

CS23336-Introduction to Python Programming

Started on Monday, 21 October 2024, 9:00 PM

State Finished

Completed on Monday, 21 October 2024, 9:50 PM

Time taken 50 mins 19 secs

Marks 10.00/10.00

Grade **100.00** out of 100.00

Question 1

Correct
Mark 1.00 out of 1.00
 Flag question

Question text

Check if a set is a subset of another set.

Example:

Sample Input1:

mango apple

mango orange

mango

output1:

yes

set3 is subset of set1 and set2

input2:

mango orange

banana orange

grapes

output2:

no



For example:

| Test | Input | Result |
|------|---|--|
| 1 | mango apple mango orange mango | yes set3 is subset of set1 and set2 |
| 2 | mango orange banana orange grapes | No |

Answer:(penalty regime: 0 %)

```
1 s1=set(input().strip().split())
2 s2=set(input().strip().split())
3 s3=set(input().strip().split())
4 if s3.issubset(s1) and s3.issubset(s2):
5
```

```
6     print("yes")
7 else:
8     print('No')
```

Feedback

| Test | Input | Expected | Got |
|------|---|--|--|
| 1 | mango apple mango orange mango | yes set3 is subset of set1 and set2 | yes set3 is subset of set1 and set2 |
| 2 | mango orange banana orange grapes | No | No |

Passed all tests!


Correct

Marks for this submission: 1.00/1.00.

Question 2

Correct

Mark 1.00 out of 1.00

 Flag question

Question text

There is a malfunctioning keyboard where some letter keys do not work. All other keys on the keyboard work properly.

Given a string text of words separated by a single space (no leading or trailing spaces) and a string brokenLetters of all distinct letter keys that are broken, return the number of words in text you can fully type using this keyboard.

Example 1:

Input: text = "hello world", brokenLetters = "ad"

Output:

1

Explanation: We cannot type "world" because the 'd' key is broken.

For example:

| Input | Result |
|-------------------|--------|
| hello world ad | 1 |

Faculty Upskilling in Python Programming 2

ak

Answer:(penalty regime: 0 %)

```
1 def func(a:str,b:str)->int:
2     a=a.lower()
3
```

```
4     w=b.ispowset()
5     b1=set(b)
6     count=0
7     for i in w:
8         if not set(i)&b1:
9             count+=1
10    return count
11
12    a=input()
13    b=input()
14    print(func(a,b))
```

Feedback


| Input | Expected Got | |
|--|--------------|---|
| hello world ad | 1 | 1 |
| Welcome to REC e | 1 | 1 |
| Faculty Upskilling in Python Programming ak | 2 | 2 |

Passed all tests!

Correct
Marks for this submission: 1.00/1.00.

Question 3

Correct
Mark 1.00 out of 1.00

 Flag question

Question text

Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive. There is only **one repeated number** in nums, return *this repeated number*. Solve the problem using set.

Example 1:

Input: nums = [1,3,4,2,2]
Output: 2

Example 2:

Input: nums = [3,1,3,4,2]
Output: 3

For example:

| Input | Result |
|-------------|--------|
| 1 3 4 4 2 4 | |

Answer:(penalty regime: 0 %)

```
1 a=input()
2 a=tuple(a)
3 n=tuple(i for i in a if i.strip())
4 b=set(a)
```

```

5- for i in b:
6-     if n.count(i)>=2:
7-         print(i)
8-         break

```

Feedback


| Input | Expected Got |
|-------------------|--------------|
| 1 3 4 4 2 | 4 |
| 1 2 2 3 4 5 6 7 2 | 2 |

Passed all tests!

Correct
Marks for this submission: 1.00/1.00.

Question 4

Correct
Mark 1.00 out of 1.00

 Flag question

Question text

Given a tuple and a positive integer k, the task is to find the count of distinct pairs in the tuple whose sum is equal to **K**.

