Algoritmia básica
3^o Grado en Ingeniería Informática
Curso 2022/23

Práctica 4: Ramificación y poda, programación lineal

1. Consideraciones generales

- La entrega de la práctica se realizará en *Moodle* a través de la tarea *Entrega práctica 4*
- Se entregará un fichero `practica4.zip` que contiene un directorio denominado `practica4_NIA1_NIA2` (siendo NIA1 y NIA2 los números identificadores de cada estudiante asignados por la Universidad de Zaragoza, y NIA1 será el NIA menor. En el caso de un grupo de práctica formado por un único alumno, el directorio tendrá como nombre `practica4_NIA` (con el identificador de ese alumno).
- El directorio incluirá los siguientes ficheros de texto:
  - Descripción general del directorio: cómo está organizado, instrucciones de instalación, compilación y ejecución, instrucciones para repetir las pruebas, etc. (tiene que llamarse `LEEME`).
  - Listados del código debidamente comentados. Deberán seguir una estructura lógica para poder encontrar y navegar adecuadamente cada una de las partes de la práctica.
  - Un programa para la *shell* denominado `ejecutar.sh` que automatice la compilación y ejecución de los programas entregados con los casos de prueba.
  - Los ficheros auxiliares de entrada necesarios para ejecutar las pruebas del punto anterior.
- El directorio incluirá también un informe con la presentación y análisis de resultados (fichero PDF, máximo 4 páginas sin portada). Indicar: nombre, apellidos y NIA de cada miembro del grupo de práctica.
- **Fechas límite de entrega para la primera convocatoria:**

Grupo	Fecha y hora
Viernes A	19/05/2023 8:00AM
Miércoles A	24/05/2023 8:00AM
Jueves A	25/05/2023 8:00AM
Miércoles B	31/05/2023 8:00AM
Jueves B	01/06/2023 8:00AM
Viernes B	02/06/2023 8:00AM

1.1. Evaluación

- En la calificación se tendrán en cuenta los siguientes aspectos: documentación, diseño e implementación, diseño de casos de prueba, análisis de las pruebas realizadas y facilidad para la repetición de las pruebas por los profesores.
- Se aplicarán las reglas de tratamiento de casos de plagio explicadas en la presentación de la asignatura.

La valoración máxima de la práctica es de **10 puntos** así repartidos:

- Tareas 1,2 y 3: **8 puntos**
- Bola extra: **2 puntos**

Para esta última práctica no se organizarán sesiones de evaluación.

2. Enunciado

La compañia *O'zbekiston* gestiona la línea ferroviaria uzbeka, a media distancia, que conecta la capital Taskent a la ciudad de Samarcanda y tiene paradas en las estaciones intermedias. Las estaciones están numeradas sucesivamente, de 0 (Taskent, estación inicio de línea) a m (Samarcanda, estación final de línea).

La compañia quiere llevar a cabo un experimento para mejorar la capacidad de transporte de pasajeros y al mismo tiempo maximizar los ingresos. Los trenes utilizados en la línea tienen una capacidad máxima de n pasajeros. Antes de que el tren salga de la estación de Taskent, se recolectan los pedidos de reservas de viaje en las estaciones de la línea. Un pedido de reservas realizado por una estación $i, (i = 0, \ldots, m-1)$ incluye las reservas de viaje de la estación i hasta una estación $j (j > i, j = 1, \ldots m)$.

En caso de que la compañía no pueda aceptar todos los pedidos, debido a las limitaciones de capacidad del tren, su política es *aceptar o rechazar completamente un pedido de reserva*, es decir no es posible aceptar un pedido de forma parcial.

Tarea 1. Diseño Se pide diseñar un algoritmo **de ramificación y poda** que, dada una lista de pedidos de estaciones individuales en la línea Taskent-Samarcanda, calcule el máximo ingreso total para la compañia uzbeka. El ingreso obtenido para un pedido aceptado es el producto del número de pasajeros incluidos en el pedido y el precio del billete de viaje. El precio del billete del viaje es igual al número de paradas (estaciones) entre la estación de salida y la estación de llegada (incluida la estación de llegada). El ingreso total es la suma de los ingresos de todos los pedidos aceptados.

Tarea 2. Implementación Se pide desarrollar un programa que implemente el algoritmo de solución propuesto. La forma de ejecutar el programa será la siguiente:

```
> transporte pruebas.txt resultados.txt
```

donde `pruebas.txt` es un fichero de texto que incluye los datos de diferentes instancias del problema y `resultados.txt` es un fichero de texto que guarda los resultados. Los formatos del fichero de entrada y de salida se detallan a continuación.

Formato del fichero de entrada. El fichero de entrada está organizado en bloques: cada bloque es una instancia diferente del problema. La primera línea de cada bloque tiene tres enteros: la capacidad n del tren, el número m (máximo 7) de la estación final de línea y el número de pedidos total p (máximo 22). Las p líneas siguientes representan los pedidos. Para cada pedido

hay tres enteros: estación de salida, estación de llegada y número de pasajeros. El bloque final del fichero está compuesto por una única línea con los tres enteros igual a cero. A continuación se muestra un ejemplo de fichero de entrada que especifica dos instancia del problema:

```
10 3 5
0 2 1
0 3 2
1 3 3
1 2 4
2 3 10
10 5 6
3 5 10
2 4 9
1 3 4
0 2 5
2 5 8
3 4 2
0 0 0
```

La capacidad del tren es la misma en las dos instancias ($n = 10$), en la primera hay $m = 3$ estaciones y $p = 5$ pedidos mientras en la segunda hay $m = 5$ estaciones y $p = 6$ pedidos.

Formato del fichero de salida. El fichero de salida tiene un número de líneas igual al número de bloques del fichero de entrada, excepto el bloque final. Cada línea incluye dos números: el valor del máximo ingreso total y el tiempo de ejecución. A continuación se muestra un ejemplo de fichero de salida con las soluciones de las dos instancias especificadas en el fichero de entrada anterior (el tiempo en este caso es en milisegundos):

```
18.0 5.539894104003906
38.0 3.2529830932617188
```

Tarea 3. Experimentación Analizar la corrección y eficiencia (tiempo de ejecución) del algoritmo implementado a través de un conjunto de pruebas.

Bola extra. Esta tarea es **opcional** y consiste en:

1. Formalizar el problema en un problema de programación lineal (entera) paramétrico.
2. Implementar el problema paramétrico y resolver las instancias del problema utilizando las API proporcionadas por una herramientas de programación lineal de libre elección.
3. Utilizar la herramienta de programación lineal para validar la implementación del algoritmo de ramificación y poda (comparando las soluciones obtenidas y los tiempos de ejecución de las dos técnicas).

2.1. Bibliografía → en Moodle

1. Transparencias de la asignatura.
2. Las páginas web *Bibliografía de referencia* en los apartados *Ramificación y poda* y *Programación lineal y reducciones*.
3. Para la bola extra: la página web *Herramientas de programación lineal* en el apartado *Programación lineal y reducciones*.