# 復習済み

A-E

# A Poisonous Oyster

正解

# B A..B..C

正解

# C Make it Simple

正解

# D Swap to Gather

正解

正解はできたが、なぜ自分のロジックが正解に辿り着けるか確信はないなかたまたま解けたから、理屈を理解した上での改善コードも書いた。結局、正解コードの理論も本質的には同じだった。

## 正解コード

N = int(input())

s = input()

ans = 0

one = []

for i, c in enumerate(s):

if c == "1":

one.append(i)

count = len(one)

base\_id = (count-1)//2

base = one[base\_id]

for i in range(count):

ans += abs(one[i]-base)-abs(i-base\_id)

print(ans)

## 改善コード

N = int(input())

s = input()

all\_one = s.count("1")

onecount = 0

ans = 0

for c in s:

if c == "1":

onecount += 1

else:

ans += min(onecount, all\_one-onecount)

print(ans)

# E GCD of Subset

不正解

不正解のコードはA[i]を素因数分解し、約数をすべて導き、その約数のカウントを1足していく。それをすべてのAに対して行った後、再びA[i]ごとに約数を見て個数がK以上のものから最大の約数を選ぶ。これはTLE。

正解コードは、Aにあるnの個数(n\_count)、Aにあるdの倍数の個数(multiple\_d\_count)、あるnと他にK-1個選んだ時の最大公約数(ans)の３つの配列を計算することによって求めている。

ここでのポイントは２つ。

・nから約数d（dの倍数)を求めるのは計算が多いが、dの倍数である数字を列挙するのは簡単ということ(イメージは縦で見るのではなく、横で見る)。

・オーダーを計算するとき、調和級数の

によって、とできること。

## 不正解コード

from collections import defaultdict

def prime(n):

ans = []

sieve = [1]\*(n+1)

for i in range(2, n+1):

if sieve[i] == 1:

ans.append(i)

for j in range(2\*i, n+1, i):

sieve[j] = 0

return ans

N, K = map(int, input().split())

A = list(map(int, input().split()))

d\_count = defaultdict(int)

D = []

prime\_upper = prime(10\*\*6)

for i in range(N):

tmp = A[i]

prime\_factorization = []

id = 0

while 1 < tmp:

while tmp % prime\_upper[id] == 0:

tmp //= prime\_upper[id]

prime\_factorization.append(prime\_upper[id])

id += 1

divisor = set()

count = len(prime\_factorization)

for num in range(2\*\*count):

tmp2 = num

d = 1

for k in range(count):

if tmp2 & 1 == 1:

d \*= prime\_factorization[k]

tmp2 >>= 1

divisor.add(d)

for d in divisor:

d\_count[d] += 1

D.append(sorted(list(divisor), reverse = True))

for i in range(N):

for d in D[i]:

if d\_count[d] >= K:

print(d)

break

## 正解コード

N, K = map(int, input().split())

A = list(map(int, input().split()))

MAX\_A = 10\*\*6+1

ans = [1]\*MAX\_A #あるnを含んだ時の最大公約数

n\_count = [0]\*MAX\_A

multiple\_d\_count = [0]\*MAX\_A

for a in A:

n\_count[a] += 1

for d in range(1, MAX\_A):

for n in range(d, MAX\_A, d):

multiple\_d\_count[d] += n\_count[n]

for d in range(1, MAX\_A):

if multiple\_d\_count[d] < K:

continue

for n in range(d, MAX\_A, d):

ans[n] = d

for a in A:

print(ans[a])

# F Prefix LIS Query

解いてない

# G Unevenness

解いてない