Problem29

April 19, 2022

1 Problem 29

Consider all integer combinations of a^b for 2 a 5 and 2 b 5:

```
2^2=4$, 2^3=8$, 2^4=16, 2^5=32
3^2=9, 3^3=27, 3^4=81, 3^5=243
4^2=16, 4^3=64, 4^4=256, 4^5=1024
5^2=25, 5^3=125, 5^4=625, 5^5=3125
```

If they are then placed in numerical order, with any repeats removed, we get the following sequence of 15 distinct terms:

4, 8, 9, 16, 25, 27, 32, 64, 81, 125, 243, 256, 625, 1024, 3125

How many distinct terms are in the sequence generated by a^b for $2 \leq a \leq 100$ and $2 \leq b \leq 100$?

If $a^b = c$, and d is a proper divisor of b, then:

$$a^b = (a^d)^{\frac{b}{d}} \tag{1}$$

e.g. for 3^8 $d = 2$ or 4. For $d = 2$:

$$3^8 = 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 = 3^2 \times 3^2 \times 3^2 \times 3^2 = 9 \times 9 \times 9 \times 9 = 9^4$$

For $d = 4$:

$$3^8 = 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 = 3^4 \times 3^4 = 81 \times 81 = 81^2$$

- Create matrix of $a \times b$, all set to 1 except values of a or $b < 2$, which are set to 0
- Loop through all numbers $2 -> 100$
- Check for proper divisors d of b
- For each divisor calculate alternate a and b
- Set alternates to 0 in matrix
- Sum matrix

[ ]:
```
import math
size = 100
abmatrix = []
printdebug = False
debug = True
```

[ ]:
```
def createMatrix(size):
    global abmatrix
    brow = []
    for a in range(0, size+1):
        brow = []
        for b in range(0, size+1):
            if a<2 or b<2:
                brow.append(False)
            else:
                brow.append(True)
        abmatrix.append(brow)

def matrixSum(matrix):
    thissum = 0
    for row in abmatrix:
        thissum = thissum + sum(row)
    return thissum

def findProperDivisors(b):
    divisorList = []
    limit = int(b/2) + 1
    d = 2
    while d < limit:
        if b % d == 0:
            divisorList.append(d)
            divisorList.append(int(b/d))
            limit = b/d
        d = d + 1
    divisorList.sort()
    return divisorList

def DebugTestOnly(size):
 # This function is going to calculate every power and check for duplicates
 # Expect that numbers will be too large for a and b approaching 100
 # Intention to use it to pinpoint the values that aren't getting picked up in
 # the other function

    mylist = []
    num = (size + 1 - 2) * (size + 1 - 2)
    print("num = ", num)

    for a in range(2, size + 1):
        for b in range(2, size + 1):
            if mylist.count(pow(a,b)) == 0: #checks to see if number already in␣
 ↪list
                mylist.append(pow(a, b))
            else:
```

```
                if printdebug == True: print("!! a = ", a, "  b = ", b, " pow =␣
 ↪", pow(a, b))

    print("list len = ", len(mylist))
```

[ ]:
```
createMatrix(size)
print(matrixSum(abmatrix))

for b in range(2,size+1):
    dList = findProperDivisors(b)
    for a in range(2, size+1):
        for d in dList:
            if  pow(a, d) <= size and int(b/d) >= 2:
                if printdebug: print("a",a,"b",b,"d",d," value = ", pow(a, b))
                abmatrix[pow(a, d)][int(b/d)] = False
print(matrixSum(abmatrix))
# if printdebug:
#     for row in abmatrix:
#         print(row)
if debug:
    DebugTestOnly(size)
```

```
9801
9277
num =  9801
list len =  9183
```

Main routine doesn't work properly because in some situations it needs to evaluate values of a and b that are beyond the limits, e.g.: - $8^4 = 4096$ - $4^6 = 4096$ - $2^{12} = 4096$

However, you would need to apply the $a^b = (a^d)^{\frac{b}{d}}$ formula to the 2^{12} value to get to the other 2 values, but for $b < 10$ this would not be evaluated.

DebugTestOnly function gives correct answer = 9183 in < 1 s