**Rule for sorting**:

This algorithm used is bubble sort, which continuously swaps adjacent elements if they satisfy the given condition i.e.., in greater than( Ascending i<j) or less than( descending i>j).

The sorting that’s been done is based on the ASCII value of the strings.

**Ex:**

[e,a,d,b,f] swaps to [a,e,d,b,f] ( As the first two elements are in the condition e>b)

[a,e,d,b,f] swaps to [a,d,e,b,f] as e>d

[a,d,e,b,f] swaps to [a,d,b,e,f] as e>b

and continues till it reaches to [a,b,d,e,f]

**Code**:

l=[] **#list for storing length of strings in given list**

p=[] **#list for storing index of strings in the list**

import matplotlib.pyplot as plt

def sortf(list): **#code for sorting**

k=len(list)

for i in list:

l.append(len(i))

p.append(list.index(i))

for i in list:

m=list.index(i)

for j in range(0,k-m-1):

if(list[j]>list[j+1]):

a=list[j]

list[j]=list[j+1]

list[j+1]=a

plt.scatter(p,l)

print(l)

print(p)

plt.show()

print(list)

def Convert(string) **#code for converting given text in to list of strings**

li = list(string.split(" "))

print(li)

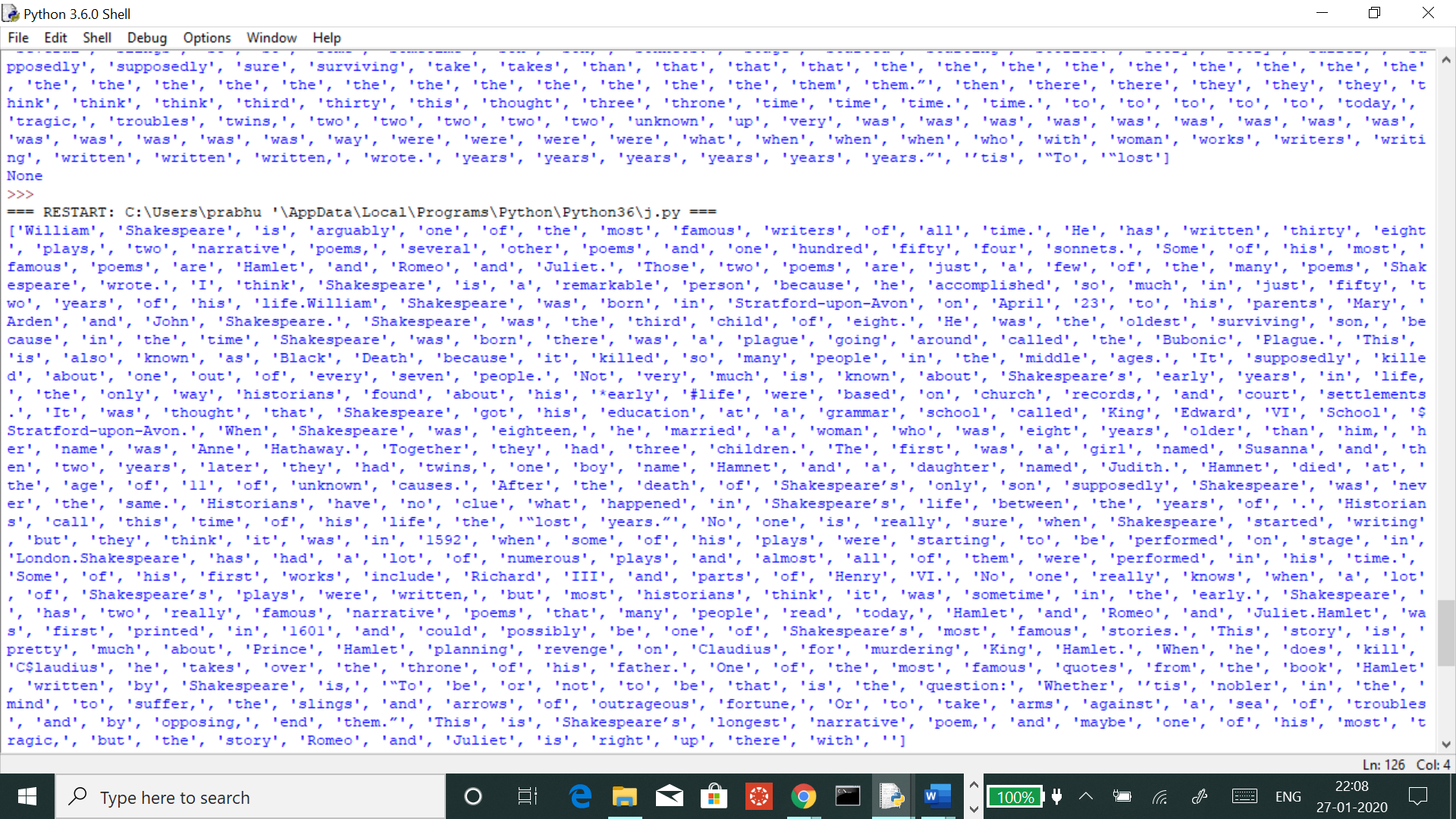
sortf(li)

