

[Dashboard](#) / [My courses](#) / [PSP/PUP](#) / [Searching techniques: Linear and Binary](#) / [Week10 Coding](#)

Started on	Friday, 24 May 2024, 9:38 AM
State	Finished
Completed on	Saturday, 25 May 2024, 11:05 AM
Time taken	1 day 1 hour
Marks	5.00/5.00
Grade	100.00 out of 100.00

Question 1

Correct

Mark 1.00 out of 1.00

Write a Python program for binary search.

For example:

Input	Result
1,2,3,5,8 6	False
3,5,9,45,42 42	True

Answer: (penalty regime: 0 %)

```

1 arr = list(map(int, input().split(',')))
2 x = int(input())
3 arr.sort()
4 low = 0
5 high = len(arr) - 1
6 found = False
7
8 while low <= high:
9     mid = (low + high) // 2
10    if arr[mid] < x:
11        low = mid + 1
12    elif arr[mid] > x:
13        high = mid - 1
14    else:
15        found = True
16        break
17
18 print(found)
19

```

	Input	Expected	Got	
✓	1,2,3,5,8 6	False	False	✓
✓	3,5,9,45,42 42	True	True	✓
✓	52,45,89,43,11 11	True	True	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 2

Correct

Mark 1.00 out of 1.00

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 n = int(input())
2 arr = list(map(int, input().split()))
3
4
5 stack = [(0, n)]
6 while stack:
7     start, end = stack.pop()
8     if end - start > 1:
9         mid = (start + end) // 2
10        stack.extend([(start, mid), (mid, end)])
11        arr[start:end] = sorted(arr[start:end])
12
13
14 print(*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 70	14 21 27 41 43 45 46 57 70	14 21 27 41 43 45 46 57 70	✓
✓	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.

Question **3**

Correct

Mark 1.00 out of 1.00

To find the frequency of numbers in a [list](#) and display in sorted order.

Constraints:
 $1 \leq n, \text{arr}[i] \leq 100$
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 arr = list(map(int, input().split()))
2
3 freq = {}
4 for num in arr:
5     freq[num] = freq.get(num, 0) + 1
6
7
8 for key in sorted(freq.keys()):
9     print(key, freq[key])
10

```

	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	✓

	Input	Expected	Got	
✓	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

Marks for this submission: 1.00/1.00.

Question 4

Correct

Mark 1.00 out of 1.00

An [list](#) contains N numbers and you want to determine whether two of the numbers sum to a given number K. For example, if the input is 8, 4, 1, 6 and K is 10, the answer is yes (4 and 6). A number may be used twice.

Input Format

The first line contains a single integer n , the length of [list](#)

The second line contains n space-separated integers, [list\[i\]](#).

The third line contains integer k.

Output Format

Print Yes or No.

Sample Input

```
7
0 1 2 4 6 5 3
1
```

Sample Output

```
Yes
```

For example:

Input	Result
5 8 9 12 15 3 11	Yes
6 2 9 21 32 43 43 1 4	No

Answer: (penalty regime: 0 %)

```

1 | n = int(input())
2 | arr = list(map(int, input().split()))
3 | k = int(input())
4 | s = set()
5 |
6 | for num in arr:
7 |
8 |     if k - num in s:
9 |         print("Yes")
10 |         break
11 |     s.add(num)
12 | else:
13 |
14 |     print("No")
15 |
```

	Input	Expected	Got	
✓	5 8 9 12 15 3 11	Yes	Yes	✓
✓	6 2 9 21 32 43 43 1 4	No	No	✓
✓	6 13 42 31 4 8 9 17	Yes	Yes	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 5

Correct

Mark 1.00 out of 1.00

Given an listof integers, sort the array in ascending order using the *Bubble Sort* algorithm above. Once sorted, print the following three lines:

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](#).

For example, given a worst-case but small array to sort: a=[6,4,1]. It took 3 swaps to sort the array. Output would be

Array is sorted in 3 swaps.

First Element: 1

Last Element: 6

Input Format

The first line contains an integer, n , the size of the [list](#) a .

The second line contains n , space-separated integers $a[i]$.

Constraints

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

Output Format

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](#).

Sample Input 0

3
1 2 3

Sample Output 0

[List](#) is sorted in 0 swaps.

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 n = int(input())
2 a = list(map(int, input().split()))
3
4 swaps = 0
5 for i in range(n):
6     for j in range(n-1):
7         if a[j] > a[j+1]:
8             a[j], a[j+1] = a[j+1], a[j]
9             swaps += 1
10
11 print(f"List is sorted in {swaps} swaps.")

```



```
12 | print(f"First Element: {a[0]}")
13 | print(f"Last Element: {a[-1]}")
```

	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.

[◀ Week10_MCQ](#)

Jump to...

Sorting ►