Universidad Rey Juan Carlos

Práctica 2: Divide y Vencerás

Diseño y Análisis de Algoritmos

Grado en Ingeniería Informática

- Valor: 10 % de la nota final
- Los códigos tendrán que probarse con **Mooshak**
  - `http://gibson.escet.urjc.es/~mooshak`
  - Registrarse en Mooshak:
    - Seleccionar la práctica DAA_13-14_Pr02_*campus* del campus que os corresponda
    - El nombre debe tener el formato "NombreApellido1Apellido2", por ejemplo: `ManuelMunozSanchez` (todo junto, con iniciales en mayúsculas, sin tildes ni eñes)
    - El grupo es el asociado a la titulación y número de expediente del alumno
- Grupos: individual
- Carácter: obligatoria
- Debéis subir los códigos fuente tanto a Mooshak como al campus virtual
- Los ejercicios deben ser aceptados por Mooshak para poder puntuar
- Fecha límite: 27 de marzo de 2014 a las 23:00

Índice

1. Recolectar madera [5 %]

1.1. Introducción

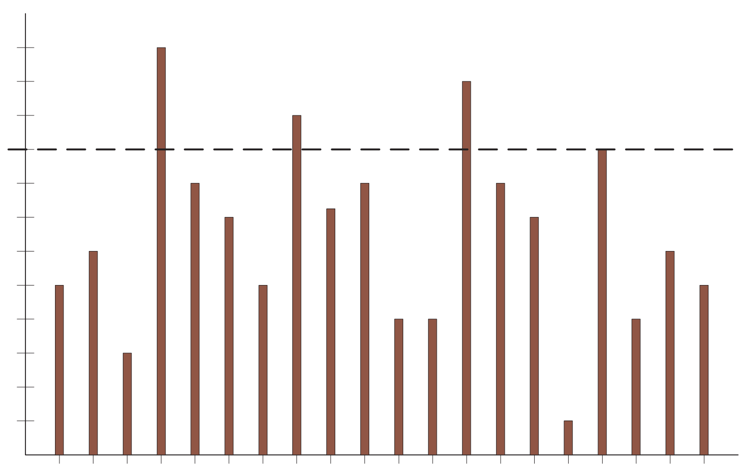
En este ejercicio se plantea un problema donde tendréis que buscar un enfoque eficiente basado en la estrategia de "decrementa y vencerás".

1.2. Enunciado del problema

Un leñador tiene una máquina bastante curiosa para cortar árboles. Se posiciona a una cierta altura $H \in \mathbb{N}$ y de ahí en adelante corta a lo largo de dicha altura todo lo que se encuentra a su paso. De esta manera, el leñador recolecta toda la madera que ha cortado la máquina por encima de la altura H.

Cuando el leñador necesita al menos k unidades de madera, debe situar la máquina a la altura más alta posible, llamémosle H, de manera que recolecte al menos esa cantidad k de madera. Si escogiese una altura mayor no llegaría a recolectar k unidades, mientras que a una altura menor recolectaría más madera de la necesaria, que no cabría en su almacén.

La siguiente imagen ilustra cómo corta la máquina (en este caso para recolectar 6 unidades de madera):



Formalmente, suponiendo que hay n árboles, y que el árbol i-ésimo tiene altura $h_i \in \mathbb{N}$, para $i = 1, \ldots, n$, el problema de optimización es:

$$\begin{array}{ll} \textit{maximizar} & H \\ \text{sujeto a} & \displaystyle\sum_{i=1}^{n} g(h_i - H) \geqslant k \\ & H \in \mathbb{N} \end{array}$$

Donde $g(x) = x$ si $x > 0$, y $g(x) = 0$ si $x \leqslant 0$.

