Ada Byron 2025

Regional de Andalucía – II Edición

Equipo del Jurado y Preparación de Problemas

Miembros del Jurado

- Pablo Dávila Herrero (US)
- Pablo Reina Jiménez (US)
- Kenny Jesús Flores Huamán (US)

Autores de los Problemas

- Pablo Dávila Herrero (US)
- Pablo Reina Jiménez (US)
- Kenny Jesús Flores Huamán (US)
- Juan Antonio Álvarez (US)
- Antonio Gutiérrez (US)
- Miguel A. Olivero (US)
- Gabriel Luque Polo(UMA)
- Rafael Guirado (UAL)
- Francisco Chicano (UMA)
- José Fidel Argudo (UCA)

Notas

- Los problemas aparecen ordenados por tasa de éxito.
- Puedes encontrar los enunciados y algunas soluciones a los problemas en este repositorio del Club de Algoritmia de la Universidad de Sevilla.

Clasificación de Problemas

Problema	Categoría
A – Torneo de Guerreros	Simulación
B - ¡Ataque Alienígena!	Trie
C – La Biblioteca de Renatia	Grafos, recubrimiento de vértices en bipartitos
D – Vales de descuento	Ad-hoc
E – El Concurso de Aba Dyron	Búsqueda binaria
F – Juego de Cifras	Combinatoria
G – En busca del genoma perdido	Ventana deslizante
H – Estrategia Estelar	Simulación, matemáticas
I – Ada en el jardín secreto de Versalles	Programación Dinámica
J – Los mineros y la extracción de cristales	Programación Dinámica
K – La receta definitiva	Búsqueda binaria, recubrimiento máximo, grafo

PROBLEMA I

Autor del Problema: Antonio Manuel Gutiérrez Fernández

I. Ada en el jardín secreto de Versalles

Envíos	Válidos	% Éxito	Primer Envío
19	8	42.1%	Los BoquerO(n³)

I. Ada en el jardín secreto

Recordatorio de Enunciado

Enunciado
- Dos hileras simétricas de flores con distintos valores de esplendor floral - Queremos recolectar el máximo esplendor floral - Nunca puede tomar dos flores consecutivas de la misma hilera - Nunca puede tomar dos flores que estén una frente a la otra

I. Ada en el jardín secreto

Ejemplo de enunciado

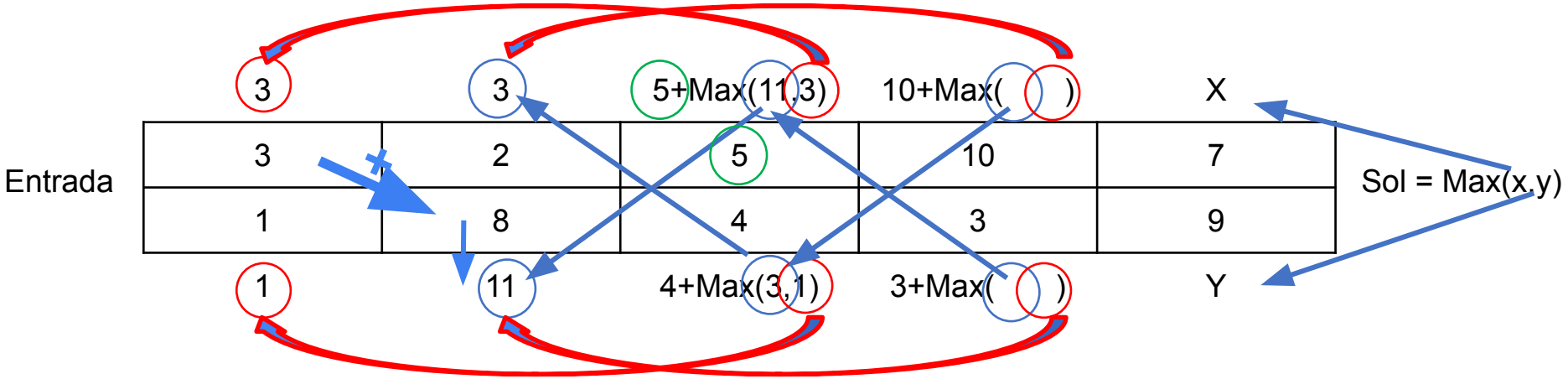
Entrada
2 4 20 2 5 27 1 8 9 1 5 3 2 5 10 7 1 8 4 3 9

Salida
56 30

I. Ada en el jardín secreto

Idea de la solución

Usando programación dinámica (no hace falta recursión, se puede usar un iterador)



I. Ada en el jardín secreto

Idea de la solución

Usando programación dinámica

- No hace falta recursión, se puede usar un iterador
- N decisiones (una por cada posición en la fila)
- Como mucho 3 opciones por cada decisión
  - Fila superior (si en el paso anterior no elegimos la misma)
  - Fila inferior (si en el paso anterior no elegimos la misma)
  - Ninguna de las dos

PROBLEMA A

Autor del Problema: Gabriel Luque Polo

A. Torneo de Guerreros

Envíos	Válidos	% Éxito	Primer Envío
109	34	31%	Los BoquerO(n³)

A. Torneo de Guerreros

Recordatorio de Enunciado

Enunciado

En el **Torneo de Guerreros**, N luchadores se enfrentan en duelos 1 vs 1. En cada ronda se elimina al perdedor de cada combate, y si queda un número impar de ganadores, se rescata al "mejor perdedor" para completar los pares. El torneo continúa hasta que solo quede uno. ¿Cuántos combates se necesitan en total?

