ISIS 1105 Diseño y Análisis de Algoritmos
Semestre 2021-10. Proyecto
Profs. R. Cardoso / J. Duitama
Entrega: Viernes, Junio 4 de 2021, 22:00 (por Sicua+)
Trabajo individual o por parejas (de la misma sección)

0 OBJETIVOS

- Diseñar soluciones computacionales para problemas.
- Estimar costos de las soluciones planteadas.
- Implementar soluciones en *Java*.

Se premiarán las mejores soluciones y se castigarán las peores, en cuanto a eficiencia en tiempo y espacio.

1 CONDICIONES GENERALES

Hay dos problemas para resolver mediante soluciones implementadas en *Java*.

Para cada problema se pide:
- Análisis temporal y espacial.
- Una solución java.

2 PROBLEMAS

A Potencia matricial en $\mathbf{Z}_m$

Para m**:nat**$^+$, se define $\mathbf{Z}_m = 0..m-1$. Se comprueba que $(\mathbf{Z}_m, +_m, *_m, 0, 1)$ es un semianillo, donde $+_m$ y $*_m$ son la suma y la multiplicación módulo m.

Recuérdese que, en un semianillo K, se definen potencias para sus elementos de la manera usual. Además que, para n**:nat**$^+$, $\mathbf{M}_n(K)$ es el conjunto de la matrices cuadradas de $n \times n$ con entradas en el semianillo K, y que $(\mathbf{M}_n(K), +_n, *_n, 0_n, I_n)$ es -a su vez- un semianillo.

Problema

Dada una matriz en $A \in \mathbf{M}_n(\mathbf{Z}_m)$, y un número natural r, calcular la matriz potencia $A^r \in \mathbf{M}_n(\mathbf{Z}_m)$.

Ejemplos:
Sea $A \in \mathbf{M}_2(\mathbf{Z}_5)$, definida como

$$A = \begin{bmatrix} 2 & 3 \\ 4 & 0 \end{bmatrix}$$

Entonces:

$$A^3 = \begin{bmatrix} 1 & 3 \\ 4 & 4 \end{bmatrix}$$

B Balanceo de carga

La carga de un centro de servicios se reparte entre p procesadores, $p>0$. Cuando se va a asignar la carga, se asignan n trabajos , $n>0$, a los p procesadores. Para hacerlo, se cuenta con una lista $t = \langle t_1, t_2, \ldots, t_n \rangle$, de modo que t_i**:nat** es el tiempo estimado que cualquier procesador debe consumir para el trabajo i, $1 \leq i \leq n$.

Una *asignación* a (de trabajos a los p procesadores) se hace adjudicando un bloque contiguo de la lista de trabajos a cada uno de ellos. Ningún trabajo se asigna a más de un procesador y todos los trabajos terminan siendo asignados. Si al procesador j le corresponde el bloque subindicado con el intervalo $j_r..j_s \subseteq 1..n$, se define $cp.j$, la *carga del procesador* j, como la suma de los tiempos estimados de los trabajos asignados, i.e., $cp.j = (+k \mid r \leq k \leq s : t_k)$.

La *carga de la asignación* a, $c.a$, se define como la carga más grande de las asignadas, según a, a los diferentes procesadores, i.e., $c.a = (\max j \mid 1 \leq j \leq p : cp.j)$.

Problema

Dados n,p y la lista t, determinar $camin$, la *carga de asignación mínima* entre todas las posibles asignaciones de los n trabajos para los p procesadores. Note que $camin$ es una cota inferior estimada para el tiempo de terminación del proceso.

Ejemplos:

[E1] $n = 9,\ p = 3,\ t = \langle 1,1,1,1,1,1,1,1,1 \rangle$.
Si el trabajo se asigna: $\langle 1 \mapsto 1..3,\ 2 \mapsto 4..6,\ 3 \mapsto 7..9 \rangle$.
Entonces: $cp.1 = cp.2 = cp.3 = 3$, de modo que $ca = 3$.
También debe ser claro que $camin = 3$.

[E2] $n = 9,\ p = 3,\ t = \langle 1,2,3,4,5,6,7,8,9 \rangle$.
Si el trabajo se asigna: $\langle 1 \mapsto 1..3,\ 2 \mapsto 4..6,\ 3 \mapsto 7..9 \rangle$.
Entonces: $cp.1 = 6;\ cp.2 = 15;\ cp.3 = 24$, de modo que $ca = 24$.
Si el trabajo se asigna: $\langle 1 \mapsto 1..5,\ 2 \mapsto 6..7,\ 3 \mapsto 8..9 \rangle$.
Entonces: $cp.1 = 15;\ cp.2 = 13;\ cp.3 = 17$, de modo que $ca = 17$.
Se puede comprobar que ésta es la carga de asignación mínima, de modo que $camin = 17$.

3 ENTRADA / SALIDA DE DATOS

En todas las soluciones que se presenten, la lectura de los datos de entrada se hace por la entrada estándar; así mismo, la escritura de los resultados se hace por la salida estándar.

Puede suponer que ninguna línea de entrada tiene espacios al principio o al final, y que los datos que se listan en cada línea están separados por exactamente un espacio.

A continuación, para cada problema, se establecen parámetros que definen su tamaño y formato de lectura de los datos, tanto de entrada como de salida.