str1 = " " # **Text that is to be converted in to list of strings**.

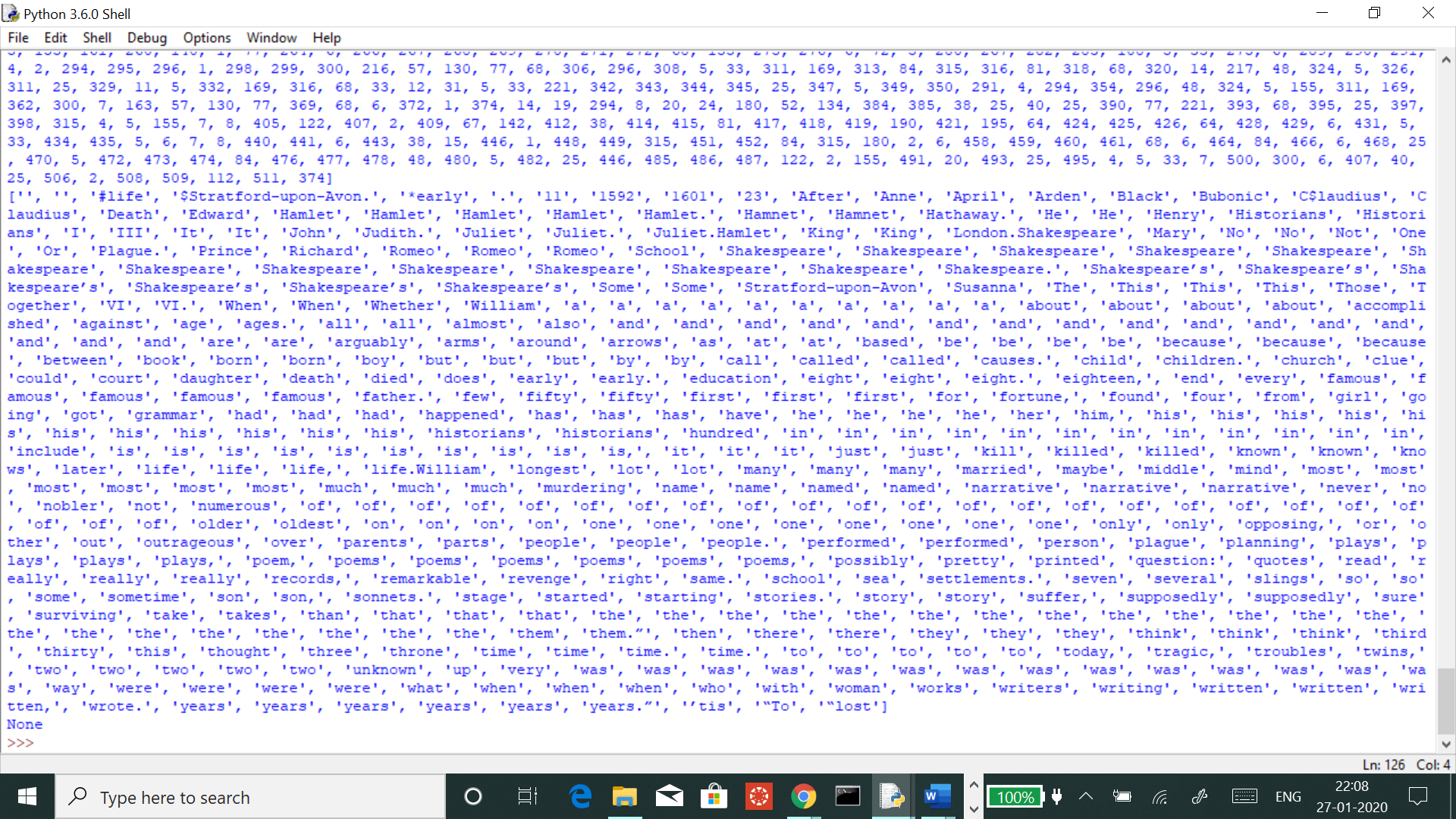
print(Convert(str1))

