TWGSS OI Team

2022/23 Mini Contest I

Editorial

Words of Apologise

- Since my brain is unclear when I prep the mini contest, there are so many bugs.
- This shows that problem checkers (aka people who check problems) are necessarily.
- Hope that next time it would be better!

- And the difficulty seems too high.
- IDK What should I do about that.
- Maybe, try subtasks?
- Try to read problems more carefully (essentially @everyone!)

Speaking Exam

TC23111

Statistics

Attempt: 11

Subtask	Score	**Solved**	Partial (40%)
1	3	**10**	1
2	7	**3**	3
3	20	**5**	2
4	35	**2**	0
5	35	**2**	0

First Solve:

duncanNG

@ 0:24:07

Problem Statement

Simplified version:

You need to write T into

- sum of K ($c_L \leq K \leq c_R$) integers
- with each integers between s_L and s_R.

Subtask 1

$s_L = s_R$
$c_L = c_R$

- You NEED to speak $s_L = s_R$ seconds for exactly $c_L = c_R$ times.
- If $T = s_L \cdot c_L$, output `Possible` and c_L copies of s_L.
- Otherwise, output `Impossible`.

Easy 3 points! (Cumulative: 3)

Subtask 2

$s_L = s_R$

- Just check whether T is divisible by s_L!

Expected Score: 7 (Cumulative: 10)

Subtask 3

$c_L = c_R$

- Idea: speak each time almost the same length
  - Let's say you speak for $a_1, a_2, a_3, \cdots, a_K$ seconds each, with $a_1 \leq a_2 \leq a_3 \leq \cdots \leq a_K$.
  - You will want to minimize the difference between a_1 and a_K! (Why?)

- Hence, each output must be $T \div c_L$ or $T \div c_L + (T \bmod c_L > 0)$.

Expected Score: 20 (Cumulative: 30)

Full Solution

- Subtask 4 and the original subtask 5 are useless ☹
  - The constraint on number of rounds is due to I/O limit
- Just check all the possible number of rounds to speak!
- We can apply the idea in subtask 3 to solve it.

Time Complexity: O(N)

Expected Score: 100 (Cumulative: 100)

Digit Product

TC2312

Statistics

Attempt: 10

Subtask	Score	**Solved**
1	5	**7**
2	20	**2**
3	15	**2**
4	24	**2**
5	36	**2**

First Solve:

Jack

@ 0:19:23

Problem Statement

- Digit product: product of digits
- Find the minimum integer N with digit product K, or output `-1` when such N not exist.

Subtask 1

K is prime

- If $K = 2, 3, 5, 7$, then just output K.
- Otherwise, it is not possible to have such N, and output `-1`.

Expected Score: 5 (Cumulative: 5)

Subtask 2

$K = 5^s \cdot 7^t$ (s, t are non-negative integers)

- What does the above means?
- This means that our N only contains 5 and 7!

- Step 1: Repeatedly divide K by 5 to find s, 7 to find t
- Step 2: Output s copied of 5, t copies of 7

Expected Score: 20 (Cumulative: 25)

EDGE CASE :(

$$K = 1$$

Subtask 3

$K = 3^r \cdot 5^s \cdot 7^t$ (r, s, t are non-negative integers)

- What does the above means?
- This means that our N only contains [3], 5, 7 and [9]!

- Step 1: Repeatedly divide K **by 3 to find r,** by 5 to find s, 7 to find t
- Step 2: Output ($r \bmod 2$) copies of 3, s copied of 5, t copies of 7, $\left\lfloor \frac{r}{2} \right\rfloor$ copies of 9

Expected Score: 15 (Cumulative: 40)

Full Solution

- Greedy: just greedily try to divide by 9, by 8, … etc, and output copies of 2, 3, …, 9 in the order (@Jack)!

