1. Given a sentence, return the number of words which have the same first and last letter.

**Examples**

count\_same\_ends("Pop! goes the balloon") ➞ 1

count\_same\_ends("And the crowd goes wild!") ➞ 0

count\_same\_ends("No I am not in a gang.") ➞ 1

2. The Atbash cipher is an encryption method in which each letter of a word is replaced with its "mirror" letter in the alphabet: A <=> Z; B <=> Y; C <=> X; etc.

Create a function that takes a string and applies the Atbash cipher to it.

**Examples**

atbash("apple") ➞ "zkkov"

atbash("Hello world!") ➞ "Svool dliow!"

atbash("Christmas is the 25th of December") ➞ "Xsirhgnzh rh gsv 25gs lu Wvxvnyvi"

3. Create a class Employee that will take a full name as argument, as well as a set of none, one or more keywords. Each instance should have a name and a lastname attributes plus one more attribute for each of the keywords, if any.

**Examples**

john = Employee("John Doe")

mary = Employee("Mary Major", salary=120000)

richard = Employee("Richard Roe", salary=110000, height=178)

giancarlo = Employee("Giancarlo Rossi", salary=115000, height=182, nationality="Italian")

john.name ➞ "John"

mary.lastname ➞ "Major"

richard.height ➞ 178

giancarlo.nationality ➞ "Italian"

4. Create a function that determines whether each seat can "see" the front-stage. A number can "see" the front-stage if it is strictly greater than the number before it.

Everyone can see the front-stage in the example below:

# FRONT STAGE

[[1, 2, 3, 2, 1, 1],

[2, 4, 4, 3, 2, 2],

[5, 5, 5, 5, 4, 4],

[6, 6, 7, 6, 5, 5]]

# Starting from the left, the 6 > 5 > 2 > 1, so all numbers can see.

# 6 > 5 > 4 > 2 - so all numbers can see, etc.

Not everyone can see the front-stage in the example below:

# FRONT STAGE

[[1, 2, 3, 2, 1, 1],

[2, 4, 4, 3, 2, 2],

[5, 5, 5, 10, 4, 4],

[6, 6, 7, 6, 5, 5]]

# The 10 is directly in front of the 6 and blocking its view.

The function should return True if every number can see the front-stage, and False if even a single number cannot.

Examples

can\_see\_stage([

[1, 2, 3],

[4, 5, 6],

[7, 8, 9]

]) ➞ True

can\_see\_stage([

[0, 0, 0],

[1, 1, 1],

[2, 2, 2]

]) ➞ True

can\_see\_stage([

[2, 0, 0],

[1, 1, 1],

[2, 2, 2]

]) ➞ False

can\_see\_stage([

[1, 0, 0],

[1, 1, 1],

[2, 2, 2]

]) ➞ False

# Number must be strictly smaller than

# the number directly behind it.

5. Create a Pizza class with the attributes order\_number and ingredients (which is given as a list). Only the ingredients will be given as input.

You should also make it so that its possible to choose a ready made pizza flavour rather than typing out the ingredients manually! As well as creating this Pizza class, hard-code the following pizza flavours.

Name Ingredients

hawaiian ham, pineapple

meat\_festival beef, meatball, bacon

garden\_feast spinach, olives, mushroom

**Examples**

p1 = Pizza(["bacon", "parmesan", "ham"]) # order 1

p2 = Pizza.garden\_feast() # order 2

p1.ingredients ➞ ["bacon", "parmesan", "ham"]

p2.ingredients ➞ ["spinach", "olives", "mushroom"]

p1.order\_number ➞ 1

p2.order\_number ➞ 2