

[Dashboard](#) / [My courses](#) / [PSPP/PUP](#) / [Experiments based on Lists and its operations.](#) / [Week6_Coding](#)

Started on	Thursday, 9 May 2024, 10:24 AM
State	Finished
Completed on	Sunday, 12 May 2024, 10:24 AM
Time taken	3 days
Overdue	1 day
Marks	9.00/10.00
Grade	90.00 out of 100.00

Question **1**

Correct

Mark 1.00 out of 1.00

Determine the factors of a number (i.e., all positive integer values that evenly divide into a number) and then return the p^{th} element of the [list](#), sorted ascending. If there is no p^{th} element, return 0.

Example

n = 20
p = 3

The factors of 20 in ascending order are {1, 2, 4, 5, 10, 20}. Using 1-based indexing, if p = 3, then 4 is returned. If p > 6, 0 would be returned.

Constraints

$1 \leq n \leq 10^{15}$
 $1 \leq p \leq 10^9$

The first line contains an integer n, the number to factor.
The second line contains an integer p, the 1-based index of the factor to return.

Sample Case 0

Sample Input 0

10
3

Sample Output 0

5

Explanation 0

Factoring n = 10 results in {1, 2, 5, 10}. Return the p = 3rd factor, 5, as the answer.

Sample Case 1

Sample Input 1

10
5

Sample Output 1

0

Explanation 1

Factoring n = 10 results in {1, 2, 5, 10}. There are only 4 factors and p = 5, therefore 0 is returned as the answer.

Sample Case 2

Sample Input 2

1
1

Sample Output 2

1

Explanation 2

Factoring n = 1 results in {1}. The p = 1st factor of 1 is returned as the answer.

For example:

Input	Result
10 3	5
10 5	0
1 1	1

Answer: (penalty regime: 0 %)

```
1 def factor(n, p):
2     factors = []
3     # Find factors of n
4     for i in range(1, int(n**0.5) + 1):
5         if n % i == 0:
6             factors.append(i)
7             if i != n // i:
8                 factors.append(n // i)
9     # Sort the factors
10    factors.sort()
11    # Return the pth factor if it exists, otherwise return 0
12    if p <= len(factors):
13        return factors[p - 1]
14    else:
15        return 0
```

```
15         return 0
16
17 # Sample Input
18 n = int(input())
19 p = int(input())
20
21 # Output
22 print(factor(n, p))
23
```

	Input	Expected	Got	
✓	10 3	5	5	✓
✓	10 5	0	0	✓
✓	1 1	1	1	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **2**

Correct

Mark 1.00 out of 1.00

Complete the program to count frequency of each element of an array. Frequency of a particular element will be printed once.

Sample Test Cases

Test Case 1

Input

7
23
45
23
56
45
23
40

Output

23 occurs 3 times
45 occurs 2 times
56 occurs 1 times
40 occurs 1 times

Answer: (penalty regime: 0 %)

```
1 def count_frequency(arr):
2     frequency_dict = {}
3
4     # Count frequency of each element
5     for elem in arr:
6         if elem in frequency_dict:
7             frequency_dict[elem] += 1
8         else:
9             frequency_dict[elem] = 1
10
11    # Print the frequency of each element
12    for key, value in frequency_dict.items():
13        print(f"{key} occurs {value} times")
14
15    # Take input
16    n = int(input())
17    arr = []
18    for _ in range(n):
19        arr.append(int(input()))
20
21    # Call the function to count frequency
22    count_frequency(arr)
23
```

	Input	Expected	Got	
✓	7 23 45 23 56 45 23 40	23 occurs 3 times 45 occurs 2 times 56 occurs 1 times 40 occurs 1 times	23 occurs 3 times 45 occurs 2 times 56 occurs 1 times 40 occurs 1 times	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **3**

Correct

Mark 1.00 out of 1.00

Write a Python program to Zip two given lists of lists.

