Programación Competitiva

Competencia #1

SEMANA i

acm International Collegiate Programming Contest

Problema #1: Alternating Sum

- Pre Calcular el bloque tamaño k, si es menor a sqrt(n), seguir pre calculando hasta que el tamaño sea aproximadamente sqrt(n), llamemos este tamano K.
- Quedan sqrt(n) bloques de tamaño sqrt(n), para cada uno de esos la solución es el valor del bloque previo * b^K * a^-K.
- **O(sqrt(n))**

- Pre calcular el bloque de tamaño k, y se hacen sumas parciales para el resto de los bloques.
- O(K)

Problema #2 Cutting Rectangle

- Obtener el gcd entre todos los counts y dividir counts entre gcd. Verificar si se puede formar un solo rectángulo de tamaño mediano usando los chicos (Si para cada width, el patrón de height tiene que ser el mismo, y los counts tienen que ser proporcionales)
- Encontrar el count de divisores del gcd y esa será la respuesta.

- El siguiente ejemplo son números válidos:
- 2 1 1
- 2 43 2
- 2 14 3
- 3 1 2
- 3 43 4
- 3 15 6

O(nlogn + sqrt(gcd))

Problema #3 Weird subtraction process

- Implementar un gcd con variación, en vez de hacer n%m usar n%2*m
- **O(logm + logn)**

Problema #4 Primal Sport

- Encontrar el factor primo más grande de X_2, y para todos los números en el rango, [X - P+1, X] llamémoslos X_{1i}, encontrar el factor primo más grande entre los X_{1i} y la respuesta será el número más chico, X_0.
- Usar Sieve para precalcular el factor primo más grande.
- **O(P + Xloglogx) donde P es el primo más grande de X_2**

- Nota: Si el número es primo solo se puede generar a sí mismo.

Problema #5 Odds and Ends

- Solo checar si el primer y último número de la secuencia sean impares, y que n sea impar.
- **O(N)**

Problema #6 Key races

- Comparar $s*V_1*T_1$ con $s*V_2*T_2$ si el primero es mayor regresar first, si el segundo es mayor regresar second, sino regresar Friendship.
- **O(1)**

Problema #7 The number on the board

- Obtener la suma de los dígitos, llamemoslo S.
- Contar la cantidad de dígitos de cada tipo y cada dígito que sea (de menor a mayor) cada dígito que no sea 9 convertirlo a 9, y sumarle a S la diferencia entre el 9 y el dígito, detenerse cuando S >= K
- **$O(\log_{10} n)$**

Problema #8 Unimodal array

- Iterar sobre el arreglo manteniendo una variable que contiene un estado, puedes estar en tres estados diferentes.
  - Estado 1: Ir subiendo, si son iguales pasar al estado 2, si baja pasar el estado 3.
  - Estado 2: Se mantiene, si sube es inválido, si baja pasa a estado 3.
  - Estado 3: Va bajando, si sube o se queda igual es inválido.
- **O(n)**