1. Write a function that counts how many concentric layers a rug.

Examples:
count_layers(["AAAA","ABBA","AAAA"]) ➞ 2
count_layers(["AAAAAAAAA","ABBBBBBBA","ABBAAABBA","ABBBBBBBA","AAAAAAAAA"])
➞ 3
count_layers(["AAAAAAAAAAA","AABBBBBBBAA","AABCCCCCBAA","AABCAAACBAA","AAB
CADACBAA","AABCAAACBAA","AABCCCCCBAA","AABBBBBBBAA","AAAAAAAAAAA"]) ➞ 5

Ans:

In [9]:
```
def count_layers(inlist):
    output = set(inlist)
    print(f'count_layers({inlist}) ➞ {len(output)}')
count_layers(["AAAA","ABBA","AAAA"])
count_layers(["AAAAAAAAA","ABBBBBBBA","ABBAAABBA","ABBBBBBBA","AAAAAAAAA"])
count_layers(["AAAAAAAAAAA","AABBBBBBBAA","AABCCCCCBAA","AABCAAACBAA","AABCADACBAA","AABCAAACBAA","AABCCCCCBAA","AABBBBBBBAA
```

```
count_layers(['AAAA', 'ABBA', 'AAAA']) ➞ 2
count_layers(['AAAAAAAAA', 'ABBBBBBBA', 'ABBAAABBA', 'ABBBBBBBA', 'AAAAAAAAA']) ➞ 3
count_layers(['AAAAAAAAAAA', 'AABBBBBBBAA', 'AABCCCCCBAA', 'AABCAAACBAA', 'AABCADACBAA', 'AABCAAACBAA', 'AABCCCCCBAA', 'AABBBBB
BBAA', 'AAAAAAAAAAA']) ➞ 5
```

2. There are many different styles of music and many albums exhibit multiple styles. Create a function that takes a list of musical styles from albums and returns how many styles are unique.

Examples:
unique_styles([
"Dub,Dancehall",
"Industrial,Heavy Metal",
"Techno,Dubstep",
"Synth-pop,Euro-Disco",
"Industrial,Techno,Minimal"
]) ➞ 9
unique_styles([
"Soul",
"House,Folk",
"Trance,Downtempo,Big Beat,House",
"Deep House",
"Soul"
]) ➞ 7

Ans:

In [2]:

```
def unique_styles(inlist):
    styles = []
    for i in inlist:
        for j in i.split(','):
            styles.append(j)
    print(f'unique_styles({inlist}) ➞ {len(set(styles))}')
unique_styles([ "Dub,Dancehall", "Industrial,Heavy Metal", "Techno,Dubstep", "Synth-pop,Euro-Disco", "Industrial,Techno,Mini
unique_styles([ "Soul", "House,Folk", "Trance,Downtempo,Big Beat,House", "Deep House", "Soul" ])
```

```
unique_styles(['Dub,Dancehall', 'Industrial,Heavy Metal', 'Techno,Dubstep', 'Synth-pop,Euro-Disco', 'Industrial,Techno,Minima
l']) ➞ 9
unique_styles(['Soul', 'House,Folk', 'Trance,Downtempo,Big Beat,House', 'Deep House', 'Soul']) ➞ 7
```

3. Create a function that finds a target number in a list of prime numbers. Implement a binary search algorithm in your function. The target number will be from 2 through 97. If the target is prime then return "yes" else return "no".

Examples:
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]
is_prime(primes, 3) ➞ "yes"
is_prime(primes, 4) ➞ "no"
is_prime(primes, 67) ➞ "yes"
is_prime(primes, 36) ➞ "no"

Ans:

In [6]:

```
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]
def is_prime(primes,n):
    if n in primes:
        print(f'is_prime(primes, {n}) ➞ "yes"')
    else:
        print(f'is_prime(primes, {n}) ➞ "no"')
is_prime(primes, 3)
is_prime(primes, 4)
is_prime(primes, 67)
is_prime(primes, 36)
```

```
is_prime(primes, 3) ➞ "yes"
is_prime(primes, 4) ➞ "no"
is_prime(primes, 67) ➞ "yes"
is_prime(primes, 36) ➞ "no"
```

4. Create a function that takes in n, a, b and returns the number of positive values raised to the nth power that lie in the range [a, b], inclusive.

Examples:
power_ranger(2, 49, 65) ➞ 2
2 squares (n^2) lie between 49 and 65, 49 (7^2) and 64 (8^2)

power_ranger(3, 1, 27) ➞ 3
3 cubes (n^3) lie between 1 and 27, 1 (1^3), 8 (2^3) and 27 (3^3)

power_ranger(10, 1, 5) ➞ 1
1 value raised to the 10th power lies between 1 and 5, 1 (1^10)

power_ranger(5, 31, 33) ⟶ 1
power_ranger(4, 250, 1300) ⟶ 3

Ans:

In [9]:
```
def power_ranger(n,a,b):
    output = []
    for i in range(0,100):
        if pow(i,n) in range(a,b+1):
            output.append(i)
    print(f'power_ranger({n},{a},{b}) ⟶ {len(output)}')
power_ranger(2, 49, 65)
power_ranger(3, 1, 27)
power_ranger(10, 1, 5)
power_ranger(5, 31, 33)
```

```
power_ranger(2,49,65) ⟶ 2
power_ranger(3,1,27) ⟶ 3
power_ranger(10,1,5) ⟶ 1
power_ranger(5,31,33) ⟶ 1
```

5. Given a number, return the difference between the maximum and minimum numbers that can be formed when the digits are rearranged.

Examples:
rearranged_difference(972882) ⟶ 760833
988722 - 227889 = 760833
rearranged_difference(3320707) ⟶ 7709823
7733200 - 23377 = 7709823
rearranged_difference(90010) ⟶ 90981
91000 - 19 = 90981

Ans:

In [18]:
```
def rearranged_difference(n):
    smallest = int(''.join(sorted(str(n))))
    largest = int(''.join(sorted(str(n)))[::-1])
    print(f'rearranged_difference({n}) ⟶ {largest-smallest}')
rearranged_difference(972882)
rearranged_difference(3320707)
rearranged_difference(90010)
```

```
rearranged_difference(972882) ⟶ 760833
rearranged_difference(3320707) ⟶ 7709823
rearranged_difference(90010) ⟶ 90981
```