

CS23336-Introduction to Python Programming

Quiz navigation

1

2

3

4

5

6

7

8

9

10

Show one page at a time

Finish review

Started on	Thursday, 7 November 2024, 10:20 PM
State	Finished
Completed on	Sunday, 10 November 2024, 11:26 PM
Time taken	3 days
Marks	10.00/10.00
Grade	100.00 out of 100.00

Question 1

Correct

Mark 100 out of 1.00

Flag question

The problem is that we want to reverse a array in $O(N)$ linear time complexity and we want the algorithm to be in-place as well!

For example: input is [1,2,3,4,5] then the output is [5,4,3,2,1]

Input

5

1 2 3 4 5

Output

5 4 3 2 1

For example:

Input	Result
5 1 2 3 4 5	5 4 3 2 1

Answer: (penalty regime: 0 %)

```
1 n=int(input())
2 arr=list(map(int,input().split()))
3 l,r=0,n-1
4 while l<=r:
```

```
1 n=int(input())
2 arr=list(map(int,input().split()))
3 l,r=0,n-1
4 while l<=r:
5     arr[l],arr[r]=arr[r],arr[l]
6     l+=1
7     r-=1
8 print(' '.join(map(str,arr)))
```

	Input	Expected	Got	
✓	5 1 2 3 4 5	5 4 3 2 1	5 4 3 2 1	✓
✓	10 0 2 4 6 8 1 3 5 7 9	9 7 5 3 1 8 6 4 2 0	9 7 5 3 1 8 6 4 2 0	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Correct

Flag question

Print the *sorted array*.

Input:

Explanation

Input:

2 3 1
Output

Explanation

Input:

Input:

-1 1 -
Output

Constraints:

- $1 \leq m$

WEEK-11-CODING-Bubble and Merge Sort Attempt review - Google Chrome

Not secure 118.185.187.137/moodle/mod/quiz/review.php?attempt=162616&cmid=1395

RAJALAKSHMI PALEPOTUNGUNGA REC-OCATS-1

Constraints:

- $1 \leq \text{nums.length} \leq 100$
- $-100 \leq \text{nums}[i] \leq 100$

For example:

Input	Result
6 1 1 2 2 2 3	3 1 1 2 2 2
5 2 3 1 3 2	1 3 3 2 2

Answer: (penalty regime: 0 %)

```
1 def fun(arr):
2     f={}
3     for n in arr:
4         if n in f:
5             f[n]+=1
6         else:
7             f[n]=1
8     a=sorted(arr,key=lambda x: (f[x],-x))
9     print(a)
10 n=int(input())
11 arr=list(map(int,input().split()))
12 fun(arr)
```

30°C Haze 14:14 11-11-2024

	Input	Expected	Got	
✓	6 1 1 2 2 2 3	3 1 1 2 2 2	3 1 1 2 2 2	✓
✓	5 2 3 1 3 2	1 3 3 2 2	1 3 3 2 2	✓
✓	9 -1 1 -6 4 5 -6 1 4 1	5 -1 4 4 -6 -6 1 1 1	5 -1 4 4 -6 -6 1 1 1	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 3
Correct
Mark 1.00 out of 1.00
Flag question

Given an array of integers `arr`, replace each element with its rank.

The rank represents how large the element is. The rank has the following rules:

- Rank is an integer starting from 1.
- The larger the element, the larger the rank. If two elements are equal, their rank must be the same.
- Rank should be as small as possible.

Example 1:

Input: `arr = [40,10,20,30]`

Output: `[4,1,2,3]`

Explanation: 40 is the largest element. 10 is the smallest. 20 is the second smallest. 30 is the third smallest.

Example 2:

Input: `arr = [100,100,100]`

Output: `[1,1,1]`

Explanation: Same elements share the same rank.

Example 3:
Input: arr = [37,12,28,9,100,56,80,5,12]
Output: [5,3,4,2,8,6,7,1,3]

- Constraints:**
- $0 \leq \text{arr.length} \leq 10^5$
 - $-10^9 \leq \text{arr}[i] \leq 10^9$

For example:

Test	Result
print(arrayRankTransform([40,10,20,30]))	[4, 1, 2, 3]

Answer: (penalty regime: 0 %)

Reset answer

```
1 def arrayRankTransform(arr):
2     s=sorted(set(arr))
3     r={value:i+1 for i,value in enumerate(s)}
4     return [r[x] for x in arr]
5
```

	Test	Expected	Got	
✓	print(arrayRankTransform([40,10,20,30]))	[4, 1, 2, 3]	[4, 1, 2, 3]	✓
✓	print(arrayRankTransform([100,100,100]))	[1, 1, 1]	[1, 1, 1]	✓
✓	print(arrayRankTransform([37,12,28,9,100,56,00,5,12]))	[5, 3, 4, 2, 8, 6, 7, 1, 3]	[5, 3, 4, 2, 8, 6, 7, 1, 3]	✓

Passed all tests! ✓

Correct
Marks for this submission: 1.00/1.00.

