Untitled

August 23, 2024

Data Structures in Python1. write a code to reverse string

[15]:
```
Name=input("Enter Your Name:")
print(Name[::-1])
```

```
Enter Your Name: vaibhav
```

```
vahbiav
```

2.Write a code to count a Vowel in string.

[25]:
```
def vowel_count(str):
    count=0
    vowel=set("aeiouAEIOU")
    for alphabet in str:

        if alphabet in vowel:
            count=count+1
    print("no of count:", count)
str=input("enter your string:")
vowel_count(str)
```

```
enter your string: dev
```

```
no of count: 1
```

[18]:
```
# Python3 code to count vowel in
# a string using set

# Function to count vowel
def vowel_count(str):

        # Initializing count variable to 0
        count = 0

        # Creating a set of vowels
        vowel = set("aeiouAEIOU")

        # Loop to traverse the alphabet
        # in the given string
```

```
        for alphabet in str:


                # If alphabet is present
                # in set vowel
                if alphabet in vowel:
                        count = count + 1


        print("No. of vowels :", count)

# Driver code
str = input("enter your string")

# Function Call
vowel_count(str)
```

```
enter your string vaibhav
No. of vowels : 3
```

[4]:
```
#3.check given string is palidrom or not
def ispalindrom(s):
    return s==s[::-1]
ans="malayalam"
ispalindrom(ans)
if ans:
    print("yes")
else:
    print("no")
```

```
yes
```

[5]:
```
#4.write a code to check given string are anagrams of each other.
def isanagram(s1,s2):
    if(sorted(s1)==sorted(s2)):
        print("the given string are anagram")
    else:
        print("The Given String Are Not Anagram")
s1="vaibhav"
s2="vaishnav"
isanagram(s1,s2)
```

```
The Given String Are Not Anagram
```

[8]:
```
#5.write a code to all occcurance of a given substring with in another string
# 1.starts with()
# 2.refindtier()
```

```
# 3.find()
# 4.reduce()

import re

test_str = "my name is "
test_sub = "i am in 12th class"

print("vaibhav is my name " + test_str)
print("I passed my 11th class " + test_sub)

# Finding all occurrences of test_sub in test_str
res=[i.start() for i in re.finditer(test_sub, test_str)]

print("The start indices of the substrings are : " + str(res))
```

```
vaibhav is my name my name is
I passed my 11th class i am in 12th class
The start indices of the substrings are : []
```

[1]:
```
#6. write a code to performe basic string compression using the counts of␣
 ↪repeated character?
s="abbacc"
s1={}
for i in s:
    if i in s1:
        s1[i]+=1
    else:
        s1[i]=1
print(s1)
```

```
{'a': 2, 'b': 2, 'c': 2}
```

[2]:
```
#7.Write a code to determine if a string has all uique characters.
s=input("Enter the strig:")
s1=set(s)
if len(s1)==len(s):
    print(True)
else:
    print(False)
```

```
Enter the strig: vaibhav
```

```
False
```

[4]:
```
#Q8.Write a code to convert a given string to uppercase or lowercase.
string="My strength IS mY FaMiLy"
print("to convert a given string to uppercase :" ,string.upper())
```

```
print("to convert a given string to lowercase : " ,string.lower())
```

```
to convert a given string to uppercase : MY STRENGTH IS MY FAMILY
to convert a given string to lowercase :  my strength is my family
```

[7]:
```
#9. Write a code to count the number of words in a string.
s="count the number"
count=0
word=False
for char in s:
    if char!=' ':
        if not word:
            count+=1
            word=True
        else:
            word=False
print(count)
```

```
7
```

[8]:
```
s="count the number"
print(len(s.split()))
```

```
3
```

[15]:
```
#Q10.Write a code to concatenate two string without using + operator
s1="hello"
s2="mikki"
print(f"{s1} {s2}")
```

```
hello mikki
```

[19]:
```
#Q11 Write a code to remove all occurrences  of a specific element from a list.
list1=[2,3,4,3,2,3,4,3,1,2,3,4,3]
l=[]
element=int(input("enter the specific number:"))
for i in list1:
    if i!= element:
        l.append(i)
print("after removing the specific element from list:",l)
```

```
enter the specific number: 123
after removing the specific element from list: [2, 3, 4, 3, 2, 3, 4, 3, 1, 2, 3,
4, 3]
```

[2]:
```
#Q12. Implement a code to find the second largest number in a given in the␣
 ↪given list of integer.
```

```
list1=[12,22,445,6787,978]
largest=0
second_largest=0;
for i in list1:
    if i>largest:
        second_largest=largest
        largest=i
    elif largest>i>second_largest:
        second_largest=i
print(second_largest)
```