A. Torneo de Guerreros

Ejemplo de enunciado

Entrada
5 6 8 16 18 20

Entrada
6 7 15 20 21

A. Torneo de Guerreros

Idea de la solución

- Cada ronda se reduce aproximadamente la mitad de los combatientes => ¿N-1? … Solo funciona en potencias de 2

- Proceso iterativo reduciendo de forma apropiada el número de combatientes (N) y contando combates

- Mejora de eficiencia: Parar cuando los combatientes sean potencia de 2 y sumar N-1
- Ojo: 10^19 no entra en un entero

PROBLEMA J

Autor del Problema: Miguel Á. Olivero

J. Los mineros y la extracción de cristales.

Envíos	Válidos	% Éxito	Primer Envío
53	15	28.3%	Los BoquerO(n^3)

Los mineros y la extracción de cristales. Problema J

Recordatorio de Enunciado

Enunciado

Buscamos el mejor camino de la posición (0, 0) a la posición (N-1, M-1) de una matriz NxM. El objetivo es maximizar la suma de los valores de las casillas por las que pasamos.

En cada paso podemos hacer uno de estos tres movimientos:

- Derecha
- Abajo
- Diagonal (derecha + abajo)

Los mineros y la extracción de cristales. Problema J

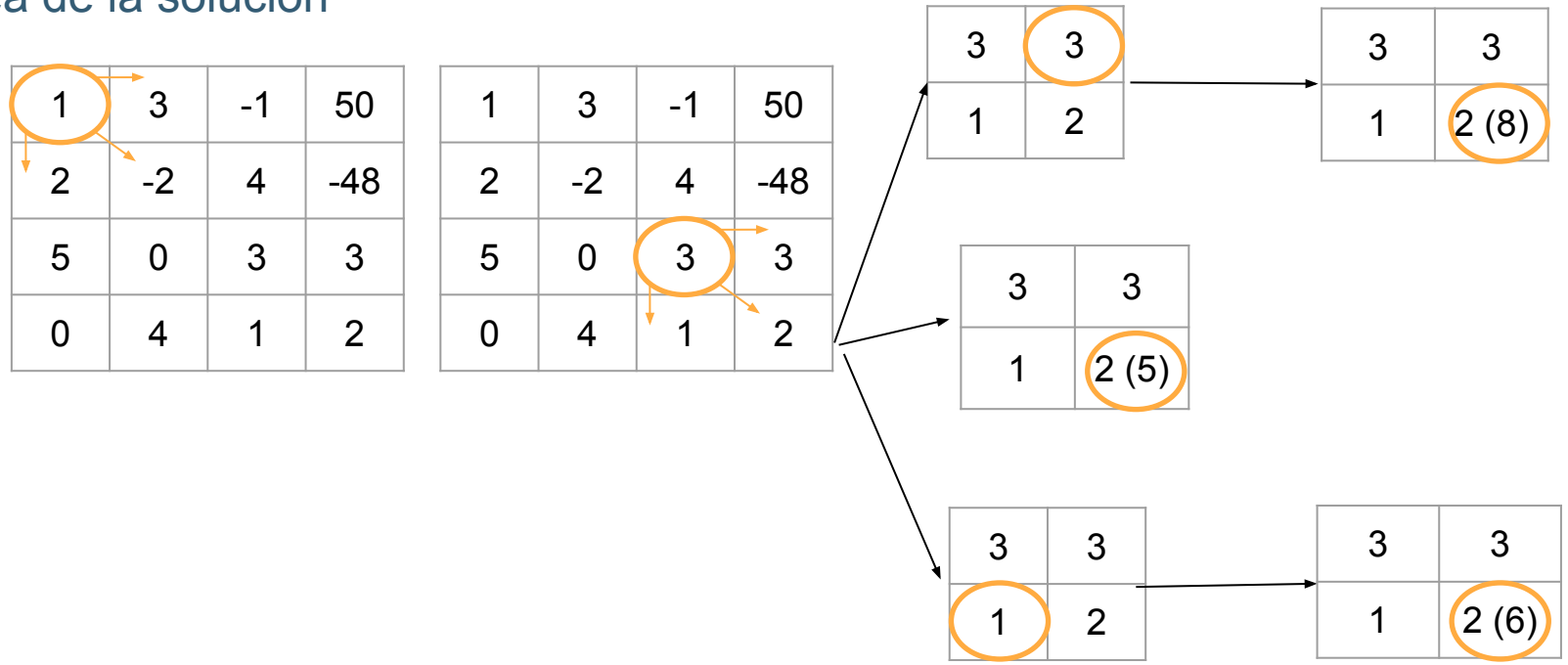
Ejemplo de enunciado

Entrada
4 4 1 3 -1 50 2 -2 4 -48 5 0 3 3 0 4 1 2

