MATCOM

Grupo Weboo
Universidad de La Habana

2do Examen de Programación

Curso 2021-2022

TuEnvío

NOTA: Si usted está leyendo este documento sin haber extraído el compactado que se le entregó, ciérrelo ahora, extraiga todos los archivos en el escritorio, y siga trabajando desde ahí. Es un error común trabajar en la solución dentro del compactado, lo cual provoca que los cambios no se guarden. Si usted comete este error y entrega una solución vacía, no tendrá oportunidad de reclamar.

Tu Envío, plataforma líder del comercio electrónico nacional, recibe cada día decenas de pedidos. Los administrativos de la empresa dedican todos sus esfuerzos a que estos pedidos sean entregados a sus respectivos destinos en el menor tiempo posible. Cada día llega al Almacén Central (**AC**) de la empresa, de donde salen todas las órdenes, un camión destinado por la empresa para realizar las entregas del día.

Cada mañana, Roberto, el chofer del camión, debe elaborar la hoja de ruta, la cual no es más que el cronograma de los viajes que va a realizar durante el día y el estimado del consumo de combustible. Para ello la empresa pone a su disposición toda la información necesaria:

1. Se conoce **cuántas órdenes** se deben entregar en el día $\{ O_1, O_2, O_3, \dots , O_n \}$ y el **peso de cada una** $\{ P_1, P_2, P_3, \dots , P_n \}$:

Orden	O_1	O_2	…	O_n
Peso	P_1	P_2		P_n

Fig. 1 Peso en kg de cada orden

2. Además, se tiene la información de los destinos $\{ D_1, D_2, D_3, \dots , D_n \}$ donde deben ser entregados los pedidos y el **consumo de combustible** de ir desde un lugar X a otro lugar Y:

	AC	D_1	D_2	…	D_n
AC	$C_{AC,AC}$	$C_{AC,1}$	$C_{AC,2}$		$C_{AC,n}$
D_1	$C_{1,AC}$	$C_{1,1}$	C_{1-2}		$C_{1,n}$
D_2	$C_{2,AC}$	$C_{2,1}$	$C_{2,2}$		$C_{2,n}$
…				…	
D_n	$C_{n,AC}$	$C_{n,2}$	$C_{n,2}$		$C_{n,n}$

Fig. 2 Relación de consumo de combustible

3. Por último, la empresa también le dice a Roberto el **peso máximo (PM)** que puede soportar el camión que estará manejando durante el día.

Fig. 3 Peso máximo que puede trasladar el camión

Con toda esta información Roberto podrá conformar la hoja de ruta, la cual será verificada al final el día teniendo en cuenta la limitante de que se haya consumido **la menor cantidad de combustible posible**. Como te podrás imaginar, la conformación de la hoja de ruta y, particularmente, el cálculo de la menor cantidad de combustible que pudiera gastar Roberto durante el día, son procesos de gran dificultad y repetitivos, por lo que la empresa necesita de tu ayuda para automatizar esta tarea.

Usted debe haber recibido junto a este documento una solución de C# con dos proyectos: una biblioteca de clases (*Class Library*) y una aplicación de consola (*Console Application*). Deberá implementar el método `CombustibleDiario` que se encuentra en la clase `TuEnvio` en el *namespace* `Weboo.Examen`. En la biblioteca de clases encontrará la siguiente definición:

```
namespace Weboo.Examen
{
    public class TuEnvio
    {
        public static int CombustibleDiario(
            int[] pesos, int[,] combustible
        )
        {
            //Borre la siguiente línea y escriba su código
            throw new NotImplementedException();
        }
    }
}
```

El método recibirá como entrada:

- `int[] pesos`: Definición de los pesos. Array de `[N+1]` elementos
  - La posición `0 (cero)` representa el peso máximo de que puede cargar el camión
  - El resto de las posiciones `(1, 2,…, n)` contienen el peso de cada una de las órdenes
- `int[,] combustible`: Relación del consumo de combustible. Matriz de `[N+1, N+1]`
  - La posición `[i,j]` indica el combustible que se consumirá de ir del lugar `i` al lugar `j`
  - La fila `0 (cero)` y la columna `0 (cero)` muestran la información del Almacén Central.
  - El resto de las filas y columnas representan los destinos de cada orden `(1, 2,…, n)`.

El método deberá retornar:

- `int`: El cálculo de la **menor cantidad de combustible** con que se puede hacer la entrega de todas las órdenes y regresar al almacén. De no se posible de hacer la entrega de las mismas, se deberá retornar `-1`.

A continuación, se muestra cómo quedarían representados los datos para un día en particular:

Entrada:

`int[] pesos`:

PM	O_1	O_2	O_3	O_4
20	15	10	13	17
0	1	2	3	4

`int[,] combustible`:

	AC	D_1	D_2	D_3	D_4	
AC	0	10	5	1	2	0
D_1	10	0	3	3	4	1
D_2	5	3	0	2	5	2
D_3	1	3	2	0	15	3
D_4	2	4	5	15	0	4
	0	1	2	3	4	

Salida:

```
10 + 10 + 5 + 5 + 1 + 1 + 2 + 2 = 36

AC -> D1 -> AC -> D2 -> AC -> D3 -> AC -> D4 -> AC
----------  -----------  ----------  ----------
  Viaje1       Viaje2       Viaje3      Viaje4
```

- Nota que Roberto no siempre se podrá llevar todas las órdenes en un solo viaje, o sea, **puede dar tantos viajes como sea necesario**. En este caso debe hacer un viaje por cada orden.

NOTA: Todo el código de la solución debe estar en este proyecto (biblioteca de clases), pues es el único código que será evaluado. Usted puede adicionar todo el código que considere necesario, pero no puede cambiar los nombres del namespace, clase o método mostrados. De lo contrario, el probador automático fallará. En particular, es imprescindible que usted no cambie los parámetros del método `CombustibleDiario`. Por supuesto, usted puede (y debe) adicionar todo el código que necesite.

NOTA: Los casos de prueba que aparecen en este proyecto son solamente de ejemplo. Que usted obtenga resultados correctos con estos casos no es garantía de que su solución sea correcta y de buenos resultados con otros ejemplos. De modo que usted debe probar con todos los casos que considere convenientes para comprobar la validez de su implementación.