ISIS 1105 Diseño y Análisis de Algoritmos
Semestre 2019-20. Proyecto
Profs. R. Cardoso / J. Duitama
Entrega: Martes, Diciembre 10 de 2019, 8:00 (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 tres problemas para resolver mediante soluciones implementadas en *Java*.

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

2 PROBLEMAS

A Subarreglo recurrente más largo

Dado un arreglo `a[0..N-1]:int`, defínase, para $0 \le p,q \le N$, el predicado

$$SR(p,q) \equiv (\exists d \mid 0<d \le p : (\forall k \mid p \le k < q : a[k-d]=a[k])).$$

Cuando $SR(p,q)$ vale, se dice que el subarreglo `a[p..q-1]` es un *subarreglo recurrente* de longitud $\max(0,q-p)$.

Problema

Desarrollar un programa que, dado un arreglo de enteros, determine la longitud máxima dentro de todos los subarreglos recurrentes.

Ejemplo:

Para

```
a = [0,1,2,1,4,1,2,1,0,5]
```

son recurrentes los subarreglos `a[2..1]`, `a[8..8]`, `a[5..7]`, `a[5..6]`, entre otros. El subarreglo recurrente más largo mide 3.

B Contraejemplo maximal

En muchas investigaciones científicas, estudiando características medibles A y B, se llega a hipótesis que afirman correlaciones como

" individuos que tienen más A deben tener más B".

Por ejemplo:
- los perros más grandes consumen más comida;
- las personas más gordas son más propensas a un infarto;
- las frutas más largas contienen más agua;
- etc.

Una afirmación como éstas puede o no ser cierta. Supóngase, por el contrario, que se sospecha que, en un caso particular, se está frente a una afirmación falsa. Para comprobarlo, se ha hecho una muestra de mediciones para la población involucrada, de manera que se tiene una lista de N parejas de datos de la forma

$$\langle (a_0,b_0),(a_1,b_1),\ldots,(a_{N-1},b_{N-1})\rangle$$

donde, para el individuo k en la población, a_k y b_k son las mediciones de las características A y B, respectivamente.

Problema

Se quiere evidenciar la posible falsedad de la afirmación exhibiendo una cadena de individuos que sea creciente en la medición de A y estrictamente decreciente en la medición de B. Más aún, se quiere mostrar un *contraejemplo maximal*, en cuanto a longitud (o sea, no hay cadenas contraejemplo más largas).

En caso de que haya varios contraejemplos de tamaño máximo, se debe escoger uno de ellos, de manera determinística. Si no existe ningún contraejemplo, debe reconocerse la situación.

Ejemplo:

En las siguientes tablas, supóngase que la población se compone de 9 individuos. La primera tabla muestra las mediciones de la población; la segunda exhibe un contraejemplo; la tercera, un contraejemplo maximal, i.e., una cadena más larga de individuos que contradicen la afirmación.

Individuo	1	2	3	4	5	6	7	8	9
A	601	600	50	100	110	600	800	600	200
B	65	105	100	200	150	100	70	60	95

Individuo	4	5	9	7
A	100	110	200	800
B	200	150	95	70

Individuo	4	5	2	6	7
A	100	110	600	600	800
B	200	150	105	100	70

C Dentro o fuera

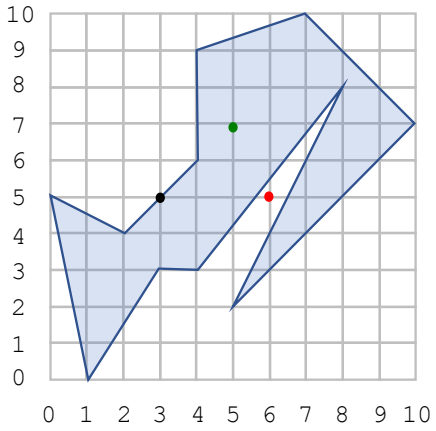
Considérese un polígono simple (lados que no se cruzan; solo los consecutivos se tocan en los vértices en común) de N vértices, en el plano cartesiano. Suponga que todos los vértices tienen coordenadas enteras no negativas y que el polígono está contenido en un cuadrado $0..M \times 0..M$, M**:nat**$^+$, que se llamará su *dominio*.

Problema

Para un punto en el dominio se quiere determinar si el punto está dentro, fuera o en la frontera del polígono.