Salida
16

Los mineros y la extracción de cristales. Problema J

Idea de la solución



PROBLEMA B

Autor del Problema: Kenny Jesús Flores Huamán

B. ¡Ataque Alienígena!

Envíos	Válidos	% Éxito	Primer Envío
92	8	8%	lemaTecnico

B. ¡Ataque Alienígena!

Recordatorio de Enunciado

Enunciado

Dado un conjunto de N códigos (strings de dígitos) y Q prefijos de consulta, para cada consulta, calcula cuántos de los N códigos comienzan con ese prefijo. Procesa múltiples casos de prueba, imprimiendo "Case X:" y luego la respuesta para cada una de las Q consultas en líneas separadas.

B. ¡Ataque Alienígena!

Ejemplo de enunciado

Entrada

```
5 3
91234
87654
91245
12345
98765
912
876
111
```

Salida

```
Case 1:
2
1
0
```

B. ¡Ataque Alienígena!

Idea ingenua

Para cada prefijo de consulta q

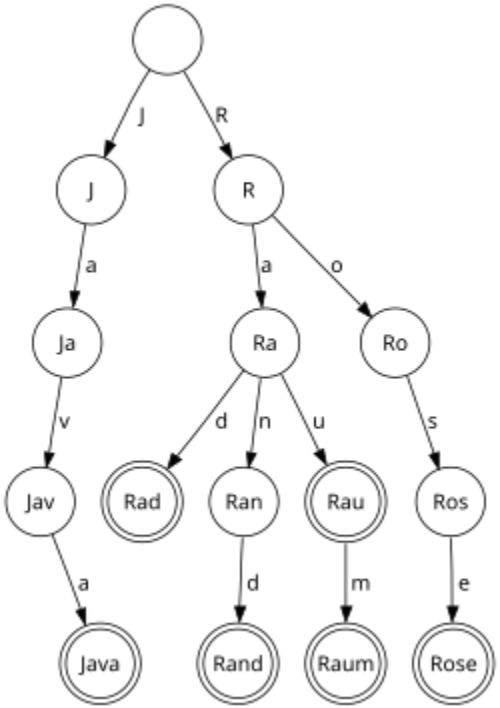
- Recorrer los N códigos alienígenas s.
- Comprobar si s comienza por p.
- Incrementar un contador si la condición se cumple

B. ¡Ataque Alienígena!

Resultado de la idea ingenua

B. ¡Ataque Alienígena!

Idea de la solución



Estructura de datos: Trie

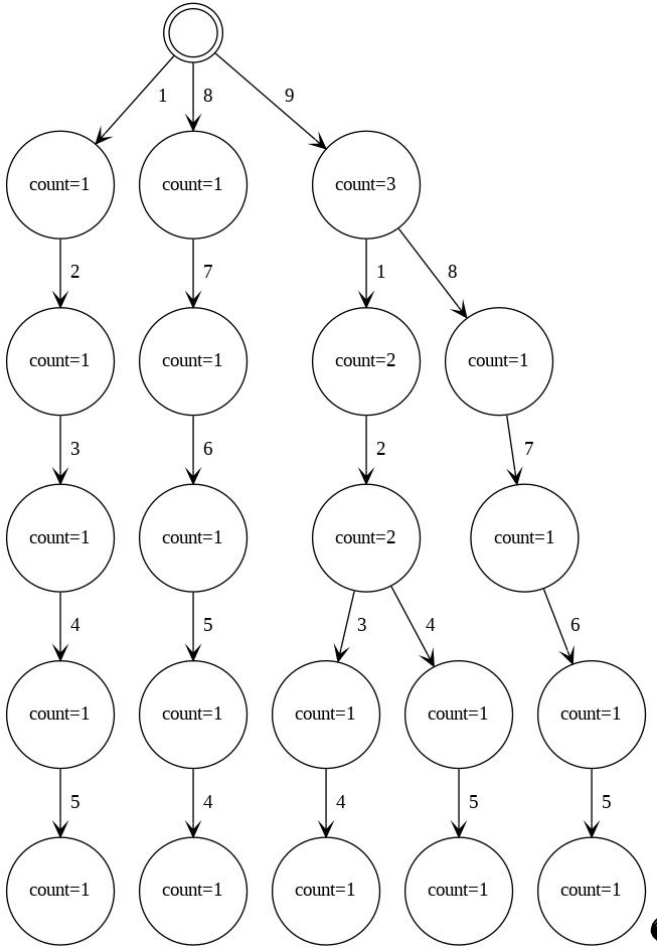
B. ¡Ataque Alienígena!