Examples:

Input: t = (5, 6, 5, 7, 7, 8), K = 13

Output: 2

Explanation:

Pairs with sum K(= 13) are {(5, 8), (6, 7), (6, 7)}.

Therefore, distinct pairs with sum K(= 13) are { (5, 8), (6, 7) }.

Therefore, the required output is 2.

For example:

Input **Result**

1,2,1,2,5 1

3

1,2 0

0

Answer:(penalty regime: 0 %)

```

1  a=input()
2  b=int(input())
3  a=set(a)
4  a.remove(',')
5  a=tuple(a)
6  res=0
7  for i in a:
8      for j in range(a.index(i),len(a)):
9          if int(i)+int(a[j])==b:

```

```
10
11 print(res)    res+=1
```

Feedback

| Input | Expected | Got |
|-------------------|----------|-----|
| 5,6,5,7,7,8 13 | 2 | 2 |
| 1,2,1,2,5 3 | 1 | 1 |
| 1,2 0 | 0 | 0 |

Passed all tests!


Correct

Marks for this submission: 1.00/1.00.

Question 5

Correct

Mark 1.00 out of 1.00

 Flag question

Question text

You are given an integer tuple `nums` containing distinct numbers. Your task is to perform a sequence of operations on this tuple until it becomes empty. The operations are defined as follows:

- 1. If the first element of the tuple has the smallest value in the entire tuple, remove it.
- 2. Otherwise, move the first element to the end of the tuple.

You need to return an integer denoting the number of operations required to make the tuple empty.

Constraints

- The input tuple `nums` contains distinct integers.
- The operations must be performed using tuples and sets to maintain immutability and efficiency.
- Your function should accept the tuple `nums` as input and return the total number of operations as an integer.

Example:

Input: `nums = (3, 4, -1)`

Output: 5

Explanation:

Operation 1: `[3, 4, -1]` -> First element is not the smallest, move to the end -> `[4, -1, 3]`

Operation 2: `[4, -1, 3]` -> First element is not the smallest, move to the end -> `[-1, 3, 4]`

Operation 3: `[-1, 3, 4]` -> First element is the smallest, remove it -> `[3, 4]`

Operation 4: `[3, 4]` -> First element is the smallest, remove it -> `[4]`

Operation 5: `[4]` -> First element is the smallest, remove it -> `[]`

Total operations: 5

For example:

| Test | Result |
|--|--------|
| <code>print(count_operations((3, 4, -1)))</code> | 5 |

Answer:(penalty regime: 0 %)

Reset answer

```
1 def count_operations(nums: tuple) -> int:
2     # Your implementation here
3     op=0
4     nums=list(nums)
5     while len(nums):
6         if nums[0]==min(nums):
7             nums.remove(nums[0])
8             op+=1
9         else:
10            t=nums[0]
11            nums.remove(t)
12            nums.append(t)
13            op+=1
14    return op
```


Feedback

| Test | Expected Got | |
|--|--------------|----|
| print(count_operations((3, 4, -1))) | 5 | 5 |
| print(count_operations((1, 2, 3, 4, 5))) | 5 | 5 |
| print(count_operations((5, 4, 3, 2, 1))) | 15 | 15 |
| print(count_operations((42,))) | 1 | 1 |
| print(count_operations((-2, 3, -5, 4, 1))) | 11 | 11 |

Passed all tests!

Correct
Marks for this submission: 1.00/1.00.

Question 6

Correct
Mark 1.00 out of 1.00
 Flag question

Question text

The **DNA sequence** is composed of a series of nucleotides abbreviated as 'A', 'C', 'G', and 'T'.

- For example, "ACGAATTCCG" is a **DNA sequence**.

When studying **DNA**, it is useful to identify repeated sequences within the DNA.

Given a string *s* that represents a **DNA sequence**, return all the **10-letter-long** sequences (substrings) that occur more than once in a DNA molecule. You may return the answer in **any order**.

