Unicode 2023

Más estudiantes + Mejores rutas

Descripción del problema 1

Estáis organizando una excursión para la asociación de CodeLabZGZ para el alumnado de la EINA. Para ello contáis con el número exacto de estudiantes de cada una de las distintas clases. El destino de la excursión tiene un aforo máximo.

Vuestro objetivo es decidir qué clases asisten a la excursión, de forma que el total de estudiantes que van no excedan el aforo máximo, pero asistan tantos estudiantes como sea posible a la excursión. Por último, si se decide que una clase asiste a la excursión, asisten todos los estudiantes de dicha clase.

Datos de entrada 1

Se os facilitan distintos datasets para organizar distintas excursiones, debéis resolver el problema propuesto para cada uno de ellos.

Formato de los ficheros

En la primera línea encontramos los siguientes datos:

- Un entero M ($1 \leq M \leq 10^9$) - Este representa el máximo aforo de la excursión.
- Un entero N ($1 \leq M \leq 10^5$) - Este representa el número total de clases.

En la segunda línea encontramos N enteros ordenados de forma creciente, estos representan los estudiantes de cada una de las clases.

$1 \leq S_0 \leq S_1 \leq \ldots \leq S_{N-1} \leq M$

Ejemplo de fichero de entrada:

```
17 4
2 5 6 8
```

(Es decir, el aforo es 17 personas y la cantidad de clases es 4. Los estudiantes que hay en cada clase son: 2 estudiantes en la clase 0, 5 estudiantes en la clase 1, 6 estudiantes en la clase 2 y 8 estudiantes en la clase 3).

Soluciones 1

Formato del fichero de resultados

En la primera línea debe estar el número X que representa el total de clases que se ha decidido que asistan a dicha excursión.
En la segunda línea se deben incluir los índices que representan las clases que asisten a la excursión de las X clases seleccionadas, respecto a los datos de entrada originales, separadas entre sí por un espacio.

Ejemplo de fichero de salida:

```
3
0 2 7
```

(Es decir, 3 clases: la 0, la 2 y la 7).

Puntuación

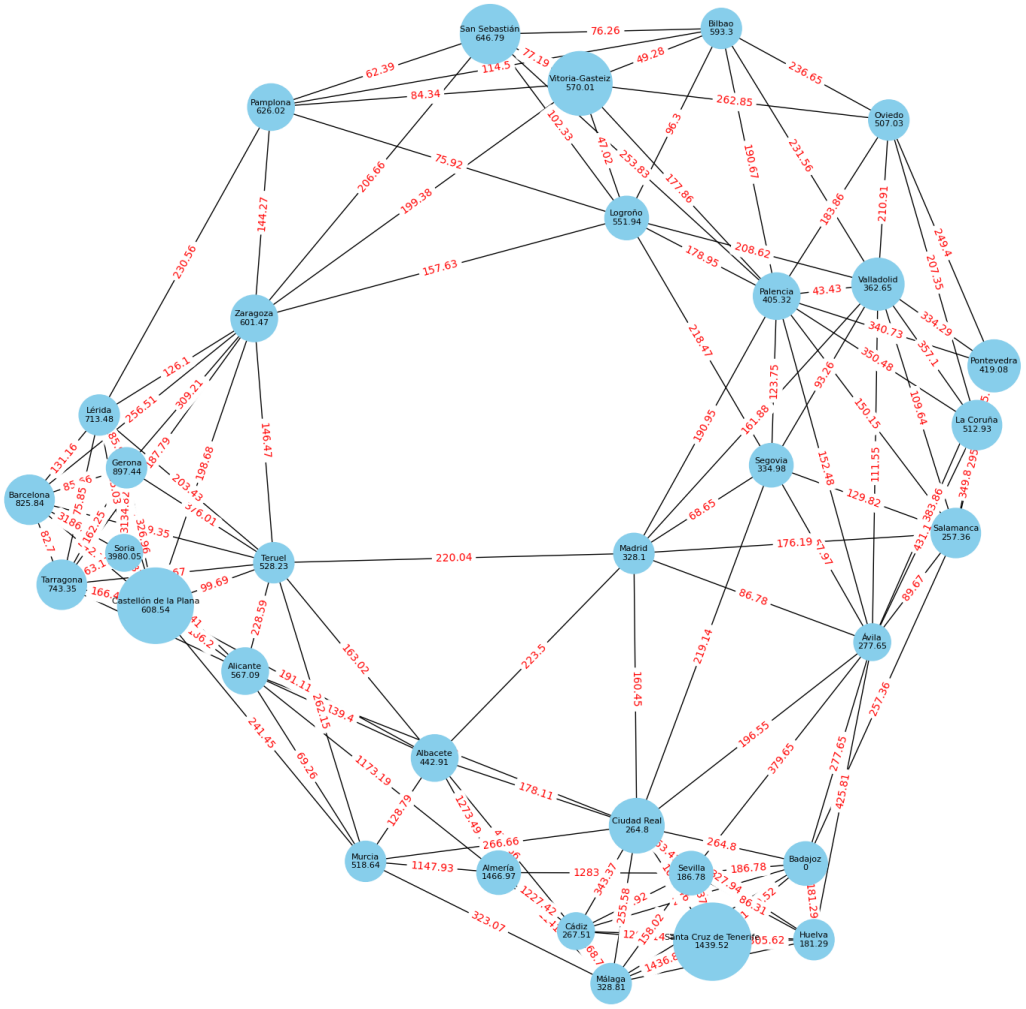
La puntuación total de cada equipo será el total de estudiantes que acuden a cada excursión. El resultado de cada excursión solo se añadirá al total **solo si es una solución válida** (no más asistentes de los que caben, no somos Ryanair).