- Simulation (if you don't believe in the greedy solution):
  - Step 1: Find the minimum number of digits by above greedy solution
  - Step 2: For each digit, try the smallest digit possible

Expected Score: 100

Subtask 4

- This is a safety net if you use normal factorization instead of just repeatedly divide the numbers by $2, 3, 5, 7$ and **use `long long`**.

Expected Score: 24 (Cumulative: 64)

Yellow Flower

TC2313

Statistics

Attempt: 7

Subtask	Score	**Solved**
1	10	**2**
2	20	**4**
3	35	**3**
4	25	**0**
5	10	**0**

Highest Score (65):

Jack

duncanNG

Story

滿地黃花堆積，憔悴損，如今有誰堪摘？

李清照《聲聲慢．秋情》

Subtask 1

- $ans = \left(\frac{N}{2}\right) \times (\frac{N}{2} + 1) \div 2$ (i.e. $1 + 2 + \cdots + \frac{N}{2}$)
- Use **`long long`**

Expected Score: 10 (Cumulative: 10)

Subtask 2

- Simulation!
- Use two layers of for-loop. (Do you know how to do that?)

```
for L = 0 to N - 1
    for R = L to N - 1
        ans += check(L, R)
```

- `check(L, R)`: return 1 if the subsegment from `L` to `R` have more 黃花 and `0` otherwise (in O(N)).

Time Complexity: $O(N^3)$

Expected Score: 20 (Cumulative: 30)

Subtask 3

- Still simulation!
- Use subtask 2 idea, but improve `check(L, R)` to constant time.
- How?
- Partial Sum!
  - Use two partial sum arrays for the 花 respectively (`s[i + 1] = s[i] + a[i]`)

Time Complexity: $O(N^2)$

Expected Score: 55 (Cumulative: 65)

Subtask 4

- Theoretically solvable (but unrelated to full sol at all!)
- Idea: Just check those that "don't work" next to the red flowers
- Maybe use `set<pair<int, int>>` to maintain?
  - `set<int>`: a container that store non-repeating things quicker and consume less memory
- I don't really think anyone will solve this subtask.

Expected Score: 10 (Cumulative: 90)

@Jack Subtask 4 Code →

#圖文不符

```
string S;
int N;
vector<int> red;
cin >> N >> S;
int ans = (N*(N+1))/2;
for (int i=0; i<N; i++){
  if (S[i] == 'R') red.push_back(i);
}
last_R[0] = -1;
for (int i=0; i<N-1; i++){
  if (S[i] == 'R'){
    last_R[i+1] = i;
  }else{
    last_R[i+1] = last_R[i];
  }
}
next_R[N-1] = N;
for (int i=N-1; i>0; i--){
  if (S[i] == 'R'){
    next_R[i-1] = i;
  }else{
    next_R[i-1] = next_R[i];
  }
}
for (int i=0; i<red.size(); i++){
  for (int j=i; j<red.size(); j++){
    int cnt = j-i+1;
    int a = cnt-(red[j]-red[i]+1-cnt);
    if (a < 0) continue;
    int l, r, now;
    l = min(a, red[i]-(last_R[red[i]]+1));
    r = min(a, (next_R[red[j]]-1)-red[j]);
    ans--;
    ans -= l;
    ans -= r;
    for (int k=1; k<=l; k++){
      ans -= min(a-k, r);
    }
  }
}
```

Full solution

- Partial Sum!
  - Define partial sum $s_k = \#黃花 - \#紅花$ at first (How to maintain this?)
  - Note that $s_a - s_b > 0$ where $a > b$ means there are more 黃花 in the segment $[a, b-1]$.

- Example: YRYYR
  - $s = [0, 1, 0, 1, 2, 1]$
  - Answer = 6

- The problem becomes counting number of pairs (s_i, s_j) where $s_i < s_j$ in an array.
- Which is well-known solvable in O(n lg n), by merge-sort (divide and conquer) or Fenwick Tree (Binary-indexed Tree).

Expected Score: 100 (Cumulative: 100)

Thank you!