1.2.1. Descripción de la entrada

La primera línea contiene dos enteros n 1 ($\leqslant n \leqslant$ 50,000) y k ($1 \leqslant k \leqslant 10^9$), separados por un espacio en blanco. La segunda línea contiene las alturas h_i ($1 \leqslant h_i \leqslant 10^9$) de los n árboles para $i = 1, \ldots, n$ (en ese orden). Se asume que el bosque de árboles siempre tendrá más madera de la que necesita el leñador. Es decir,

$$\sum_{i=1}^{n} h_i \geqslant k.$$

1.2.2. Descripción de la salida

Deberá imprimir el entero H, seguido de un salto de línea.

Ejemplo de entrada 1

```
4␣7↵
20␣15␣10␣17↵
```

Salida para el ejemplo de entrada 1

```
15↵
```

Ejemplo de entrada 2

```
5␣20↵
4␣42␣40␣26␣46↵
```

Salida para el ejemplo de entrada 2

```
36↵
```

2. Producto de matrices [5 %]

En este ejercicio el objetivo es implementar una multiplicación de matrices particular empleando la estrategia de “divide y vencerás”.

2.1. Problema a implementar

Sea una matriz $\mathbf{A} \in \mathbb{Z}^{p,q}$, y otra $\mathbf{B} \in \mathbb{Z}^{q,r}$, cuyas dimensiones son $(p \times q)$ y $(q \times r)$, respectivamente, donde la máxima dimensión es 10 (es decir, $1 \leqslant p \leqslant 10$, $1 \leqslant q \leqslant 10$, y $1 \leqslant r \leqslant 10$). Ambas contienen números enteros. Se desea obtener el producto de éstas:

$$\mathbf{A} \cdot \mathbf{B} = \mathbf{C}$$

Donde $\mathbf{C} \in \mathbb{Z}^{p,r}$ es la matriz resultante del producto.

Para calcular el producto de las matrices el algoritmo descompondrá las matrices por bloques, **obligatoriamente**, de la siguiente manera:

$$\mathbf{A} \cdot \mathbf{B} = \left(\frac{\mathbf{A}_1}{\mathbf{A}_2} \right) \cdot \left(\begin{array}{c|c} \mathbf{B}_1 & \mathbf{B}_2 \end{array} \right) = \begin{pmatrix} \mathbf{A}_1\mathbf{B}_1 & \mathbf{A}_1\mathbf{B}_2 \\ \mathbf{A}_2\mathbf{B}_1 & \mathbf{A}_2\mathbf{B}_2 \end{pmatrix} = \mathbf{C}$$

Sugerencia: para implementar el algoritmo se pueden declarar matrices de tamaño fijo (10×10) en memoria, aunque luego no lleguen a llenarse completamente. De esta manera, se pueden usar índices para indicar qué submatriz se está utilizando realmente.

2.1.1. Descripción de la entrada

La entrada contiene, en su primera línea, tres enteros (de valores entre 1 y 10) correspondientes a las dimensiones p, q y r de las matrices $\mathbf{A}$ y $\mathbf{B}$. Posteriormente se especifican las matrices. Para la matriz $\mathbf{A}$ habrá p nuevas filas, cada una con q enteros. A continuación, para la matriz $\mathbf{B}$, habrá q nuevas filas, cada una con r enteros. Todos los enteros de una línea están separados por un espacio en blanco.

2.1.2. Descripción de la salida

La salida contendrá el producto $\mathbf{C} = \mathbf{A} \cdot \mathbf{B}$. Por tanto, tendrá p filas, cada una con r enteros. Todos los enteros de una línea están separados por un espacio en blanco. Después del último entero de cada línea habrá un salto de línea.

2.1.3. Ejemplo de entrada

```
3 5 4
-3 4 1 6 9
2 -1 -4 0 -2
1 1 4 -2 3
1 3 -2 -5
0 2 1 2
-4 -5 -3 -2
1 3 4 2
0 0 3 0
```

2.1.4. Salida para el ejemplo de entrada

```
-1 12 58 33
18 24 1 -4
-17 -21 -12 -15
```