Table of content

num_seq

Your mission_EN

Write a function that takes an integer n as parameter, and returns the result of the following sequence: n + (n-2) + (n-4) ... (as long as n-x > 0).

Only an integer will be sent to the function.

Your function must be named num_seq and in the file named num_seq.py (and in a directory named num_seq).

Your mission_VN

Viết một hàm mà nhận một số nguyên làm tham số, và trả về kết quả của chuỗi sau: n + (n -2) + (n – 4) … (miễn là n-x > 0).

Chỉ có một số nguyên sẽ được gửi đến hàm.

Hàm phải được đặt tên là num_seq trong tập tin đặt tên là num_seq.py (và trong thư mục đặt tên là num_seq)

Example

```
1     def num_seq(n):
2         """ Takes an integer n as parameter, and returns the result of
the following sequence: n + (n-2) + (n-4) ... (as long as n-x > 0)
3         >>> num_seq(1)
4         1
5         >>> num_seq(22)
6         132
7         >>> num_seq(-89)
8         0
9         """
```

dic_depth

Your Mission EN

Write a function that takes a dictionary and returns the depth of the dictionary.

Only a valid dictionary will be sent to the function.

Your function must be named dic_depth and in the file named dic_depth.py (and in the directory dic_depth).

Your Mission VN

Viết một hàm nhận một từ điển và trả về độ sâu của từ điển đó.

Chỉ một từ điển hợp lệ được truyền đến hàm.

Hàm phải được đặt tên là dic_depth trong tập tin đặt tên là dic_depth.py (và trong thư mục dic_depth)

Signature

```
1     def dic_depth(dict):
2         """
3         >>> dic_depth({'a': 3})
4         1
5         >>> dic_depth({})
6         0
7         >>> dic_depth({'a':1, 'b': {'c':{}}})
8         2
9         >>> dic_depth({'one': 1, 'two': {'three': {'flat':
'entry'}}})
10        3
11        """
```

longest_sum

Your mission EN

Write a function that takes 2 arguments, one being an array of integers and the other a maximum value, and returns the size of the longest subarray whose sum is equal or below the maximum value.

Subarray means contiguous values in the original array. We have added comments to the examples to show you the corresponding subarray, but your function must of course return the size of the subarray only.

Only valid arguments will be sent to the function.

Your function must be named longest_sum and in the file named longest_sum.py (and in the directory longest_sum).

Your mission VN

Viết một hàm nhận vào 2 tham số, tham số thứ nhất là một mảng số nguyên, thứ hai là một 'giá trị lớn nhất', hàm trả về kích thước của mảng con dài nhất mà tổng của nó nhỏ hơn hoặc bằng 'giá trị lớn nhất'.

Mảng con nghĩa là các số liên tục nhau trong mảng ban đầu. Chúng tôi đã thêm bình luận vào các ví dụ để cho bạn thấy về các mảng con tương ứng, nhưng hàm của bạn tất nhiên chỉ trả về kích thước của mảng con.

Chỉ có các giá trị hợp lệ được truyền làm đầu vào cho hàm.

Hàm phải được đặt tên longest_sum trong tập tin đặt tên là longest_sum.py' (và trong thư mục longest_sum)

Signature

```
1     def longest_sum(arr, max):
2         """ Takes an array of integers and a maximum value, and
returns the size of the longest subarray whose sum is equal or below the
maximum value.
3         >>> longest_sum([1, 4, 10, 9, 7, 3, 2, 0, 5, 6, 8], 10)
4         4
5         # The subarray is [3, 2, 0, 5]
6
7         >>> longest_sum([10, 1, 6, 0, 5, 3, 4, 8, 2, 7, 9], 5)
8         2
9         # The subarray is [0, 5]
10
11        >>> longest_sum([9, 0, 7, 6, 2, 3, 12], 12)
```

```
12          3
13          # the subarray is [6, 2, 3]
14          """
```

can_wave

Your mission_EN

Write a function that takes a list of integers, and returns whether they can be arranged in a wave pattern such as i1 > i2 < i3 > i4 < i5 > i6 < ... (or i1 < i2 > i3 < i4 > i5 < i6 > ...)

Your function will return True if the integers can be arranged in such a pattern (using once each integer of the list), or False if they cannot.

Only a list of integers will be sent to the function.

Your function must be named can_wave and in the file named can_wave.py (and in the directory can_wave).

Your mission_VN

Viết một hàm nhận một danh sách các số nguyên, và trả về liệu chúng có thể sắp xếp trong một dạng sóng như là: i1 > i2 < i3 > i4 < i5 > i6 < ... (hoặc 1 < i2 > i3 < i4 > i5 < i6 > ...)

Hàm sẽ trả về True nếu các số nguyên có thể sắp xếp theo dạng như vậy (sử dụng mỗi lần một số nguyên của danh sách), hoặc trả về False nếu không thể sắp xếp.

Chỉ có đối số hợp lệ sẽ được gửi cho hàm.

Hàm phải được đặt tên là can_wave trong tập tin đặt tên là can_wave.py (và trong thư mục đặt tên là can_wave).

Example

```
1     def can_wave(list):
2         """ Takes an array of integers and returns whether the numbers
can be arranged in a wave pattern: a1 > a2 < a3 > a4 < a5 > …
3         >>> can_wave([-2, -1, 0, 2, -1, -1, -1])
4         False
5         >>> can_wave([5, -8, 9, 22, 10, 0])
6         True
7         >>> can_wave([5])
8         True
9         >>> can_wave([-8, -8])
10        False
11        """
```