Question 4
Correct
Mark 1.00 out of 1.00
Flag question

Write a Python program to sort a list of elements using the merge sort algorithm.

For example:

Input	Result
5 6 5 4 3 8	3 4 5 6 8

Answer: (penalty regime: 0 %)

```
1 def merge(arr,l,m,r):
2     l1=arr[l:m+1]
3     r1=arr[m+1:r+1]
4     i=0
5     j=0
6     k=l
7     while i<len(l1) and j<len(r1):
8         if l1[i]<=r1[j]:
9             arr[k]=l1[i]
10            i+=1
11        else:
12            arr[k]=r1[j]
```

```
13         j+=1
14         k+=1
15     while i<len(l1):
16         arr[k]=l1[i]
17         i+=1
18         k+=1
19     while j<len(r1):
20         arr[k]=r1[j]
21         j+=1
22         k+=1
23 def merge1(arr,l,r):
24     if l<r:
25         m=(l+r)//2
26         merge1(arr,l,m)
27         merge1(arr,m+1,r)
28         merge(arr,l,m,r)
29 n=int(input())
30 arr=list(map(int,input().split()))
31 merge1(arr,0,len(arr)-1)
32 print(" ".join(map(str,arr)))
```

	Input	Expected	Got	
✓	5 6 5 4 3 8	3 4 5 6 8	3 4 5 6 8	✓
✓	9 14 46 43 27 57 41 45 21 78	14 21 27 41 43 45 46 57 78	14 21 27 41 43 45 46 57 78	✓
✓	4 86 43 23 49	23 43 49 86	23 43 49 86	✓

Passed all tests! ✓

Correct
Marks for this submission: 1.00/1.00.

WEEK-11-CODING-Bubble and Merge Sort Attempt review - Google Chrome

Not secure 118.185.187.137/moodle/mod/quiz/review.php?attempt=162616&cmid=1395

RAJALAKSHMI PAPERBOUNCE REC-OCATS-1

Question 5
Correct
Mark 1.00 out of 1.00
Flag question

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order. You read an list of numbers. You need to arrange the elements in ascending order and print the result. The sorting should be done using bubble sort.

Input Format: The first line reads the number of elements in the array. The second line reads the array elements one by one.

Output Format: The output should be a sorted list.

For example:

Input	Result
6 3 4 8 7 1 2	1 2 3 4 7 8
5 4 5 2 3 1	1 2 3 4 5

Answer: (penalty regime: 0 %)

```
1 def bubble(arr):  
2     n=len(arr)  
3     for i in range(n):  
4         for j in range(0,n-i-1):  
5             if arr[j]>arr[j+1]:  
6                 arr[j],arr[j+1]=arr[j+1],arr[j]  
7 n=int(input())  
8 arr=list(map(int,input().split()))  
9 bubble(arr)  
10 print(" ".join(map(str,arr)))
```

Type here to search

	Input	Expected	Got	
✓	6 3 4 8 7 1 2	1 2 3 4 7 8	1 2 3 4 7 8	✓
✓	6 9 13 1 3 4 6	1 3 4 6 9 13	1 3 4 6 9 13	✓
✓	5 4 5 2 3 1	1 2 3 4 5	1 2 3 4 5	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 6

Correct

Mark 1.00 out of 1.00

Flag question

Objective:

Develop a Python program to find the k-th maximum value in a given list of integers. The program should handle various edge cases, including lists with duplicate values, empty lists, and invalid values of k. The k-th maximum value refers to the k-th largest distinct element in the list.

Background:

Finding the k-th maximum value in a list is a common problem in computer science, often encountered in fields like data analysis, competitive programming, and software development. This problem requires an understanding of sorting algorithms, data structures, and efficient problem-solving techniques. By solving this problem, one gains insights into how to handle large datasets and optimize performance in practical applications.

Problem Description:

Given a list of integers, the task is to determine the k-th maximum value in the list. The program should meet the following requirements:

1. Input:

- A list of integers, which can contain both positive and negative values.
- An integer k, representing the position of the maximum value to find.