**Test cases:**

**Input list:**

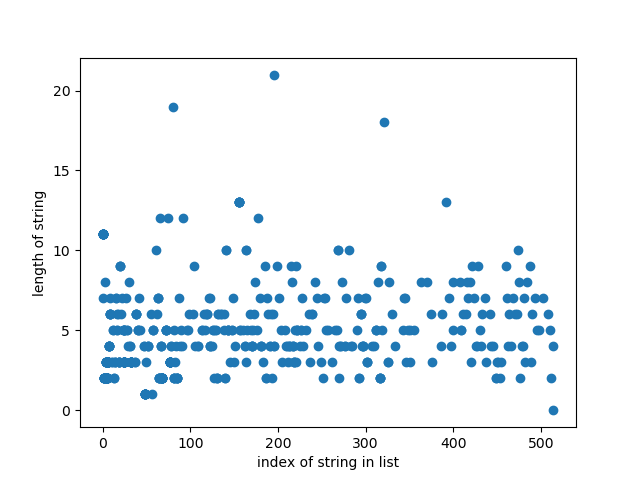


**Sorted list:**



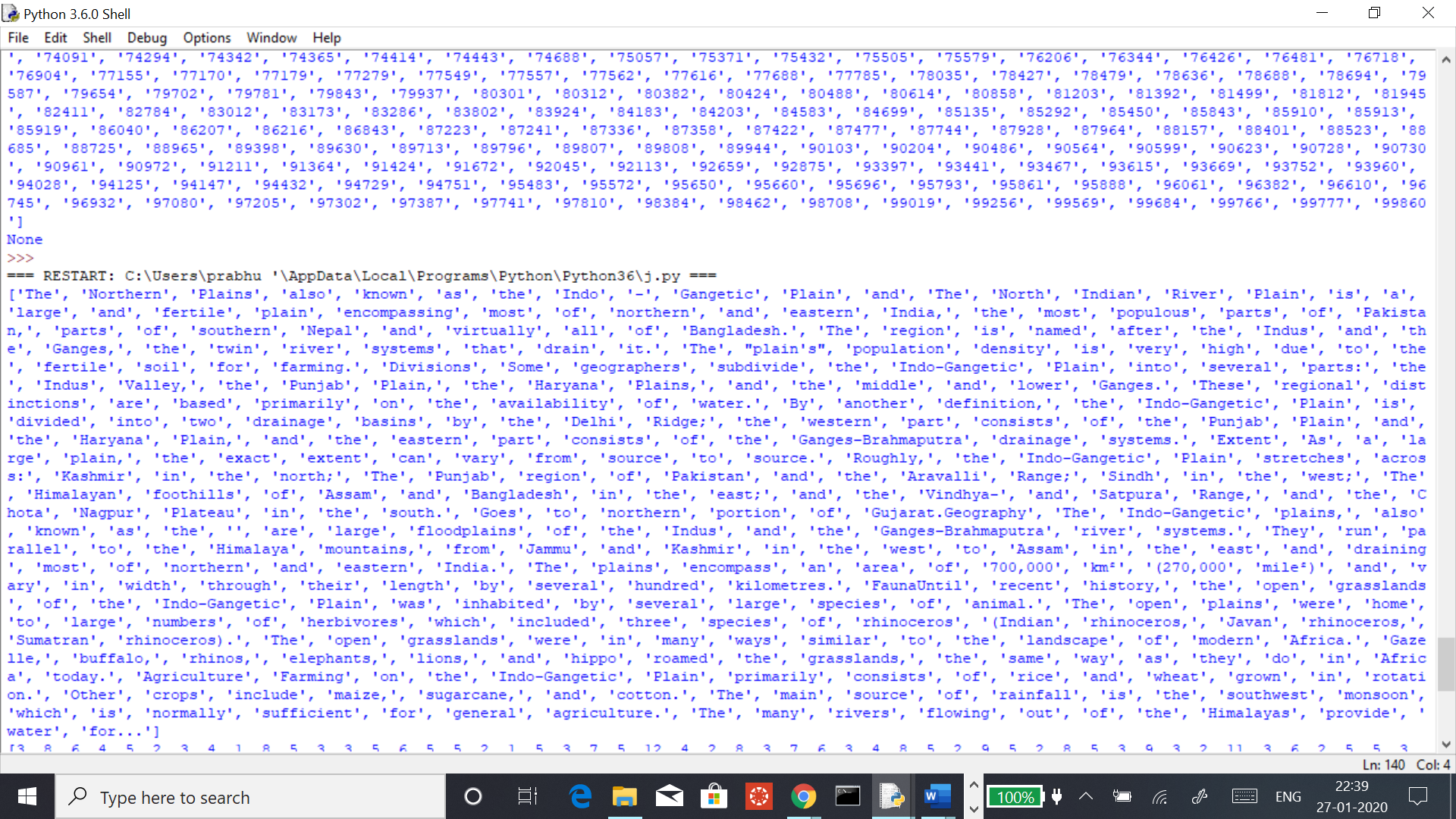
**Visualization output**:

Scatter plot showing “index of string in the list” vs “length of string”