978

[11]:
```
#13. Create a code to count the occurence of each element in a list and return␣
 ↪a dictionary with element as keys
#and their count as values.
list=[2,2,32,3,34,44,44,55,66,77]
dict={}
for i in list:
    if i in dict:
        dict[i]+=1
    else:
        dict[i]=1

print(dict)
```

{2: 2, 32: 1, 3: 1, 34: 1, 44: 2, 55: 1, 66: 1, 77: 1}

[16]:
```
#Q14. Write a code to reverse a list in-place without using any built in␣
 ↪reverse functions.
list=[2,2,32,3,34,44,44,55,66,7]
print(list[::-1])
```

[7, 66, 55, 44, 44, 34, 3, 32, 2, 2]

[19]:
```
#Q15.Implement a code to find and remove duplicates from a list while␣
 ↪perserving the original order of element.
list=[2,2,32,3,34,44,44,55,66,7]
list1=[]
list2=[]
for i in list:
    if i not in list1:
        list1.append(i)
    else:
        list2.append(i)
print("find the duplicate element:",list1)
```

```
print("after remove the duplicates element from list while perserving the␣
 ↪original order of element :",list2)
```

```
find the duplicate element: [2, 32, 3, 34, 44, 55, 66, 7]
after remove the duplicates element from list while perserving the original
order of element : [2, 44]
```

[23]:
```
#Q16.create a code given list is sorted(ascendig and discendig order )or not
list=[2,2,32,3,34,44,44,55,66,7]
list1=sorted(list)
if list==list1:
    print(True)
else:
    print(False)
```

```
False
```

[25]:
```
l=[2,2,32,3,34,44,44,55,66,7]
f=False
for i in range(len(l)-1):
     if i<i+1:
        f=True
        break
print(f)
```

```
True
```

[28]:
```
#Q17.Write a code to merge two sorted lists into a single sorted list.
l=[2,2,32,3,34,44,44,55,66,7]
l1=[433,435,55,66,7,8,99,00]
l3=l1+l

sorted(l3)
```

[28]:
```
[0, 2, 2, 3, 7, 7, 8, 32, 34, 44, 44, 55, 55, 66, 66, 99, 433, 435]
```

[7]:
```
#Q18. Implement a code to find the intersection of two given lists.
list1=[2,3,4,55,776,89]
list2=[2,3,4,65,676,89]
set1=set(list1)
set2=set(list2)
print("print list1",list1)
print("print list1",list2)
print("Intersectin Number",(set1&set2))
```

```
print list1 [2, 3, 4, 55, 776, 89]
print list1 [2, 3, 4, 65, 676, 89]
Intersectin Number {89, 2, 3, 4}
```

[1]:
```
#Q19. Create a code to find the union of two list without duplicates.
list1=[2,3,4,55,776,89]
list2=[2,3,4,65,676,89]
for i in list1:
    if i not in list2:
        list2.append(i)
print("union of list1 and list2:",list2)
```

```
union of list1 and list2: [2, 3, 4, 65, 676, 89, 55, 776]
```

[2]:
```
#Q20. Write a code to shuffle a give list randomly without using any built-in␣
 ↪shuffle fuctions.
import random
def shuffle_list(input_list):
    shuffled_list = input_list[:]
    n = len(shuffled_list)
    for i in range(n):
        j= random.randint(0,n-1)
        shuffled_list[i],shuffled_list[j]=shuffled_list[j],shuffled_list[i]
        return shuffled_list
my_list=[1,2,3,4,5,6,7,8]
shuffled=shuffle_list(my_list)
print(shuffled)
```

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

[4]:
```
#Q21. Write a code that takes two tuples as input and returns a new tuple␣
 ↪containing elements
#that are common to both  input tuple.
tuple1=(2,3,4,5,6)
tuple2=(3,4,2,6,7)
set1=set(tuple1)
set2=set(tuple2)
tuple3=tuple(set1&set2)
print("first tuple:",tuple1)
print("second tuple:",tuple2)
print("tuple containing elements that are common to both input tuple;",tuple3)
```

```
first tuple: (2, 3, 4, 5, 6)
second tuple: (3, 4, 2, 6, 7)
tuple containing elements that are common to both input tuple; (2, 3, 4, 6)
```

[ ]:
```
# Q22. Create a code that prompt the user to enter two sets of integers␣
 ↪seperated by commas Then,print the
#intersection of these two sets.
set1=set(map(int, input("enter the integer of set1 :").split(',')))
set2=set(map(int, input("enter the integer of set2:").split(',')))
print("set1 :",set1)
print("set2 :",set2)
print("intersection of two set :",set1&set2)
```

[ ]:
```
#Q23 Write a code to concatenate two tuples .The fuction should take two tuples␣
 ↪as input
#and retrun a new tuple containing elements
tuple1=("a","b","c","d")
tuple2=(1,2,3,4)
print("tuple first :",tuple1)
print("tuple second:",tuple2)
print("concatenate ttwo tuples : ",tuple1+tuple2)
```