Input:

m : row size

n: column size

list1 and [list 2](#) : Two lists

Output

Zipped [List](#) : [List](#) which combined both list1 and list2

Sample test case

Sample input

2

2

1

3

5

7

2

4

6

8

Sample Output

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

Answer: (penalty regime: 0 %)

```
1 def zip_lists(list1, list2):
2     zipped_list = list(zip(list1, list2))
3     return [sum(sublist, []) for sublist in zipped_list]
4
5 # Input
6 m = int(input())
7 n = int(input())
8
9 list1 = []
10 for _ in range(m):
11     sublist = []
12     for _ in range(n):
13         sublist.append(int(input()))
14     list1.append(sublist)
15
16 list2 = []
17 for _ in range(m):
18     sublist = []
19     for _ in range(n):
20         sublist.append(int(input()))
21     list2.append(sublist)
22
23 # Output
24 zipped_list = zip_lists(list1, list2)
25 print(zipped_list)
26
27
28
29
```

	Input	Expected	Got	
✓	2 2 1 2 3 4 5 6 7 8	[[1, 2, 5, 6], [3, 4, 7, 8]]	[[1, 2, 5, 6], [3, 4, 7, 8]]	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **4**

Correct

Mark 1.00 out of 1.00

Given an array of numbers, find the index of the smallest array element (the pivot), for which the sums of all elements to the left and to the right are equal. The array may not be reordered.

Example

arr=[1,2,3,4,6]

- the sum of the first three elements, 1+2+3=6. The value of the last element is 6.
- Using zero based indexing, arr[3]=4 is the pivot between the two subarrays.
- The index of the pivot is 3.

Constraints

- $3 \leq n \leq 10^5$
- $1 \leq arr[i] \leq 2 \times 10^4$, where $0 \leq i < n$
- It is guaranteed that a solution always exists.

The first line contains an integer n, the size of the array arr.

Each of the next n lines contains an integer, arr[i], where $0 \leq i < n$.

Sample Case 0

Sample Input 0

4
1
2
3
3

Sample Output 0

2

Explanation 0

- The sum of the first two elements, 1+2=3. The value of the last element is 3.
- Using zero based indexing, arr[2]=3 is the pivot between the two subarrays.
- The index of the pivot is 2.

Sample Case 1

Sample Input 1

3
1
2
1

Sample Output 1

1

Explanation 1

- The first and last elements are equal to 1.
- Using zero based indexing, arr[1]=2 is the pivot between the two subarrays.
- The index of the pivot is 1.

For example:

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

Answer: (penalty regime: 0 %)

```
1 def find_pivot_index(arr):
2     n = len(arr)
3     total_sum = sum(arr)
4     left_sum = 0
5
```

```
6  |   for i in range(n):
7  |       total_sum -= arr[i] # Update total_sum to exclude the current element
8  |       if left_sum == total_sum:
9  |           return i
10 |       left_sum += arr[i]
11 |
12 |       return -1 # No pivot found
13 |
14 | # Read the size of the array
15 | n = int(input())
16 |
17 | # Read the array elements
18 | arr = [int(input()) for _ in range(n)]
19 |
20 | # Find the pivot index and print the result
21 | print(find_pivot_index(arr))
22 |
```

	Input	Expected	Got	
✓	4 1 2 3 3	2	2	✓
✓	3 1 2 1	1	1	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **5**
Correct
Mark 1.00 out of 1.00

Write a Python program to check if a given [list](#) is strictly increasing or not. Moreover, If removing only one element from the [list](#) results in a strictly increasing [list](#), we still consider the [list](#) true

Input:

n : Number of elements

List1: [List](#) of values

Output

Print "True" if [list](#) is strictly increasing or decreasing else print "False"

