```python
# -*- coding: utf-8 -*-
"""PPA4.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1PnGRZK7uy6RU21aI-WUz00aVXEsYlCzI
"""

def is_sorted(lst):

    for i in range(len(lst) - 1):
        if lst[i] > lst[i + 1]:
            return False
    return True

my_list = [1, 3, 5, 7, 9]
print(is_sorted(my_list))

my_list = [1, 5, 3, 7, 9]
print(is_sorted(my_list))  #

def has_duplicates(lst):
    unique_elements = set()
    for item in lst:
        if item in unique_elements:
            return True
        unique_elements.add(item)
    return False
def remove_duplicates(lst):
    unique_elements = []
    for item in lst:
        if item not in unique_elements:
            unique_elements.append(item)
    return unique_elements
my_list = [1, 2, 3, 4, 5, 5, 6]
print(has_duplicates(my_list))

unique_list = remove_duplicates(my_list)
print(unique_list)

def read_words_from_file(filename):
    with open(filename, 'r') as file:
        words = file.read().splitlines()
    return words
def add_single_letter_words(word_list):
    word_list.extend(["I", "a", ""])

filename = "words.txt"
words_list = read_words_from_file(filename)
add_single_letter_words(words_list)
print(words_list)

def read_dictionary_from_user():
     dictionary = {}
    n = int(input("Enter the number of key-value pairs: "))
    for i in range(n):
        key = input("Enter key: ")
        value = input("Enter value: ")
        dictionary[key] = value
    return dictionary
def invert_dictionary(dictionary):
        inverted_dict = {}
    for key, value in dictionary.items():
        inverted_dict[value] = key
    return inverted_dict
user_dict = read_dictionary_from_user()
print("Original Dictionary:", user_dict)
inverted_dict = invert_dictionary(user_dict)
print("Inverted Dictionary:", inverted_dict)

def add_characters(word):
    return ','.join(word)
word = 'Apple'
modified_word = add_characters(word)
```

```
print(modified_word)

def words(sentence):
    sentence = ""
    is_start_of_word = True

    for char in sentence:
        if is_start_of_word:
            sentence += char.upper()
            is_start_of_word = False
        else:
            sentence += char.lower()

        if char == ' ':
            is_start_of_word = True

    return sentence

sentence = "this is a Sample sentence to Test the Function"
result = words(sentence)
print(result)

text=input("enter a STRING: ")
text=text.split()
word=input("enter the word to remove: ")
for string in text:
    if string ==word:
        text.remove(word)
text=''.join(text)
print("the text after removing the word is: ",text)

def convert(s):
 ch = list(s)
 for i in range(len(s)):
  if (i == 0 and ch[i] != ' ' or
     ch[i] != ' ' and
     ch[i - 1] == ' '):
   if (ch[i] >= 'a' and ch[i] <= 'z'):
    ch[i] = chr(ord(ch[i]) - ord('a') +
       ord('A'))
  elif (ch[i] >= 'A' and ch[i] <= 'Z'):

   ch[i] = chr(ord(ch[i]) + ord('a') -
      ord('A'))
  st = "".join(ch)

 return st;
if __name__=="__main__":
  s = "darshan raval vagmi sree"
  print(convert(s));

def capitals(strings):
    lis=strings.split()
    list1=[]
    for words in lis:
        word = word.capitalize()
        list1=list1+[word]
    my_string=''.join (str(word)for word in list1)

    return my_string
string="i love darshan raval"
result= capitals(string)
print(result)

text=input("enter the text")
print("the text is:",text)
newText=" "
for word in text.split():
    word.lower()
   # newText+=word.replace(word[0],word[0].upper())+ ""
    newText+=word.capitalize()+" "
print(newText)

def printTheArray(arr, n):
```

```
 for i in range(0, n):
  print(arr[i], end = " ")
 print()
def generateAllBinaryStrings(n, arr, i):
 if i == n:
  printTheArray(arr, n)
  return
  arr[i] = 0
 generateAllBinaryStrings(n, arr, i + 1)
 generateAllBinaryStrings(n, arr, i + 1)
if __name__ == "__main__":
 n = 4
 arr = [1] * n
 generateAllBinaryStrings(n, arr, 0)

def binary_strings(n,binary_strings):
    if len(binary_string)== n:
        print(binary_string)
    else:
       binary_strings(n,binary_string+"0")
       binary_strings(n,binary_string+"1")
n=3
binary_string=""
binary_strings(n,binary_string)

def factorial
```