Idea de la solución

Construcción Trie

Para cada código alienígena de N:

- Recorro dígito por dígito
- Crea un nodo si un dígito no tiene camino con un contador a 1.
- Incrementa el contador en cada nodo del camino

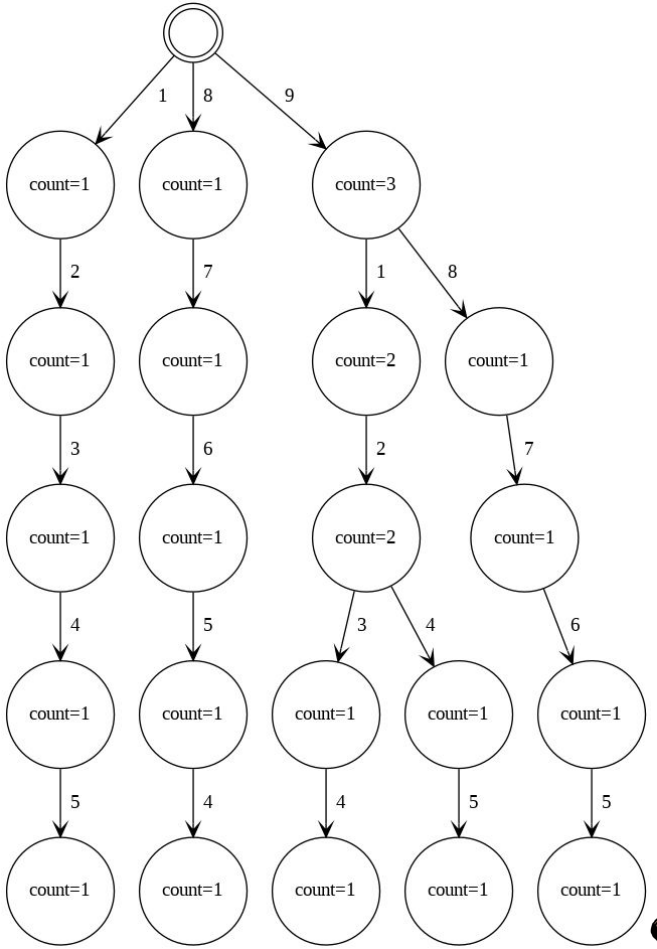


B. ¡Ataque Alienígena!

Idea de la solución

Consulta de prefijo en el Trie

- Recorre los dígitos del prefijo buscado desde la raíz del Trie
- Si un digito no tiene camino, el prefijo no existe (resultado 0)
- Si llegas al final del prefijo, el contador del nodo final es la respuesta