[ ]:
```
#Q24 Develop a code that prompts the user to input two sets of strings .Then,␣
 ↪print the elements that are present
#in the first set but not in the second set.
set_first=set(map(str, input("enter the string in first set:").split(',')))
set_second=set(map(str,input("enter the string in second set:").split(',')))
print("First set:",set_first)
print("second set:",set_second)
print("whose element are present in the first set but not in second set :
 ↪",set_first-set_second)
```

[ ]:
```
#Q25 Create a code that takes a tuple and two integers as input .The function␣
 ↪should returan a new tuple containing elements
#from the original tuple within the specified range of indicies.
tuple1=(1,2,3,3,2,1,2,3,4,5,4,1,2,3,2,5,6,7,6,5,8,9)
index1=2
index2=5
list1=[]
for i in range(index1,index2):
    list1.append(f" index {i} : {tuple1[i]}")
new_tuple=tuple(list1)
print("original tuple within the specified range of indicies :",new_tuple)
```

[ ]:
```
#Q26 Write  code that prompts the user to input two sets of chracters .
 ↪Then,print the union of these two  sets.
set1=set(input("enter the character:"))
set2=set(input("enter the character :"))
print("unoin of two sets :",set1|set2)
```

[ ]:
```
#Q27. Develop a code that takes a tuple of integers as input.The function␣
 ↪should return the maximum and minimum values
#from the tuple using tuple unpacking.
def maxminnum(tuple2):
    print("maximum number :",max(tuple2),"\nminimum number :",min(tuple2))
tuple1=(33,22,11,66,78,54)
tuple2=tuple1
maxminnum(tuple2)
```

[ ]:
```
#Q28.Create a code that defines two sets of integers.Then, print the␣
 ↪union,intersection and differences of these two sets.
set1={2,3,4,5,34,22,11,45,67}
set2={3,4,5,8,9,67,89,43,47}
print("set1 : ",set1)
print("set2 : ",set2)
print("union of two sets : ",set1|set2)
print("intersections of two sets : ",set1&set2)
print("differences  of these two sets : " ,set1-set2)
```

[ ]:
```
#Q29. Write a code that takes a tuple and an element as input .The function␣
 ↪should return the count
#of occurrences of the given element in tuple.
tuple1=(3,4,5,6,7,3,4,5,4,3,7,6,4,3,2,3,4)
print("occurrences of given element :",tuple1.count(3))
```

[ ]:
```
#Q30. Develop a code that prompts the user to input two sets of strings.Then,␣
 ↪print the symmetric differencesof these
#two sets.
set1=set(map(str,input("enter the string of set first:").split(',')))
set2=set(map(str,input("enter the string of set second:").split(',')))
print("first set ",set1)
print("second set ",set2)
print("symmetric differences of these two sets : ", set1^set2)
```

[ ]:
```
#Q31 Write a code that takes a list of words as input and returns a dicionary␣
 ↪where the keys are unique words and the
#values are the frequecies of those words in the input list.
list1=[2,3,4,5,6,7,8,7,6,5,4,3,2]
dict1={}
for i in list1:
    if i in dict1:
        dict1[i]+=1
    else:
        dict1[i]=1
print(dict1)
```

[ ]:
```
#Q32. Write a code that takes two dictionaries as input and merges them into a single dictionary .If there are coomon keys,
# the values should be added together.
dict1={"a":5,"b":3,"c":3}
dict2={"c":7,"d":4,"b":4}
merge_dict=dict1.copy()
for key,value in dict1.items():
    if key in dict2:
        merge_dict[key]+=value
    else:
       merge_dict[key]=value
print("merges them into a single dictionary :",merge_dict)
```

[ ]:
```
#Q33. Write a code to access a value in a nested dictionary. The function should take the dictionary and a list of keys
#as input and return the corresponding value.If any of the keys do not exit in the dictionary,the function should
#return None
people={1:{"name" : "Vaibhav","age":21, "class":12}}
print("information :
 ",people[1]['name'],"\n",people[1]['age'],"\n",people[1]['class'])
```

[ ]:
```
#Q34. Write a code that takes a dictionary as input and returns a sorte version of it based on the values .You con choose whether to sort in ascending
#or descending order.
dict1={"a":2,"e":3,"c":4,"d":4}
sorted(dict1.values())
```

[ ]:
```
#Q35. Write a code that inverts a dictionary ,swapping keys and values. Ensure that the inverted dictionary correctly handles cases
#where multiple keys have
original_dict={"a":34,"c":4,"d":5}
inverted_dict={}
for key,value in original_dict.items():
    if value in inverted_dict:
        inverted_dict[value].append(key)
    else:
        inverted_dict[value]=[key]
print(inverted_dict)
```

[ ]:
```

```

[ ]:
```

```

[ ]: