Question1

Create a function that takes three parameters where:

* x is the start of the range (inclusive).
* y is the end of the range (inclusive).
* n is the divisor to be checked against.

Return an ordered list with numbers in the range that are divisible by the third parameter n. Return an empty list if there are no numbers that are divisible by n.

**Examples**

list\_operation(1, 10, 3) ➞ [3, 6, 9]

list\_operation(7, 9, 2) ➞ [8]

list\_operation(15, 20, 7) ➞ []

Question2

Create a function that takes in two lists and returns True if the second list follows the first list by **one** element, and False otherwise. In other words, determine if the second list is the first list shifted to the right by 1.

**Examples**

simon\_says([1, 2], [5, 1]) ➞ True

simon\_says([1, 2], [5, 5]) ➞ False

simon\_says([1, 2, 3, 4, 5], [0, 1, 2, 3, 4]) ➞ True

simon\_says([1, 2, 3, 4, 5], [5, 5, 1, 2, 3]) ➞ False

**Notes**

* Both input lists will be of the same length, and will have a minimum length of 2.
* The values of the 0-indexed element in the second list and the n-1th indexed element in the first list do not matter.

Question3

A group of friends have decided to start a secret society. The name will be the first letter of each of their names, sorted in alphabetical order.

Create a function that takes in a list of names and returns the name of the secret society.

### Examples

society\_name(["Adam", "Sarah", "Malcolm"]) ➞ "AMS"

society\_name(["Harry", "Newt", "Luna", "Cho"]) ➞ "CHLN"

society\_name(["Phoebe", "Chandler", "Rachel", "Ross", "Monica", "Joey"])

Question4

An isogram is a word that has no duplicate letters. Create a function that takes a string and returns either True or False depending on whether or not it's an "isogram".

**Examples**

is\_isogram("Algorism") ➞ True

is\_isogram("PasSword") ➞ False

# Not case sensitive.

is\_isogram("Consecutive") ➞ False

**Notes**

* Ignore letter case (should not be case sensitive).
* All test cases contain valid one word strings.

Question5

Create a function that takes a string and returns True or False, depending on whether the characters are in order or not.

### Examples

is\_in\_order("abc") ➞ True

is\_in\_order("edabit") ➞ False

is\_in\_order("123") ➞ True

is\_in\_order("xyzz") ➞ True

### Notes

You don't have to handle empty strings.

Solutions:

Sure! I'll provide you with solutions to each of the questions you asked.

Question 1:

Here's a function that takes three parameters `x`, `y`, and `n` and returns an ordered list of numbers within the range `[x, y]` that are divisible by `n`:

```python

def list\_operation(x, y, n):

return [num for num in range(x, y+1) if num % n == 0]

```

Example usage:

```python

print(list\_operation(1, 10, 3)) # Output: [3, 6, 9]

print(list\_operation(7, 9, 2)) # Output: [8]

print(list\_operation(15, 20, 7)) # Output: []

```

Question 2:

Here's a function that takes two lists and determines if the second list is the first list shifted to the right by 1 element:

```python

def simon\_says(lst1, lst2):

return lst1[:-1] == lst2[1:]

```

Example usage:

```python

print(simon\_says([1, 2], [5, 1])) # Output: True

print(simon\_says([1, 2], [5, 5])) # Output: False

print(simon\_says([1, 2, 3, 4, 5], [0, 1, 2, 3, 4])) # Output: True

print(simon\_says([1, 2, 3, 4, 5], [5, 5, 1, 2, 3])) # Output: False

```

Question 3:

Here's a function that takes a list of names and returns the name of the secret society:

```python

def society\_name(names):

return ''.join(sorted([name[0] for name in names]))

```

Example usage:

```python

print(society\_name(["Adam", "Sarah", "Malcolm"])) # Output: "AMS"

print(society\_name(["Harry", "Newt", "Luna", "Cho"])) # Output: "CHLN"

print(society\_name(["Phoebe", "Chandler", "Rachel", "Ross", "Monica", "Joey"])) # Output: "CJMPRR"

```

Question 4:

Here's a function that checks whether a string is an isogram (no duplicate letters):

```python

def is\_isogram(string):

lower\_string = string.lower()

return len(lower\_string) == len(set(lower\_string))

```

Example usage:

```python

print(is\_isogram("Algorism")) # Output: True

print(is\_isogram("PasSword")) # Output: False

print(is\_isogram("Consecutive")) # Output: False

```

Question 5:

Here's a function that checks whether the characters in a string are in order:

```python

def is\_in\_order(string):

return string == ''.join(sorted(string))

```

Example usage:

```python

print(is\_in\_order("abc")) # Output: True

print(is\_in\_order("edabit")) # Output: False

print(is\_in\_order("123")) # Output: True

print(is\_in\_order("xyzz")) # Output: True

```

These are the solutions to the questions you provided. Let me know if you need any further assistance!