3.1 Problema A: Potencia matricial en $\mathbb{Z}_m$

- Tamaño del problema: n, r, m
- Condiciones de los casos de prueba: $0<n\leq 50$, $0\leq r<1000$, $2\leq m<1000$

Descripción de la entrada

La entrada contiene varios casos de prueba. Cada caso de prueba describe una matriz A para la que debe calcularse una potencia. El caso comienza con línea de texto de la forma
$n\ r\ m$

donde n, r y m son un números naturales positivos que corresponden, respectivamente, al lado de la matriz A, la potencia a la que ésta debe ser elevada y el módulo para operaciones aritméticas.

Enseguida hay n líneas, cada una con n valores en $\mathbf{Z}_m$, de manera que la i-sima línea corresponde a la fila i de la matriz $A \in \mathbf{M}_n(\mathbf{Z}_m)$.

El fin de la entrada se indica con una línea que no debe procesarse, que contiene tres valores 0:
$0\ 0\ 0$.

Descripción de la salida

Por cada caso de prueba para resolver, imprimir n líneas de respuesta que contengan las filas de la matriz potencia correspondiente al dato leído, i.e., A^r.

Ejemplos de entrada / salida

Entrada	Salida
2 3 5 2 3 4 0 2 4 7 2 3 4 0 0 0 0	1 3 4 4 3 0 0 3

3.2 Problema B: Balance de carga

- Tamaño del problema: n, p
- Condiciones de los casos de prueba: $1 \leq p \leq n \leq 10^2$.

Descripción de la entrada

La entrada contiene varios casos de prueba. Cada caso de prueba comienza con una línea que contiene dos números naturales positivos:
$n\ p$

cuyos valores corresponden al número de trabajos y de procesadores, respectivamente. Enseguida hay una línea con valores enteros positivos de la forma
$t_1\ t_2\ \ldots\ t_n$

donde el valor t_i corresponde al tiempo estimado para realizar el trabajo i, $0 < t_i < 1000$, $1 \leq i \leq m$.

El fin de la entrada se indica con una línea que contiene dos ceros, i.e.,

$0\ 0$

Descripción de la salida

Para cada caso de prueba se debe imprimir una línea con el valor correspondiente al valor de la carga mínima de asignación para los datos definidos en la entrada.

Ejemplo de entrada / salida

Entrada	Salida
9 3 1 1 1 1 1 1 1 1 1 9 3 1 2 3 4 5 6 7 8 9 0 0	3 17

4 ENTREGABLES

El proyecto puede desarrollarse por grupos de uno o dos estudiantes de la misma sección. La entrega se hace por Sicua+ (una sola entrega por grupo de trabajo).

El grupo debe entregar, por Sicua+, un archivo de nombre `proyectoDAlgo.zip`. Este archivo es una carpeta de nombre `proyectoDAlgo`, comprimida en formato `.zip`, dentro de la cual hay archivos fuente de soluciones propuestas y archivos que documentan cada una de las soluciones.

4.1 Archivos fuente de soluciones propuestas

Todos los programas implementados en *Java* deben compilar en *JDK 8*.

Para el problema `X`, siendo `X∈{A,B}`:

- Entregar un archivo *Java* (`.java`) con su código fuente de la solución que se presenta.
- Incluir como encabezado de cada archivo fuente un comentario que identifique el (los) autor(es) de la solución.
- Denominar `ProblemaX.java` el archivo de la solución que se presente.

Nótese que, si bien puede utilizarse un *IDE* como *Eclipse* durante el desarrollo del proyecto, la entrega requiere incluir solo un archivo `.java` por cada solución. El archivo debe poderse compilar y ejecutar independientemente (sin depender de una estructura de directorios, de un IDE, de librerías no estándar, etc.).

4.2 Archivos que documentan soluciones propuestas

Toda solución propuesta debe acompañarse de un archivo que la documente, con extensión `.doc`, `.docx` o `.pdf`. El nombre del archivo debe ser el mismo del código *Java* correspondiente. Por ejemplo, si incluyó un archivo `ProblemaB.java`, como solución para el problema B, debe incluirse un archivo `ProblemaB.docx` (o `ProblemaB.pdf`) que lo documente.

Un archivo de documentación debe contener los siguientes elementos:

0 Identificación
Nombre de autor(es)
Identificación de autor(es)

1 Algoritmo de solución
Explicación del algoritmo elegido. Si hubo alternativas de implantación diferentes, explicar por qué se escogió la que se implementó.
Deseable:
Anotación (contexto, pre-, poscondición, …) para cada subrutina o método que se use.

2 Análisis de complejidades espacial y temporal
Cálculo de complejidades y explicación de las mismas. Debe realizarse un análisis para cada solución entregada.

3 Comentarios finales
Comentarios al desempeño observado de la solución.

Téngase en cuenta que los análisis de *2* tienen sentido en la medida que la explicación de *1* sea clara y correcta. No se está exigiendo formalismo a ultranza, pero sí que, como aplicación de lo estudiado en el curso, se pueda describir un algoritmo de manera correcta y comprensible.

No describa un algoritmo con código GCL a menos que lo considere necesario para explicarlo con claridad. Si se consideran ciclos no triviales, sí deben anotarse invariantes y cotas que sirvan para probar la corrección (pero la demostración de la corrección no es obligatoria). En cualquier caso, asegúrese de incluir aserciones explicativas, fáciles de leer y de comprender.

Recuerde que usar ejemplos para explicar sus soluciones no es lo que se pide. Tampoco que el código sea la única documentación y que quien lea su explicación deba "ejecutar el programa". Solo que sea documentación clara y suficiente para demostrar tanto corrección como eficiencia.