2. Output:

- The k-th maximum value in the list.
- If k is greater than the number of distinct elements in the list or if the list is empty, the program should return an appropriate message indicating the

error.

Constraints:

- The list may contain duplicate values.
- The value of k should be a positive integer.
- The list may contain up to 10^6 elements, and each element can be as large as 10^9 in magnitude.

Examples:

Consider the following examples for better understanding:

1. **Example 1:**

- **Input:** list = [3, 1, 5, 4, 2], k = 2
- **Output:** 4
- **Explanation:** The distinct elements in the list are [1, 2, 3, 4, 5]. The 2nd maximum value is 4.

2. **Example 2:**

- **Input:** list = [7, 7, 7, 7, 7], k = 1
- **Output:** 7
- **Explanation:** The distinct elements in the list are [7]. The 1st maximum value is 7.

3. **Example 3:**

- **Input:** list = [2, 1, 2, 1, 2], k = 3
- **Output:** -1
- **Explanation:** The distinct elements in the list are [1, 2]. There is no 3rd maximum value.

For example:

Input	Result
5 3 1 5 4 2 2	4
6 7 7 7 7 7 7 1	7
10	-1

10

2 1 2 1 2 1 2 1 2 1

-1

3

Answer: (penalty regime: 0 %)

```
1 def fun(arr,k):
2     d=sorted(set(arr),reverse=True)
3     if k>len(d):
4         return -1
5     return d[k-1]
6 n=int(input())
7 arr=list(map(int,input().split()))
8 k=int(input())
9 print(fun(arr,k))
10
```

	Input	Expected	Got	
✓	5 3 1 5 4 2 2	4	4	✓
✓	6 7 7 7 7 7 7 1	7	7	✓

	1			
✓	10	-1	-1	✓
2	1 2 1 2 1 2 1 2 1			
3				

Passed all tests! ✓

Correct
Marks for this submission: 1.00/1.00.

Question 7
Correct
Mark 1.00 out of 1.00
[Flag question](#)

Given an integer array **nums** sorted in **non-decreasing** order, return an array of **the squares of each number** sorted in **non-decreasing** order.

Example 1:
Input: `nums = [-4,-1,0,3,10]`
Output: `[0,1,9,16,100]`
Explanation: After squaring, the array becomes `[16,1,0,9,100]`.
After sorting, it becomes `[0,1,9,16,100]`.

Example 2:
Input: `nums = [-7,-3,2,3,11]`
Output: `[4,9,9,49,121]`

- Constraints:**
- `1 <= nums.length <= 104`
 - `-103 <= nums[i] <= 104`
 - `nums` is sorted in **non-decreasing** order.

For example:

Test	Result
<code>print(sortedSquares([-4, -1, 0, 3, 10]))</code>	<code>0 1 9 16 100</code>

print(sortedSquares([-4,-1,0,3,10]))

[0, 1, 9, 16, 100]

Answer: (penalty regime: 0 %)

Reset answer

```
1 def sortedSquares(nums):
2     n=len(nums)
3     res=[0]*n
4     l,r=0,n-1
5     pos=n-1
6     while l<=r:
7         ll=nums[l]**2
8         rr=nums[r]**2
9         if ll>rr:
10            res[pos]=ll
11            l+=1
12        else:
13            res[pos]=rr
14            r-=1
15        pos-=1
16    return res
17
18
```

	Test	Expected	Got	
✓	print(sortedSquares([-4,-1,0,3,10]))	[0, 1, 9, 16, 100]	[0, 1, 9, 16, 100]	✓
✓	print(sortedSquares([-7,-3,2,3,11]))	[4, 9, 9, 49, 121]	[4, 9, 9, 49, 121]	✓

Passed all tests! ✓

Question 8
Correct
Mark 1.00 out of 1.00
[Flag question](#)

To find the frequency of numbers in a list and display in sorted order.

Constraints:

1<=n, arr[i]<=100

Input:

1 68 79 4 90 68 1 4 5

output:

1 2

4 2

5 1

68 2

79 1

90 1

For example:

Input	Result
4 3 5 3 4 5	3 2
	4 2
	5 2

Answer: (penalty regime: 0 %)

```
1 def frequency(arr):  
2     freq={}  
3     for num in arr:  
4         freq[num]=freq.get(num,0)+1  
5     for num in sorted(freq):  
6         print(num,freq[num])  
7  
8 d=list(map(int,input().split()))
```

```
7  
8 d=list(map(int,input().split()))  
9 frequency(d)  
10  
11  
12
```

	Input	Expected	Got	
✓	4 3 5 3 4 5	3 2 4 2 5 2	3 2 4 2 5 2	✓
✓	12 4 4 4 2 3 5	2 1 3 1 4 3 5 1 12 1	2 1 3 1 4 3 5 1 12 1	✓
✓	5 4 5 4 6 5 7 3	3 1 4 2 5 3 6 1 7 1	3 1 4 2 5 3 6 1 7 1	✓

Passed all tests! ✓

Correct

Correct

Flag question

1. List is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.
2. First Element: firstElement, the *first* element in the sorted list.
3. Last Element: lastElement, the *last* element in the sorted list.