B. ¡Ataque Alienígena!

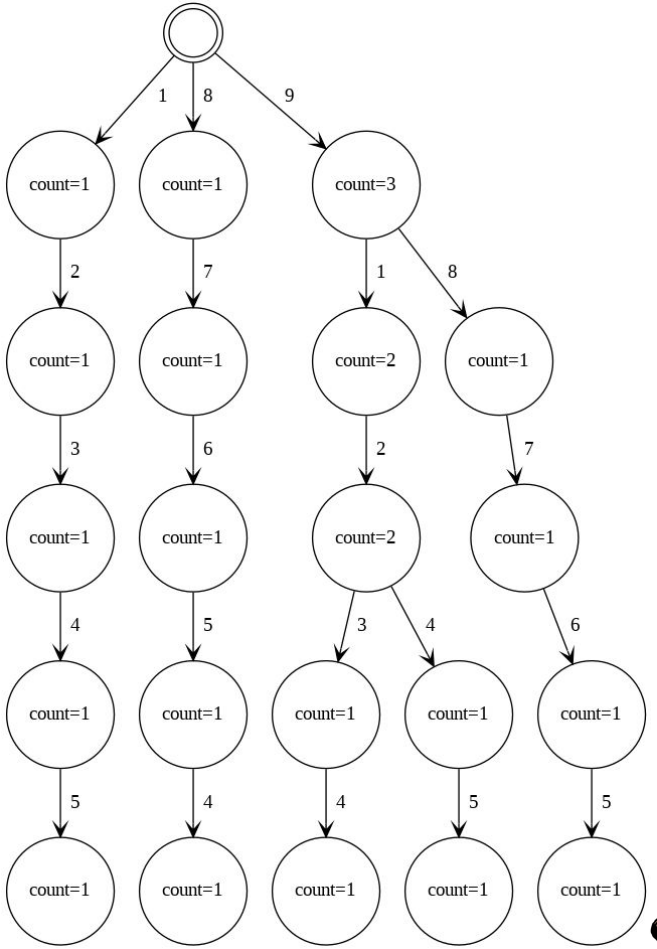
Ejemplo de consulta

Entrada

```
5 3
91234
87654
91245
12345
98765
912
876
111
```

Salida

```
Case 1:
2
1
0
```



PROBLEMA D

Autor del Problema: José Fidel Argudo

D. Vales de descuento

Envíos	Válidos	% Éxito	Primer Envío
6	117	5%	TrieNPHards

D. Vales de descuento

Recordatorio de Enunciado

Enunciado
Un supermercado quiere repartir n vales entre sus clientes de forma proporcional a sus gastos. La idea es dividir cada uno de los gastos entre 1,…,n y asignar un vale a los clientes con los n mayores cocientes. En caso de empate se lo lleva el cliente que más se ha gastado (Sistema D'Hondt)

D. Vales de descuento

Ejemplo de enunciado

Entrada

```
3 3
100 25 50
4 5
50 20 35 132
0 0
```

Entrada

```
2 0 1
1 0 1 3
```

D. Vales de descuento

Idea de la solución

- A cada gasto g_i le asociamos la lista de cocientes correspondiente [g_i/1],…,[g_i/n]
- Bastaría con extraer los primeros n cocientes y su cliente correspondiente (cuidado con los desempates)
- Para que la búsqueda de estos máximos sea eficiente podemos usar una estructura de max-heap, que mediante una estructura de árbol nos permite extraer los mayores cocientes sin necesidad de tenerlos ordenados

PROBLEMA E

Autor del Problema: Pablo Reina Jiménez

E. La competición de Aba Dyron

Envíos	Válidos	% Éxito	Primer Envío
146	17	11%	TrieNPHards

E. La competición de Aba Dyron

Recordatorio de Enunciado

Enunciado
Dada una lista de puntuaciones y unas puntuaciones mínimas ¿cuántas puntuaciones son mayores o iguales a cada una de las puntuaciones de corte?

E. La competición de Aba Dyron

Ejemplo de enunciado

Entrada
2 5 3 90 75 80 60 85 80 90 60 4 2 120 90 100 105 100 110

Salida
3 1 5 3 1 2

E. La competición de Aba Dyron

Idea de la solución

- Si solo hubiera una nota de corte para cada conjunto de puntuaciones nos bastaría con recorrerlas de forma lineal y ver cuántas son mayores o iguales que esta
- Sin embargo, para cada lista de puntuaciones nos harán varias consultas de puntuaciones mínimas (hasta 10^5) por tanto no es factible recorrer toda la lista tantas veces O(N*Q)
- Un primer enfoque sería ordenar la lista al principio y luego recorrerla solo hasta la primera nota que sea mayor o igual que la de corte y a partir de ahí todas pasarían. Esto mejoraría algunos casos pero no es suficiente O(N * log(N)) + O(N*Q) ~ O(N*Q)

E. La competición de Aba Dyron

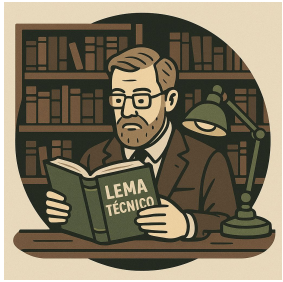
Idea de la solución

- Solución: ordenar la lista al principio y usar búsqueda binaria para buscar la primera puntuación mayor o igual a la de corte. Esto nos da una complejidad O(N * log(N)) + O(Q * log(N)) ~ O(N * log(N)) que ya si entraría en tiempo

PROBLEMA C

Autor del Problema: Pablo Reina Jiménez

C. La Biblioteca de Renatia



Envíos	Válidos	% Éxito	Primer Envío	Pendientes
0	0	0%	-	

C. La biblioteca de Renatia

Recordatorio de Enunciado

Enunciado

Dada una configuración de un piso de una biblioteca con paredes, casillas vacías y puestos a iluminar determinar cuantos faroles son necesarios para iluminar todos los puestos. Los faroles iluminan todas las casillas en vertical u horizontal desde su posición hasta llegar a una pared o borde de la planta. Podemos colocar hasta dos faroles en casillas vacías o puestos de estudios,

