Question1. Write a function that stutters a word as if someone is struggling to read it. The first two letters are repeated twice with an ellipsis ... and space after each, and then the word is pronounced with a question mark ?.

**Examples**

stutter("incredible") ➞ "in... in... incredible?"

stutter("enthusiastic") ➞ "en... en... enthusiastic?"

stutter("outstanding") ➞ "ou... ou... outstanding?"

Hint :- Assume all input is in lower case and at least two characters long.

# Write a function that stutters a word as if someone is struggling to read it.

# The first two letters are repeated twice with an ellipsis ... and space after each, and then the word

# is pronounced with a question mark ?.

# Define a function, function1 which takes a string as input parameter and

# and returns a required string

def function1(s):

return s[0:2]+"..."+s[0:2]+"..."+s+"?"

# Verify function with various inputs given

# stutter("incredible") ➞ "in... in... incredible?"

s1 = "incredible"

print(function1(s1))

# stutter("enthusiastic") ➞ "en... en... enthusiastic?"

s1 = "enthusiastic"

print(function1(s1))

# stutter("outstanding") ➞ "ou... ou... outstanding?"

s1 = "outstanding"

print(function1(s1))

Question 2.Create a function that takes an angle in radians and returns the corresponding angle in degrees rounded to one decimal place.

### Examples

radians\_to\_degrees(1) ➞ 57.3

radians\_to\_degrees(20) ➞ 1145.9

radians\_to\_degrees(50) ➞ 2864.8

# Create a function that takes an angle in radians and returns the corresponding angle in degrees

# rounded to one decimal place.

# Define a constant

pi = 3.142857

# Define a function to convert randians into degrees

def RadiansToDegrees(a):

#Formula to convert randians into degrees is

return round(a\*180/pi,1)

# radians\_to\_degrees(1) ➞ 57.3

print(RadiansToDegrees(1))

# radians\_to\_degrees(20) ➞ 1145.5

print(RadiansToDegrees(20))

# radians\_to\_degrees(50) ➞ 2863.6

print(RadiansToDegrees(50))

Question 3. In this challenge, establish if a given integer num is a Curzon number. If 1 plus 2 elevated to num is exactly divisible by 1 plus 2 multiplied by num, then num is a Curzon number.

Given a non-negative integer num, implement a function that returns True if num is a Curzon number, or False otherwise.

### Examples

is\_curzon(5) ➞ True

# 2 \*\* 5 + 1 = 33

# 2 \* 5 + 1 = 11

# 33 is a multiple of 11

is\_curzon(10) ➞ False

# 2 \*\* 10 + 1 = 1025

# 2 \* 10 + 1 = 21

# 1025 is not a multiple of 21

is\_curzon(14) ➞ True

# 2 \*\* 14 + 1 = 16385

# 2 \* 14 + 1 = 29

# 16385 is a multiple of 29

# In this challenge, establish if a given integer num is a Curzon number.

# If 1 plus 2 elevated to num is exactly divisible by 1 plus 2 multiplied by num, then num is a Curzon number.

# Given a non-negative integer num, implement a function that returns True if num is a Curzon number, or False otherwise.

# Define a function, function1 which takes a number as input

# and returns whether a number is curzon no or not

def function1(n):

x = pow(2,n) + 1

y = 2 \* n +1

if (x % y == 0):

return True

else:

return False

# Check the function, function1 with multiple inputs and check the result

print(function1(5)) # It should print True

print(function1(10)) # It should print False

print(function1(14)) # It should print True

Question 4.Given the side length x find the area of a hexagon.



### Examples

area\_of\_hexagon(1) ➞ 2.6

area\_of\_hexagon(2) ➞ 10.4

area\_of\_hexagon(3) ➞ 23.4

# Given the side length x find the area of a hexagon.

# import math function

import math

# Define a function to calculate area of a hexagon

def AreaHexagon(a):

# Formula to calculate Area of a hexagon is

return round(3\*math.sqrt(3)\*a\*a/2,1)

# Check area\_of\_hexagon(1) ➞ 2.6

print("Area of a hexagon: ",AreaHexagon(1))

# area\_of\_hexagon(2) ➞ 10.4

print("Area of a hexagon: ",AreaHexagon(2))

# area\_of\_hexagon(3) ➞ 23.4

print("Area of a hexagon: ",AreaHexagon(3))

Question 5. Create a function that returns a base-2 (binary) representation of a base-10 (decimal) string number. To convert is simple: ((2) means base-2 and (10) means base-10) 010101001(2) = 1 + 8 + 32 + 128.

Going from right to left, the value of the most right bit is 1, now from that every bit to the left will be x2 the value, value of an 8 bit binary numbers are (256, 128, 64, 32, 16, 8, 4, 2, 1).

### Examples

binary(1) ➞ "1"

# 1\*1 = 1

binary(5) ➞ "101"

# 1\*1 + 1\*4 = 5

binary(10) ➞ "1010"

# 1\*2 + 1\*8 = 10

# Create a function that returns a base-2 (binary) representation of a base-10 (decimal) string number.

# Define a function, function1 which returns binary output

def function1(n):

return bin(n).replace("b","")

# Verify function, function1 with multiple inputs given

print(function1(1)) # binary(1) ➞ "01"

print(function1(5)) # binary(5) ➞ "0101"

print(function1(10)) # binary(10) ➞ "01010"