Sample Test Case

Input

7

1

2

3

0

4

5

6

Output

True

Answer: (penalty regime: 0 %)

```
1 def is_strictly_increasing(lst):
2     # Check if the list is strictly increasing or strictly decreasing
3     increasing = all(lst[i] < lst[i + 1] for i in range(len(lst) - 1))
4     decreasing = all(lst[i] > lst[i + 1] for i in range(len(lst) - 1))
5
6     return increasing or decreasing
7
8 def is_strictly_increasing_with_one_removed(lst):
9     # Check if removing any element results in a strictly increasing list
10    for i in range(len(lst)):
11        temp_list = lst[:i] + lst[i+1:]
12        if is_strictly_increasing(temp_list):
13            return True
14
15    return False
16
17 # Test Case
18 if __name__ == "__main__":
19     n = int(input()) # Number of elements
20     lst = [int(input()) for _ in range(n)] # List of values
21
22     # Check if the list is strictly increasing, strictly decreasing, or if removing one element results in a str
23     if is_strictly_increasing(lst) or is_strictly_increasing_with_one_removed(lst):
24         print("True")
25     else:
26         print("False")
27
```

	Input	Expected	Got	
✓	7 1 2 3 0 4 5 6	True	True	✓
✓	4 2 1 0 -1	True	True	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **6**

Correct

Mark 1.00 out of 1.00

Consider a program to insert an element / item in the sorted array. Complete the logic by filling up required code in editable section. Consider an array of size 10. The eleventh item is the data is to be inserted.

Sample Test Cases

Test Case 1

Input

1
3
4
5
6
7
8
9
10
11
2

Output

ITEM to be inserted:2
After insertion array is:
1
2
3
4
5
6
7
8
9
10
11

Test Case 2

Input

11
22
33
55
66
77
88
99
110
120
44

Output

ITEM to be inserted:44
After insertion array is:
11
22
33
44
55
66
77
88
99
110
120

Answer: (penalty regime: 0 %)

```
1 def sort(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
8     return arr
```

```
9 | n=10
10 | arr=[]
11 | for i in range(n):
12 |     element=int(input())
13 |     arr.append(element)
14 | add=int(input())
15 | print(f"ITEM to be inserted:{add}")
16 | arr.append(add)
17 | sorted_arr=sort(arr)
18 | print("After insertion array is:")
19 | for i in range(len(sorted_arr)):
20 |     print(sorted_arr[i])
```

	Input	Expected	Got	
✓	1 3 4 5 6 7 8 9 10 11 2	ITEM to be inserted:2 After insertion array is: 1 2 3 4 5 6 7 8 9 10 11	ITEM to be inserted:2 After insertion array is: 1 2 3 4 5 6 7 8 9 10 11	✓
✓	11 22 33 55 66 77 88 99 110 120 44	ITEM to be inserted:44 After insertion array is: 11 22 33 44 55 66 77 88 99 110 120	ITEM to be inserted:44 After insertion array is: 11 22 33 44 55 66 77 88 99 110 120	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **7**

Incorrect

Mark 0.00 out of 1.00

Given an array A of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that $A[i] - A[j] = k$, $i \neq j$.

Input Format

- 1. First line is number of test cases T. Following T lines contain:
- 2. N, followed by N integers of the array
- 3. The non-negative integer k

Output format

Print 1 if such a pair exists and 0 if it doesn't.

Example

Input

1

3

1

3

5

4

Output:

1

Input

1

3

1

3

5

99

Output

0

For example:

Input	Result
1 3 1 3 5 4	1
1 3 1 3 5 99	0

Answer: (penalty regime: 0 %)

```
1 def find_pair_with_difference(arr, k):
2     i = 0
3     j = 1
4     n = len(arr)
5
6     while i < n and j < n:
7         if i != j and arr[j] - arr[i] == k:
8             return 1
9         elif arr[j] - arr[i] < k:
10            j += 1
11        else:
12            i += 1
13
14            # Ensure i is always less than j
15        if i == j:
16            j += 1
17
18    return 0
19
20 # Function to process each test case
21 def process_test_case():
22     # Read input
23     n = int(input()) # Number of integers
24     arr = list(map(int, input().split())) # Array of integers
25     k = int(input()) # Difference
```

```
25     k = int(input()) # difference
26
27     # Find if such a pair exists
28     result = find_pair_with_difference(arr, k)
29
30     # Print the result
31     print(result)
32
33 # Read the number of test cases
34 t = int(input())
35
36 # Process each test case
37 for _ in range(t):
38     process_test_case()
39
40
41
42
```

	Input	Expected	Got	
✗	1 3 1 3 5 4	1	0	✗
✓	1 3 1 3 5 99	0	0	✓

Some hidden test cases failed, too.
Your code must pass all tests to earn any marks. Try again.