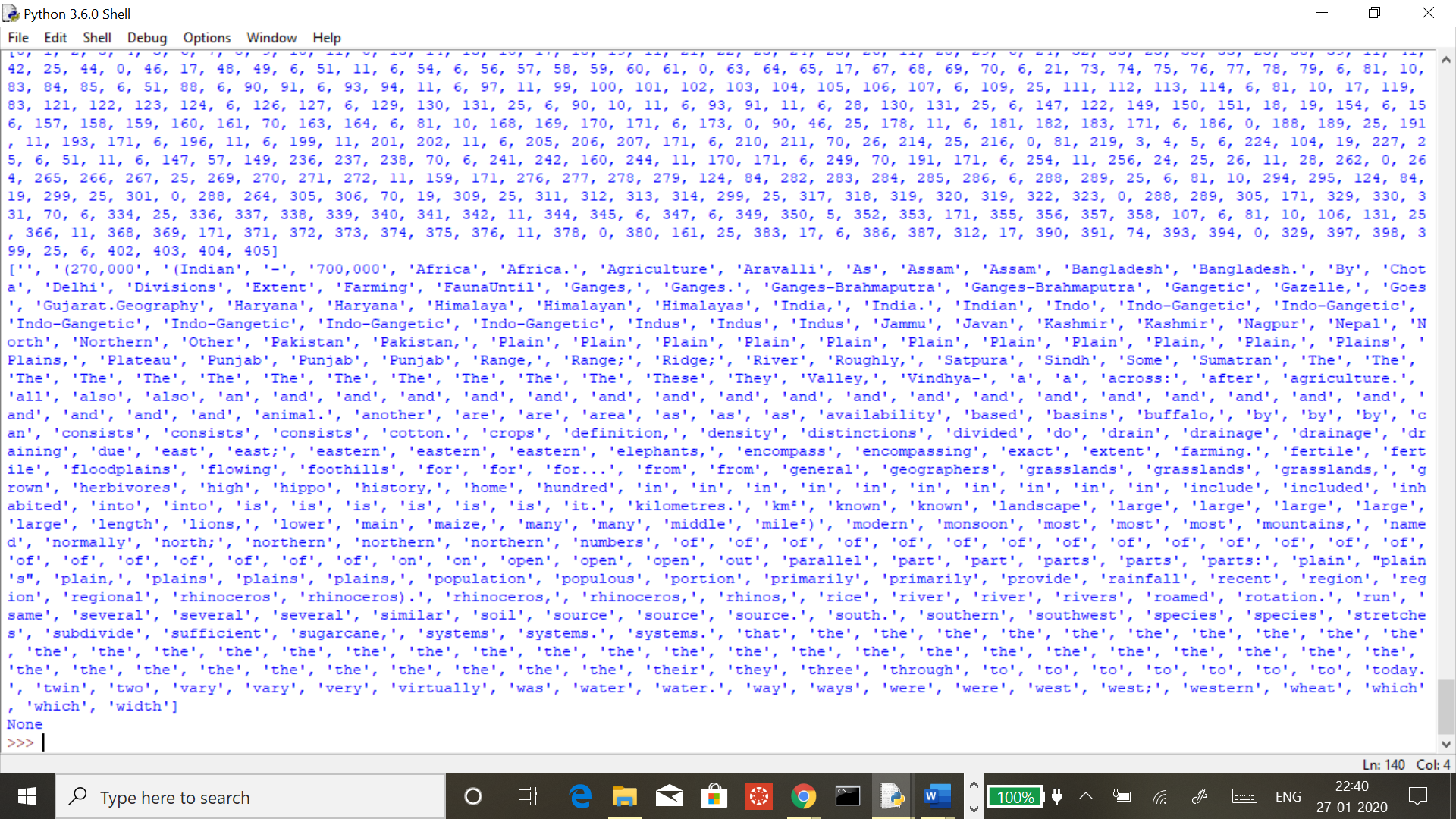


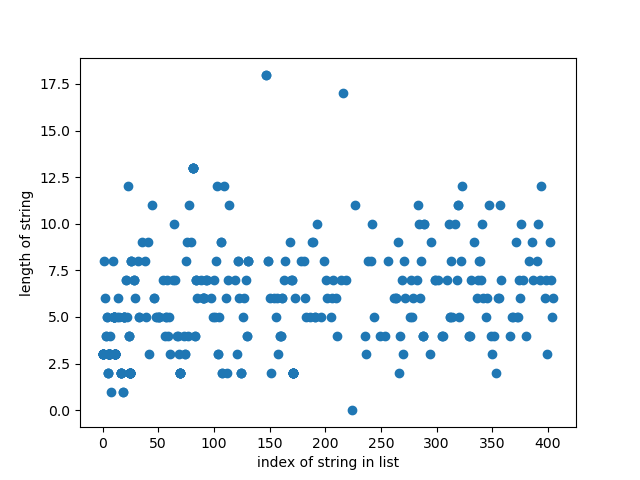
Test case 1:

Input:



Output:

