1. 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.

In [1]:

```
def stutter(word):
    return word[:2] + "... " + word[:2] + "... " + word + "?"
```

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

In [1]:

```
import math

def radians_to_degrees(radians):
    """
    Converts an angle from radians to degrees and rounds the result to one decimal place.
    """
    degrees = radians * 180 / math.pi
    return round(degrees, 1)
```

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.

In [3]:

```
def is_curzon(num):
    numerator = 2**num + 1
    denominator = 2*num + 1
    return numerator % denominator == 0
```

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

In [ ]:

```
Here's the breakdown of the formula:

√3 is the square root of 3
The formula for the area of a regular hexagon is (3/2) * apothem * perimeter, where the apothem is the distance
Since the hexagon has equal sides, the perimeter is simply 6x, and the apothem is x times the square root of 3
Simplifying the formula gives us (3√3/2) * x^2
Therefore, to find the area of a hexagon with side length x, you can simply plug x into the formula and evaluat

Area = (3√3/2) * x^2
```

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.

In [5]:

```
def decimal_to_binary(decimal_str):
    decimal_int = int(decimal_str)
    binary_str = bin(decimal_int)[2:]  # remove the "0b" prefix
    return binary_str








decimal_str = "42"
binary_str = decimal_to_binary(decimal_str)
```

```
print(binary_str)  # output: 101010
```

```
101010
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js