Example 1:

Input: *s* = "AAAAACCCCCAAAAACCCCCAAAAGGGTTT"
Output: ["AAAAACCCCC", "CCCCCAAAAA"]

Example 2:

Input: *s* = "AAAAAAAAAAAA"
Output: ["AAAAAAAAAA"]

For example:

| Input | Result |
|---------------------------------|--------------------------|
| AAAAACCCCCAAAAACCCCCAAAAAGGGTTT | AAAAACCCCC CCCCCAAAAA |

Answer:(penalty regime: 0 %)


```
1 s=input()
2 n=set()
3 p=set()
4 for i in range(len(s)-9):
5     c=s[i:i+10]
6     if c in n:
7         p.add(c)
8     else:
9         n.add(c)
10 s=list(p)
11 for i in range(len(s)-1,-1,-1):
12     print(s[i])
```

Feedback

| Input | Expected | Got |
|---------------------------------|--------------------------|--------------------------|
| AAAAACCCCCAAAAACCCCCAAAAAGGGTTT | AAAAACCCCC CCCCCAAAAA | AAAAACCCCC CCCCCAAAAA |
| AAAAAAAAAAAAA | AAAAAAAAAAAA | AAAAAAAAAAAA |

Passed all tests!
Correct
Marks for this submission: 1.00/1.00.

Question 7

Correct
Mark 1.00 out of 1.00
 Flag question

Question text

Program to print all the distinct elements in an array. Distinct elements are nothing but the unique (non-duplicate) elements present in the given array.

Input Format:
First line take an Integer input from stdin which is array length n.
Second line take n Integers which is inputs of array.
Output Format:
Print the Distinct Elements in Array in single line which is space Separated
Example Input:

1 2 2 3 4

Output:

1 2 3 4

Example Input:

6

1 1 2 2 3 3

Output:

1 2 3

For example:

Input Result

5
1
2
2 1 2 3 4
3
4

Answer:(penalty regime: 0 %)

```
1 a=int(input())
2 b=[]
3 for i in range(a):
4     f=int(input())
5     b.append(f)
6 c=set(b)
7 b=tuple(c)
8 print(*b)
```

Feedback

Input Expected Got

5
1
2 1 2 3 4 1 2 3 4
2
3
4

6
1
1
2 1 2 3 1 2 3
2
3
3

5
11
22
11 11 22 11 22


```
22
11


10
1
2
3
4
5      1 2 3 4 5   1 2 3 4 5
1
2
3
4
5
```

Passed all tests!

Correct
Marks for this submission: 1.00/1.00.

Question 8

Correct
Mark 1.00 out of 1.00

 Flag question

Question text

Write a program to eliminate the common elements in the given 2 arrays and print only the non-repeating elements and the total number of such non-repeating elements.

Input Format:

The first line contains space-separated values, denoting the size of the two arrays in integer format respectively.

The next two lines contain the space-separated integer arrays to be compared.

[Sample](#) Input:

```
5 4
1 2 8 6 5
2 6 8 10
```

[Sample](#) Output:

```
1 5 10
3
```

[Sample](#) Input:

```
5 5
1 2 3 4 5
1 2 3 4 5
```

[Sample](#) Output:

NO SUCH ELEMENTS

For example:

| Input | Result |
|------------------------------|------------------|
| 5 4 1 2 8 6 5 2 6 8 10 | 1 5 10 3 |
| 5 5 1 2 3 4 5 | NO SUCH ELEMENTS |

1 2 3 4 5

Answer:(penalty regime: 0 %)

```
1 import re
2 a=input()
3 b=input()
4 c=input()
5 b=(re.findall(r'[0-9]+' ,b))
6 c=(re.findall(r'[0-9]+' ,c))
7 b=set(b)
8 c=set(c)
9 d=b^c
10 b={0}
11 for i in d:
12     b.add(int(i))
13 b.discard(0)
14 b=list(b)
15 b.sort()
16 if len(b)==0:
17     print("NO SUCH ELEMENTS")
18 else:
19     print(*b)
20     print(len(b))
```

Feedback

| Input | Expected | Got |
|-------------------------------|------------------|------------------|
| 5 4 1 2 8 6 5 2 6 8 10 | 1 5 10 3 | 1 5 10 3 |
| 3 3 10 10 10 10 11 12 | 11 12 2 | 11 12 2 |
| 5 5 1 2 3 4 5 1 2 3 4 5 | NO SUCH ELEMENTS | NO SUCH ELEMENTS |

Passed all tests!


