**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.**

def stutter(word):

if len(word) >= 2:

stuttered\_word = word[:2] + "... " + word[:2] + "... " + word + "?"

return stuttered\_word

else:

return "Input word must be at least two characters long."

# Test cases

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

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

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

**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**

import math

def radians\_to\_degrees(radians):

degrees = radians \* (180 / math.pi)

return round(degrees, 1)

# Test cases

print(radians\_to\_degrees(1)) # ➞ 57.3

print(radians\_to\_degrees(20)) # ➞ 1145.9

print(radians\_to\_degrees(50)) # ➞ 2864.8

**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**

def is\_curzon(num):

numerator = 2 \*\* num + 1

denominator = 2 \* num + 1

if numerator % denominator == 0:

return True

else:

return False

# Test cases

print(is\_curzon(5)) # ➞ True

print(is\_curzon(10)) # ➞ False

print(is\_curzon(14)) # ➞ 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**

import math

def area\_of\_hexagon(x):

area = (3 \* math.sqrt(3) \* x\*\*2) / 2

return round(area, 1)

# Test cases

print(area\_of\_hexagon(1)) # ➞ 2.6

print(area\_of\_hexagon(2)) # ➞ 10.4

print(area\_of\_hexagon(3)) # ➞ 23.4

**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**

def binary(decimal):

binary\_str = ""

while decimal > 0:

remainder = decimal % 2

binary\_str = str(remainder) + binary\_str

decimal = decimal // 2

return binary\_str

# Test cases

print(binary(1)) # ➞ "1"

print(binary(5)) # ➞ "101"

print(binary(10)) # ➞ "1010"