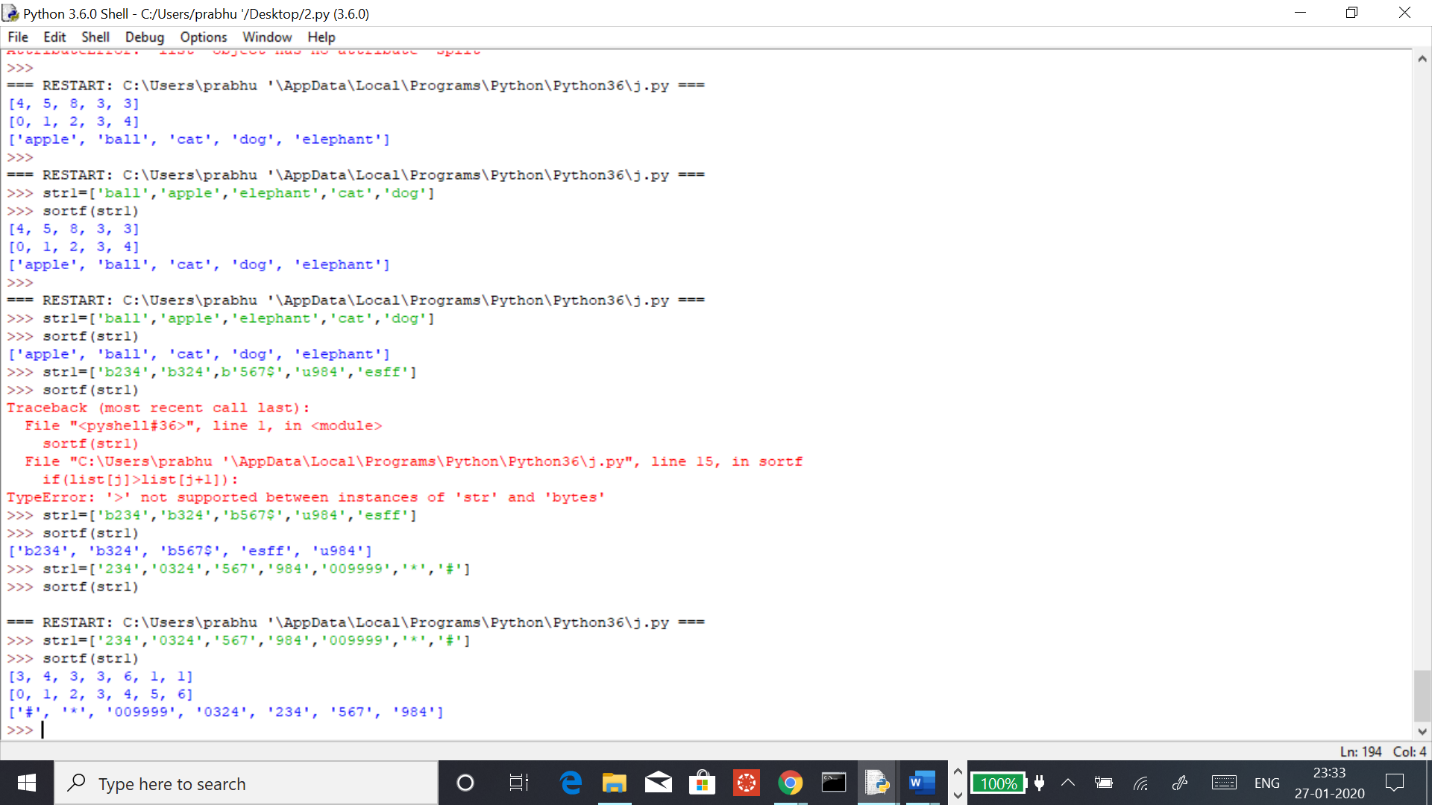




Test case 2:

Input list=[‘b234’,’b324’,’b567$’,’u984’,’esff’]