Correct

Marks for this submission: 1.00/1.00.

Question 9

Correct

Mark 1.00 out of 1.00

 Flag question

Question text

Given an array of strings `words`, return *the words that can be typed using letters of the alphabet on only one row of American keyboard like the image below.*

In the **American keyboard**:

- the first row consists of the characters "qwertyuiop",
- the second row consists of the characters "asdfghjkl", and
- the third row consists of the characters "zxcvbnm".



Example 1:

Input: words = ["Hello","Alaska","Dad","Peace"]
Output: ["Alaska","Dad"]

Example 2:

Input: words = ["omk"]
Output: []

Example 3:

Input: words = ["adsdf","sfd"]
Output: ["adsdf","sfd"]

For example:

Input Result

```
4
Hello
Alaska Alaska
Dad Dad
Peace
```

```
2
adsfd adsfd
afd afd
```

Answer:(penalty regime: 0 %)

```
1 a=int(input())
2 b=()
3 b=list(b)
4 r1={'q','w','e','r','t','y','u','i','o','p'}
5 r2={'a','s','d','f','g','h','j','k','l'}
6 r3={'z','x','c','v','b','n','m'}
7 for i in range(a):
8     c=input()
9     f=set(c.lower())
10    if f.issubset(r1):
11        b.append(c)
12    elif f.issubset(r2):
13        b.append(c)
14    elif f.issubset(r3):
15        b.append(c)
16 if len(b)!=0:
17     for i in b:
18         print(i)
19 else:
20     print("No words")
21
```

Feedback

Input Expected Got

| | | |
|--------|--------|--------|
| 4 | | |
| Hello | | |
| Alaska | Alaska | Alaska |
| Alaska | Dad | Dad |
| Dad | | |
| Peace | | |

| | | |
|-----|----------|----------|
| 1 | | |
| omk | No words | No words |

| | | |
|-------|-------|-------|
| 2 | | |
| adsfd | adsfd | adsfd |
| afd | afd | afd |

Passed all tests!

Correct
Marks for this submission: 1.00/1.00.

Question 10

Correct
Mark 1.00 out of 1.00

Question text

Coders here is a simple task for you, Given string str. Your task is to check whether it is a binary string or not by using python set.

Examples:

Input: str = "01010101010"

Output: Yes

Input: str = "REC101"

Output: No

For example:

| Input | Result |
|-------|--------|
|-------|--------|

| | |
|-------------|-----|
| 01010101010 | Yes |
|-------------|-----|

| | |
|--------------|----|
| 010101 10101 | No |
|--------------|----|

Answer:(penalty regime: 0 %)

```
1 a=input()
2 b=set(a)
3 c={'1','0'}
4 if c==b:
5     print("Yes")
6 else:
7     print("No")
```

Feedback

| Input | Expected Got |
|-------|--------------|
|-------|--------------|

| | |
|-------------|-----|
| 01010101010 | Yes |
|-------------|-----|

| | |
|--------|----|
| REC123 | No |
|--------|----|

| | |
|--------------|----|
| 010101 10101 | No |
|--------------|----|

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

[Finish review](#)

[Skip Quiz navigation](#)

Quiz navigation

[Question 1 This page](#) [Question 2 This page](#) [Question 3 This page](#) [Question 4 This page](#) [Question 5 This page](#) [Question 6](#)