C. La biblioteca de Renatia

Ejemplo de enunciado

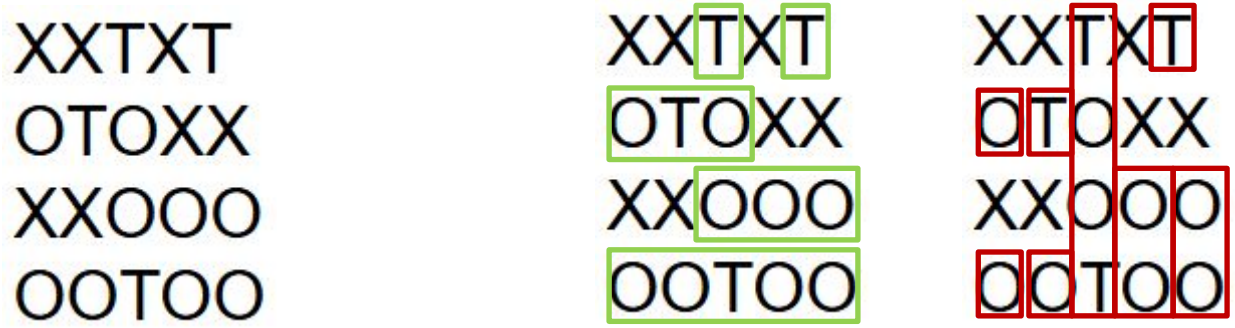
Entrada
3 2 3 T0O T0T 4 5 000XT 000XX XX000 TX000 4 5 XXTXT 0T0XX XX000 00T00

Salida
2 2 3

C. La biblioteca de Renatia

Idea de la solución

- En primer lugar, hay que darse cuenta que podemos mirar la planta como segmentos verticales y horizontales en lugar de celdas individuales, ya que no importará en que posición coloques el farol dentro de cada segmento ya que iluminará lo mismo



C. La biblioteca de Renatia.

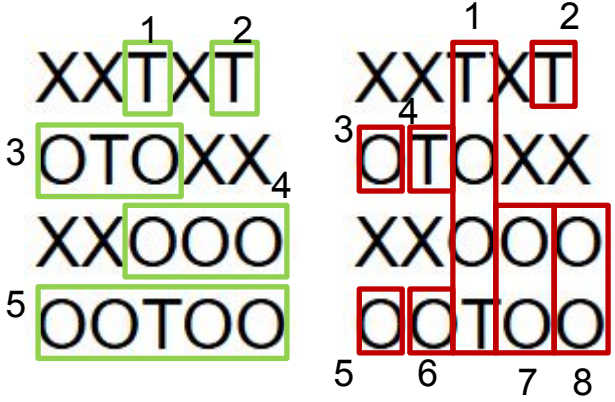
Idea de la solución

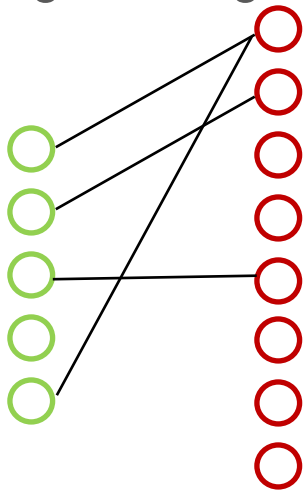
- Una vez hecho esto, podríamos pensar en hacer un backtracking e ir colocando faroles en cada segmento y quedarnos con la mejor configuración. Por desgracia esto dará TLE.
- Otra opción sería un enfoque greedy: comenzar a colocar faroles en las filas o columnas donde más puestos se iluminen. Sin embargo, esto no nos garantiza llegar a la mejor solución siempre, debido a la forma en la que se resuelven los empates.

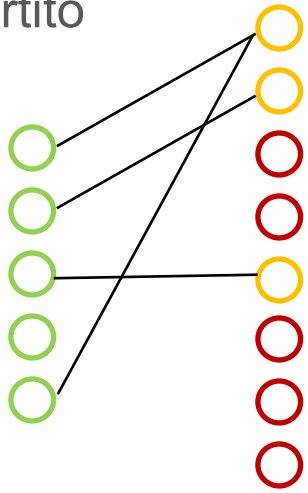
C. La biblioteca de Renatia

Idea de la solución

- Alternativa: ver el problema como un grafo. Cada segmento vertical y horizontal será un vértice del grafo y tendremos aristas entre un segmento vertical y uno horizontal si en la intersección de ambos hay un puesto de estudios. Esto da lugar a un grafo bipartito







C. La biblioteca de Renatia

Idea de la solución

- Por tanto, el problema se resuelve a encontrar un recubrimiento mínimo por vértices de todas las aristas del grafo. En grafos bipartitos esto es equivalente a encontrar un emparejamiento máximo (Teorema de König). Un emparejamiento máximo de un grafo bipartito es el mayor conjunto de aristas sin vértices comunes y puede encontrarse mediante el algoritmo de Hopcroft–Karp (BFS + DFS).

- Una vez obtenido el emparejamiento máximo, este será equivalente al mínimo de vértices para recubrir el grafo que a su vez es equivalente al mínimo de faroles necesarios para iluminar todos los puestos.

PROBLEMA F

Autor del Problema: Juan Antonio Álvarez

F. Juego de Cifras

Envíos	Válidos	% Éxito	Primer Envío
0	0	0%	-

F. Juego de Cifras

Recordatorio de Enunciado

Enunciado
Determinar cuántos números enteros positivos distintos se pueden formar usando cada cifra de un conjunto dado exactamente una vez, combinándolas con las cuatro operaciones aritméticas básicas (+, -, *, /) y paréntesis, sin permitir concatenaciones de dígitos.