Descripción del problema 2

Una vez elegidas las clases que van a asistir a las excursiones, se debe planificar la ruta más corta de la misma. Las excursiones van desde la EINA en Zaragoza hasta Ingenieros Libres SL en Badajoz (es decir, no hay tren).

Para ello contáis con un listado de ciudades relevantes, cada una asociada con una distancia estimada desde dicha ciudad a la ciudad de destino. También contáis con las carreteras que conectan las ciudades y su kilometraje exacto.

Vuestro objetivo es decidir cuál es la ruta más corta para realizar la excursión. Aquí se adjunta una representación del mapa en forma de grafo:



Datos de entrada 2

Se os facilitan dos ficheros de texto, el primero de ellos contiene un listado con las ciudades y la distancia estimada de dicha ciudad hasta Badajoz. El segundo de ellos contiene las carreteras que unen cada par de ciudades y su distancia exacta.

Formato de los ficheros

Fichero de ciudades:

```
Vitoria-Gasteiz;570.01
Albacete;442.91
....
```

(Es decir Albacete está a 442.91Km de Badajoz **en línea recta**).

IMPORTANTE: **no es la distancia a través de carreteras**, es la distancia en línea recta. Como no vale llegar en avión a Badajoz este dato podría parecer inutil... ¿o no?

Fichero de carreteras:

```
Vitoria-Gasteiz;Logroño;47.01891187073324
Vitoria-Gasteiz;Bilbao;49.282117699052705
....
```

(Es decir, la carretera de Vitoria a Logroño mide 47.01Km).

Soluciones 2

Restricciones

Dado el tamaño pequeño del problema, al no ser un dataset tan grande como el del primer problema queda completamente PROHIBIDO el uso de un algoritmo de fuerza bruta (probar todas las combinaciones sin criterio alguno).

Formato del fichero de resultados

En la primera línea debe estar el número X que representa el total de kilómetros de carretera exactos que se han de recorrer, este valor debe tener toda la precisión decimal posible (por si hay empate).
En la segunda línea se deben incluir los nodos que conforman la solución de forma ordenada(siguiendo la ruta de vuestra solución).

Ejemplo de fichero de salida:

```
6666.282117699052705
Zaragoza;Cuenca;Málaga;Badajoz
```

Puntuación

La puntuación total de cada equipo será calculada mediante la siguiente fórmula:

$$score(d) = \max\left\{\frac{-s_{max}}{d_{max}}d + s_{max}, 0\right\}$$

Donde la puntuación máxima (s_max) es 250000000 y la distancia máxima (d_max) es la suma de las distancias de todas las aristas. d es tu distancia.

Puntuación final global

La puntuación final será la suma de las puntuaciones obtenidas individualmente en cada problema. Como el segundo problema es más complejo, el primer problema tendrá mayor peso en la evaluación final.

Solución

Los resultados se subirán a la página web de Unicode 23, que podéis encontrar en https://codelabzgz.duckdns.org. Podéis registraros en la página web con una cuenta de Discord, Google o GitHub ya existente. Si no tenéis cuenta o preferís crear una nueva, podéis registraros con un email.

Para validar vuestra solución, pinchad en uno de los problemas y podréis subir los ficheros de resultados para ver vuestra puntuación, o en caso de que vuestra solución no sea válida, una pista sobre el error cometido. También podéis ver un ranking de los participantes con sus puntuaciones en cada problema.

Finalmente, si necesitáis una base de código, tenéis disponible varias plantillas en varios lenguajes en el repositorio base de Unicode 23, que se encuentra en https://github.com/CodeLabZGZ/base-unicode-23. También están disponibles todos los conjuntos de datos de entrada necesarios.