splits the string at spaces.

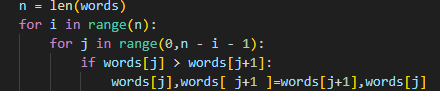


For example, if the input is "apple banana cherry", the words list will be:

["apple", "banana", "cherry"]

**3. Bubble Sort Algorithm**

The code uses the **Bubble Sort** algorithm to sort the words in alphabetical order. Bubble Sort works by repeatedly comparing adjacent elements and swapping them if they are in the wrong order.



* **Outer Loop (for i in range(n))**:
  + This loop runs n times, where n is the number of words.
  + After each iteration, the largest unsorted word "bubbles up" to its correct position at the end of the list.
* **Inner Loop (for j in range(0, n - i - 1))**:
  + This loop compares adjacent words and swaps them if they are in the wrong order.
  + For example, if words[j] is "banana" and words[j + 1] is "apple", they will be swapped because "banana" comes after "apple" alphabetically.

**4. Join the Sorted Words**

After sorting the words, they are joined back into a single string using the join() method. A space (" ") is used as the separator between words.



For example, if the sorted list is:

["apple", "banana", "cherry"]

The joined string will be:

"apple banana cherry"

**5. Return the Sorted Sentence**

The function returns the sorted sentence.



**6. User Input and Function Call**

The program prompts the user to enter a sentence, calls the sort\_word function to sort the words, and prints the result.



**Example Walkthrough**

Let’s say the user enters the following sentence:

"banana apple cherry"

1. The sentence is split into a list of words:

["banana", "apple", "cherry"]

1. The Bubble Sort algorithm sorts the words:
   * Compare "banana" and "apple" → Swap them:

["apple", "banana", "cherry"]

* + Compare "banana" and "cherry" → No swap needed.

1. The sorted list of words is:

["apple", "banana", "cherry"]

1. The words are joined back into a single string:

"apple banana cherry"

1. The program prints:

Sorted words in alphabetical order: apple banana cherry