F. Juego de Cifras

Ejemplo de enunciado

Entrada

```
3
1 2 3 4
2 5 7 9
1 2 5 8
```

Entrada

```
31
104
70
```

F. Juego de Cifras

Idea de la solución

- Se trata de un problema de combinatoria.
- Al poderse usar fracciones puede ocurrir que haya problemas de precisión.
- En Python existe el tipo Fraction que maneja todo eso bien, pero Java estándar no tiene nada y manejar decimales puede tener problemas de precisión por lo que es aconsejable usar un epsilon que compruebe que un valor puede ser entero (por ejemplo 28.0000007 podrá aproximarse a 28)

F. Juego de Cifras

Idea de la solución

- Simplemente hacer combinatoria con todas las posibles combinaciones de números y operaciones (teniendo en cuenta de que hay que usar 4 números)
- Al final comprobar si el número es entero o no. Si es entero y no lo habíamos calculado antes sumar 1 a nuestro acumulador.

PROBLEMA G

Autor del Problema: Pablo Reina Jiménez

G. En busca del genoma perdido

Envíos	Válidos	% Éxito	Primer Envío
2	0	0%	-

G. En busca del genoma perdido

Recordatorio de Enunciado

Enunciado
Dada una secuencia de ADN (compuestas por caracteres A, G, C y T), un número mínimo de N y un máximo M y una longitud mínima Kmin y una máxima Kmax, determinar cuantas subsecuencias de ADN cumplen que su longitud está entre Kmin y Kmax, contienen al menos N caracteres G y C y contienen estrictamente menos de M caracteres T y A

En busca del genoma perdido. Problema G

Ejemplo de enunciado

Entrada
2 10 3 5 1 2 ACGTGGCATG 8 2 4 1 2 GATTACAG

Salida
16 1

En busca del genoma perdido. Problema G

Idea de la solución

- Una primera idea podría ser comprobar todas las posibles subsecuencias y ver cuántas de ellas cumplen todas las condiciones una a una. Esto nos dará TLE
- Solución: usar sumas precalculadas y ventanas deslizantes
- En primer lugar, calculamos para cada posición del string cúantos caracteres A, G, C y T hay desde el principio hasta su posición. Con esto podremos calcular en tiempo constante cuántos caracteres de cada tipo hay en cada subsecuencia, sin tener que recorrerla cada vez.
- Tras esto, vamos deslizando una ventana por el array y ampliándola por la derecha siempre que se cumplan las condiciones y disminuyéndola por la izquierda cuando no

En busca del genoma perdido. Problema G

Idea de la solución

10 3 5 1 2
ACGTGGCATG

Buscamos secuencias de longitud entre 3 y 5, con al menos 1 nucléotido del tipo G y C y menos de 2 A y T

Precalculando las sumas llegamos a los siguientes arrays
A = [0 1 1 1 1 1 1 1 2 2 2]
C = [0 0 1 1 1 1 1 2 2 2 2]
G = [0 0 0 1 1 2 3 3 3 3 4]
T = [0 0 0 0 1 1 1 1 1 2 2]

Iniciamos un ans = 0 y el puntero derecho r = 0

En busca del genoma perdido. Problema G

Idea de la solución

10 3 5 1 2
ACGTGGCATG

Para cada i in [0..9]:
Si r < i --> r = i
Vamos aumentando r mientras que se cumpla la restricción de máximos nucleótidos A y T y no se exceda el tamaño máximo de ventana
Si la ventana entre i y r no igual o mayor al mínimo tamaño pasamos al siguiente i
Si se cumple el mínimo buscamos la primera posición j de la ventana a partir del cual se cumple la restricción de mínimos nucleótidos C y G
En este punto, todo intervalo [j, r] cumple que tiene longitud válida, no excede el límite de A/T y supera el umbral de G/C
Por tanto, podemos hacer ans += r-j+1, es decir, sumamos todos los intervalos que comienzan en i y acaban en k con k desde j hasta r

En busca del genoma perdido. Problema G

Idea de la solución

10 3 5 1 2
ACGTGGCATG

i	seq	r	i_min	j	sum	ans
0	ACGTG	4	2	2	3	3
1	CGTGG	5	3	3	3	6
2	GTGGC	6	4	6	1	7
3	TGGCA	7	5	6	2	9
4	GGCAT	8	6	6	3	12
5	GCATG	9	7	7	3	15
6	CATG	9	8	9	1	16
7	ATG	9	9	-	0	16
8	-					
9	-					

PROBLEMA H

Autor del Problema: Gabriel Luque Polo

H. Estrategia Estelar

Envíos	Válidos	% Éxito	Primer Envío
7	0	0%	-

H. Estrategia Estelar

Recordatorio de Enunciado

Enunciado

Tres flotas se desplazan por el espacio tridimensional. Cada día, las velocidades de las flotas se ajustan comparando sus posiciones. Si una flota está por detrás en un eje respecto a otra, aumenta su velocidad en ese eje en 1, mientras que la otra la reduce en 1; si están en la misma coordenada, no hay cambio. Después de actualizar las velocidades, cada flota avanza sumando su velocidad a su posición.