Show differences

Incorrect

Marks for this submission: 0.00/1.00.



Question **8**

Correct

Mark 1.00 out of 1.00

Output is a merged array without duplicates.

Input Format

N1 - no of elements in array 1

Array elements for array 1

N2 - no of elements in array 2

Array elements for array2

Output Format

Display the merged array

Sample Input 1

5

1

2

3

6

9

4

2

4

5

10

Sample Output 1

1 2 3 4 5 6 9 10

Answer: (penalty regime: 0 %)

```
1 def merge_and_remove_duplicates(arr1, arr2):
2     merged_set = set(arr1 + arr2)
3     merged_list = sorted(list(merged_set))
4     print(*merged_list)
5
6 # Input
7 n1 = int(input())
8 arr1 = []
9 for _ in range(n1):
10     arr1.append(int(input()))
11
12 n2 = int(input())
13 arr2 = []
14 for _ in range(n2):
15     arr2.append(int(input()))
16
17 # Output
18 merge_and_remove_duplicates(arr1, arr2)
```

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

	Input	Expected	Got	
✓	7	1 3 4 5 7 8 10 11 12 13 22 30 35	1 3 4 5 7 8 10 11 12 13 22 30 35	✓
	4			
	7			
	8			
	10			
	12			
	30			
	35			
	9			
	1			
	3			
	4			
	5			
	7			
	8			
	11			
	13			
	22			

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **9**

Correct

Mark 1.00 out of 1.00

Write a program to print all the locations at which a particular element (taken as input) is found in a [list](#) and also print the total number of times it occurs in the [list](#). The location starts from 1.

For example, if there are 4 elements in the array:

5
6
5
7

If the element to search is 5 then the output will be:

5 is present at location 1
5 is present at location 3
5 is present 2 times in the array.

Sample Test Cases

Test Case 1

Input

4
5
6
5
7
5

Output

5 is present at location 1.
5 is present at location 3.
5 is present 2 times in the array.

Test Case 2

Input

5
67
80
45
97
100
50

Output

50 is not present in the array.

Answer: (penalty regime: 0 %)

```
1 |
2 | def find_element_locations(arr, target):
3 |     locations = [] # List to store locations where target is found
4 |     count = 0 # Variable to store the count of occurrences
5 |
6 |     for i, num in enumerate(arr, start=1): # Loop through the array with 1-based indexing
7 |         if num == target:
8 |             locations.append(i) # Append the location to the list
9 |             count += 1 # Increment the count
10 |
11 |     if count > 0:
12 |         # Print each location and count of occurrences separately
13 |         for location in locations:
14 |             print(f"{target} is present at location {location}.")
15 |             print(f"{target} is present {count} times in the array.")
16 |     else:
17 |         print(f"{target} is not present in the array.")
18 |
19 | # Test Case
20 | if __name__ == "__main__":
21 |     n = int(input()) # Input: number of elements in the array
22 |     arr = [int(input()) for _ in range(n)] # Input: array elements
23 |     target = int(input()) # Input: element to search
24 |
25 |     find_element_locations(arr, target)
26 |
```

	Input	Expected	Got	
✓	4 5 6 5 7 5	5 is present at location 1. 5 is present at location 3. 5 is present 2 times in the array.	5 is present at location 1. 5 is present at location 3. 5 is present 2 times in the array.	✓
✓	5 67 80 45 97 100 50	50 is not present in the array.	50 is not present in the array.	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **10**

Correct

Mark 1.00 out of 1.00

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:

5

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 3 4	1 2 3 4
6 1 1 2 2 3 3	1 2 3

Answer: (penalty regime: 0 %)

```
1 n = int(input())
2 arr = [int(input()) for _ in range(n)]
3
4
5 distinct_elements = set(arr)
6 print(*distinct_elements)
7
```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

◀ Week6_MCQ

Jump to...

Tuples ▶