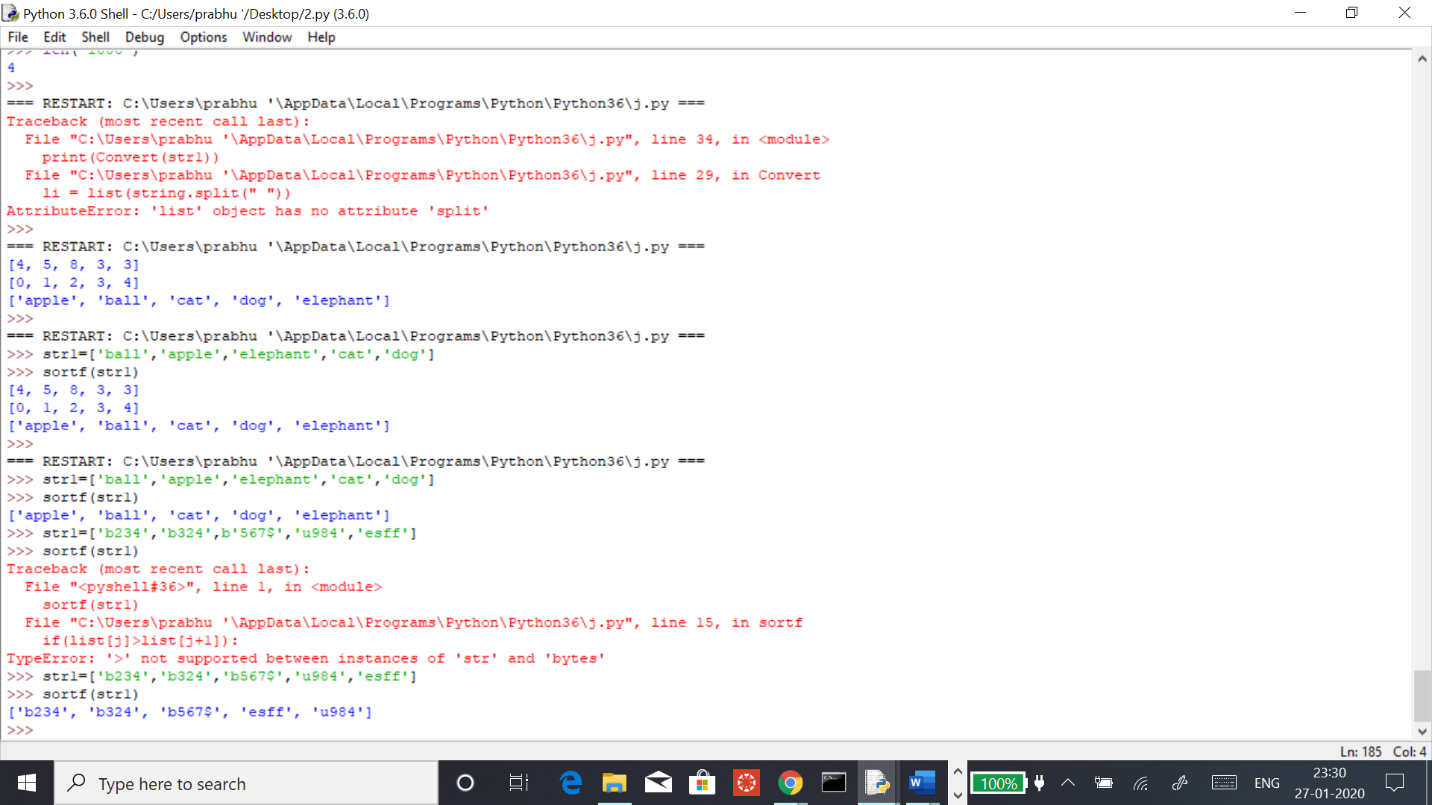
Output: [‘b234’,’b324’,’b567$’,’u984’,’esff’]



Test case3:

Input: ['ball','apple','elephant','cat','dog']