First Element: 1

Last Element: 6

The first line contains an integer, n , the size of the list a .
The second line contains n , space-separated integers $a[i]$.

- $2 \leq n \leq 600$
- $1 \leq a[i] \leq 2 \times 10^6$

You must print the following three lines of output:

1. List is sorted in numSwaps swaps, where numSwaps is the number of swaps that took place.
2. First Element: firstElement, the *first* element in the sorted list.
3. Last Element: lastElement, the *last* element in the sorted list.

3

123

List is sorted in 0 swaps.

First Element: 1

First Element: 1

WEEK-11-CODING-Bubble and Merge Sort Attempt review - Google Chrome

Not secure 118.185.187.137/moodle/mod/quiz/review.php?attempt=162616&cmid=1395

RAJALAKSHMI PAPERBONDING COLLEGE REC-OCATS-1

First Element: 1
Last Element: 3

For example:

Input	Result
3 3 2 1	List is sorted in 3 swaps. First Element: 1 Last Element: 3
5 1 9 2 8 4	List is sorted in 4 swaps. First Element: 1 Last Element: 9

Answer: (penalty regime: 0 %)

```
1 def bubble(arr):
2     n=len(arr)
3     num=0
4     for i in range(n):
5         swapped=False
6         for j in range(0,n-1-i):
7             if arr[j]>arr[j+1]:
8                 arr[j],arr[j+1]=arr[j+1],arr[j]
9                 num+=1
10                swapped=True
11            if not swapped:
12                break
13        return num,arr[0],arr[-1]
14 n=int(input())
15 arr=list(map(int,input().split()))
16 num,f,l=bubble(arr)
17 print(f"List is sorted in {num} swaps.")
18 print(f"First Element: {f}")
19 print(f"Last Element: {l}")
```

Windows taskbar: Type here to search, Rain on Friday, 11-11-2024, 15:05

	Input	Expected	Got	
✓	3 3 2 1	List is sorted in 3 swaps. First Element: 1 Last Element: 3	List is sorted in 3 swaps. First Element: 1 Last Element: 3	✓
✓	5 1 9 2 8 4	List is sorted in 4 swaps. First Element: 1 Last Element: 9	List is sorted in 4 swaps. First Element: 1 Last Element: 9	✓

Passed all tests! ✓

Correct
Marks for this submission: 1.00/1.00.

Question 10
Correct
Mark 1.00 out of 1.00
Flag question

Given an integer array `nums`, return an integer array `counts` where `counts[i]` is the number of smaller elements to the right of `nums[i]`.

Example 1:
Input: `nums = [5,2,6,1]`
Output: `[2,1,1,0]`
Explanation:
To the right of 5 there are 2 smaller elements (2 and 1).
To the right of 2 there is only 1 smaller element (1).
To the right of 6 there is 1 smaller element (1).
To the right of 1 there is 0 smaller element.

Example 2:
Input: `nums = [-1]`
Output: `[0]`

Example 3:
Input: `nums = [-1,-1]`
Output: `[0,0]`

Constraints:

- $1 \leq \text{nums.length} \leq 10^5$
- $-10^4 \leq \text{nums}[i] \leq 10^4$

For example:

Test	Result
<code>print(countSmaller([5,2,6,1]))</code>	<code>[2, 1, 1, 0]</code>
<code>print(countSmaller([-1]))</code>	<code>[0]</code>

Answer: (penalty regime: 0 %)

Reset answer

```
1 def countSmaller(n):
2     def merge(e):
3         m=len(e)//2
4         if m:
5             l,r=merge(e[:m]),merge(e[m:])
6             for i in range(len(e)-1,-1,-1):
7                 if not r or l and l[-1][1]>r[-1][1]:
8                     res[l[-1][0]]+=len(r)
9                     e[i]=l.pop()
10                else:
11                    e[i]=r.pop()
12            return e
13    res=[0]*len(n)
14    merge(list(enumerate(n)))
15    return res
16
17
18
19
```