El objetivo es determinar cuántos días pasarán hasta que las tres flotas vuelvan exactamente a la situación inicial.

H. Estrategia Estelar

Ejemplo de enunciado

Entrada
2 1 0 2 2 1 0 3 0 4 -8 -3 2 12 7 -14 25 -20 6

Entrada
12 3261286

H. Estrategia Estelar

Idea de la solución

- Simular todo directamente puede requerir billones de pasos, ¡no viable!
- **Idea clave**: el movimiento en cada eje x, y, z es independiente:
  - Calcular el número de pasos necesarios para que las posiciones y velocidades en ese eje vuelvan a su estado inicial en cada eje
- El primer instante en que los tres ejes coinciden a la vez en su estado original es el **mínimo común múltiplo** de los ciclos de cada eje

PROBLEMA K

Autor del Problema: Pablo Dávila Herrero

K. La receta definitiva

Envíos	Válidos	% Éxito	Primer Envío
6	0	0	-

La receta definitiva. Problema K

Recordatorio de Enunciado

Enunciado

Hay ingredientes distribuidos en un plano bidimensional y debemos juntarlos haciendo el menor número posible de trayectos y minimizando la longitud del trayecto más largo. Además, los trayectos deben sumar suficiente satisfacción por buenas vistas para contentar a los repartidores.

La receta definitiva. Problema K (Ejemplo)

Entrada

```
2
1 0 0
1
1 1
1
0 1 1
3 14 15
30
92 65
35 89
79 32
6
0 1 3
0 2 8
0 3 4
1 2 6
1 3 2
3 3 6
```

Salida

```
1.41
-1.00
```

La receta definitiva. Problema K

Idea de la solución

Hagamos primero un par de simplificaciones:

- Supongamos que la longitud máxima de las aristas viene prefijada
- Supongamos que el número de trayectos no tiene por qué ser mínimo

La receta definitiva. Problema K

Idea de la solución

Hagamos primero un par de simplificaciones:

- Supongamos que la longitud máxima de las aristas viene prefijada
- Supongamos que el número de trayectos no tiene por qué ser mínimo

Entonces, para saber si hay solución basta con comprobar que el grafo es conexo y que las aristas "aceptadas" suman suficiente satisfacción

La receta definitiva. Problema K

Idea de la solución

Hagamos primero un par de simplificaciones:

- Supongamos que la longitud máxima de las aristas viene prefijada
- ~~Supongamos que el número de trayectos no tiene por qué ser mínimo~~

La receta definitiva. Problema K

Idea de la solución

Hagamos primero un par de simplificaciones:

- Supongamos que la longitud máxima de las aristas viene prefijada
- ~~Supongamos que el número de trayectos no tiene por qué ser mínimo~~

Entonces:

- El grafo resultante va a ser un árbol
- El número de trayectos debe ser exactamente *n*
- Para saber si hay solución hay que comprobar que el grafo es conexo y que al hacer un recubrimiento de peso máximo (con Kruskal) obtenemos suficiente satisfacción

La receta definitiva. Problema K

Idea de la solución

~~Hagamos primero un par de simplificaciones:~~

- ~~Supongamos que la longitud máxima de las aristas viene prefijada~~
- ~~Supongamos que el número de trayectos no tiene por qué ser mínimo~~

La receta definitiva. Problema K

Idea de la solución

~~Hagamos primero un par de simplificaciones:~~

- ~~Supongamos que la longitud máxima de las aristas viene prefijada~~
- ~~Supongamos que el número de trayectos no tiene por qué ser mínimo~~

Entonces:

- Obtenemos la longitud máxima mediante búsqueda binaria
- En cada iteración comprobamos si la longitud máxima propuesta da lugar a una solución
- ¿Cuándo paramos?
  - Listado con todas las distancias entre nodos --> Lista finita
  - Intervalos con flotantes --> Cuando la longitud sea despreciable

La receta definitiva. Problema K

Idea de la solución

Resumen:

- Búsqueda binaria para elegir la distancia máxima de los trayectos
- En cada iteración comprobamos si es factible:
  - Comprobar que el grafo es conexo
  - Obtener un recubrimiento máximo por aristas utilizando el algoritmo de Kruskal para maximizar la satisfacción
  - Comprobar que el nivel de satisfacción obtenido alcanza el mínimo
- Tener mucho cuidado con el redondeado de la solución

¡Gracias por participar!