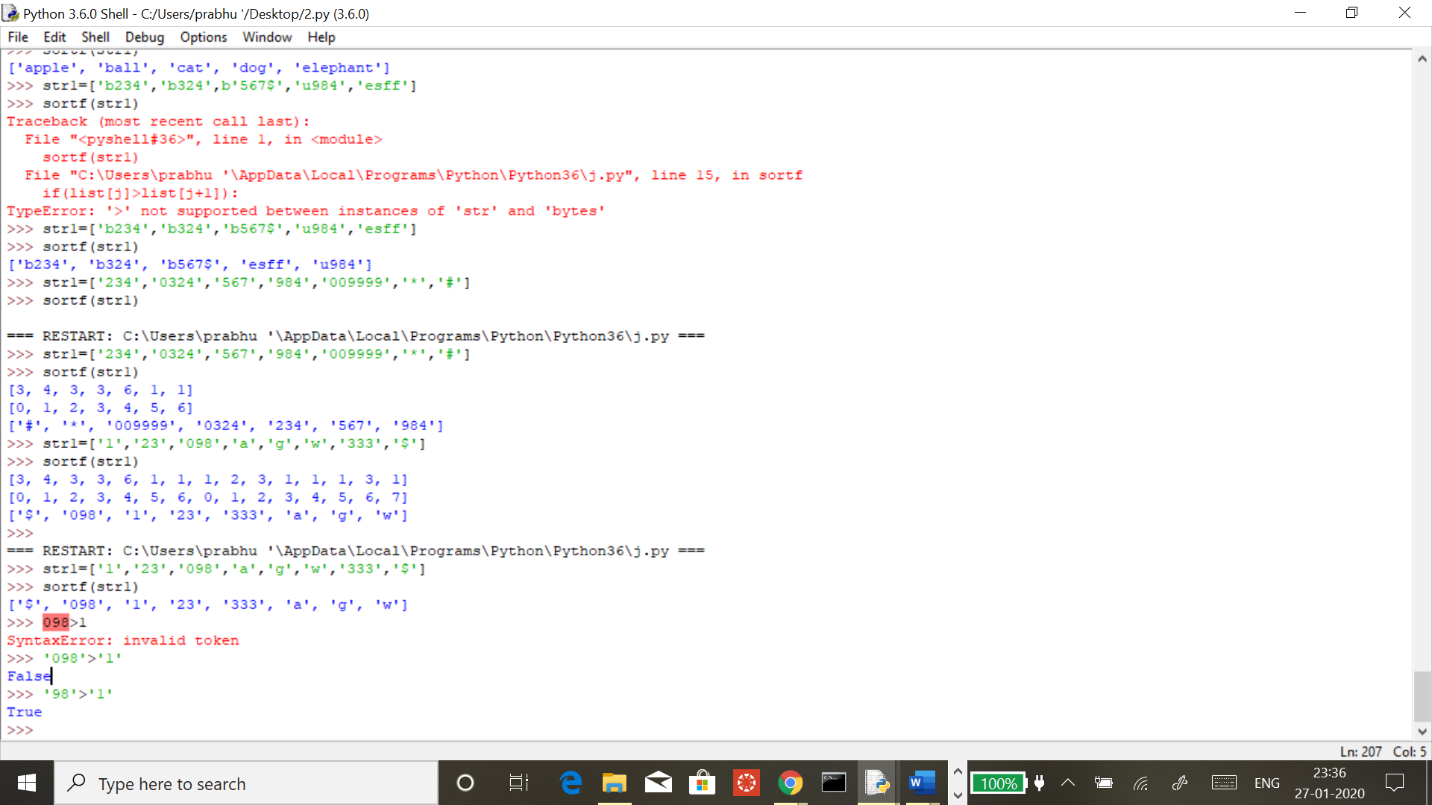
Output: ['apple', 'ball', 'cat', 'dog', 'elephant']



Test case 4:

Input string: ['1','23','098','a','g','w','333','$']

Output: ['$', '098', '1', '23', '333', 'a', 'g', 'w']



In the above result, 098 is said to be the smallest than 1 because, numbers are in the form of strings and

Which while comparing ASCII value is taken in to consideration.

Test case 5:

Input string = ['\*','^','$','&','@','!','#']

Output : ['!', '#', '$', '&', '\*', '@', '^']