Ejemplo

La figura muestra un polígono de `12` lados (la figura sombreada), con vértices en el dominio `0..10` × `0..10`. Además, tres puntos: uno fuera del polígono, en `(6,5)`; otro dentro de él, en `(5,7)`; y uno en la frontera, en `(3,5)`.



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: Subarreglo recurrente más largo

- Tamaño del problema: N
- Condiciones de los casos de prueba: $0<N<10^5$.

Descripción de la entrada

La entrada contiene varios casos de prueba. Cada caso de prueba se describe con una línea de texto de la forma

N

donde N es un número natural positivo, que determina el tamaño del arreglo a. Después hay una línea de la forma (cada a_k es un número entero, $0 \leq a_k < 10^6$):

$a_0\ a_0\ \ldots\ a_{N-1}$

El fin de la entrada se indica con una línea que contiene un `0`.

Descripción de la salida

Por cada caso de prueba para resolver, imprimir una línea de respuesta que contenga un número natural correspondiente al valor de la longitud del subarreglo recurrente más largo.

Ejemplos de entrada / salida

Entrada	Salida
4 1 4 2 1 10 0 1 2 1 4 1 2 1 0 5 0	1 3

3.2 Problema B: Contraejemplo maximal

- Tamaño del problema: N
- Condiciones de los casos de prueba: $1 \leq N \leq 10^4$

Descripción de la entrada

La entrada contiene varios casos de prueba. Cada caso de prueba comienza con una línea que contiene un entero positivo:
N
cuyo valor corresponde al número de individuos en la población considerada.

A continuación hay N líneas, cada una con tres números enteros. La k-sima línea tiene la forma:
$k\ a\ b$
y, en esta línea, a y b se representan -respectivamente- las mediciones de las características A y B del k-simo individuo de la población. Puede suponer que $0 \leq a, b < 10^5$.

El fin de la entrada se indica con una línea que contiene un 0.

Descripción de la salida

Para cada caso de prueba se deben imprimir líneas de la forma
M
$k_1\ k_2\ \ldots\ k_M$

donde M es la longitud de un contraejemplo maximal y $k_1, \ldots, k_M$ es una secuencia de identificadores de individuos que constituyen un contraejemplo de longitud maximal. Si hay más de un contraejemplo de longitud maximal, en la segunda línea se imprime el que sea menor, lexicográficamente, en cuanto a la secuencia $\langle k_1\ k_2\ \ldots\ k_M \rangle$.

Note que, si no hay contraejemplos, la primera línea contendría un 0 y la segunda debería ser una línea en blanco. Para asegurar que se reconoce esta situación, en este caso se imprime
0
*

Ejemplo de entrada / salida

Entrada	Salida
9 1 601 65	5 4 5 2 6 1

2 600 105 3 50 100 4 100 200 5 110 150 6 600 100 7 800 70 8 600 60 9 200 95 3 1 100 50 2 300 70 3 200 60 4 1 601 65 2 600 105 3 50 100 4 100 200 0	0 * 3 4 2 1

3.3 Problema C: Dentro o fuera

- Tamaño del problema: M, N
- Condiciones de los casos de prueba: $1 \leq M < 10^3$, $3 \leq N < M^2$

Descripción de la entrada

La entrada contiene varios casos de prueba. Cada caso de prueba se describe con una línea de texto de la forma (enteros separados por un blanco):
`M N a b`

donde M determina el dominio del problema, N es el número del lados del polígono y (a,b) son las coordenadas del punto del dominio que se quiere evaluar. Enseguida hay una línea con $2*N$ números naturales, de la forma (enteros separados por un blanco):
x_1 y_1 x_2 y_2 … x_N y_N

El fin de la entrada se indica con una línea que no se procesa, de la forma
`0 0 0 0`

Descripción de la salida

La salida debe contener una línea con exactamente un entero en $\{-1,0,1\}$, dependiendo de que el punto (a,b) esté fuera (-1), en la frontera (0) o dentro del poligono (1).

Ejemplo de entrada / salida

Entrada	Salida
6 3 2 4 1 1 2 5 6 3 6 3 5 1 1 1 2 5 6 3 10 11 6 5 3 3 4 3 8 8 5 2 10 7 7 10 4 9 4 6 2 4 0 5 1 0 10 11 5 7	1 -1 -1 1 0

3 3 4 3 8 8 5 2 10 7 7 10 4 9 4 6 2 4 0 5 1 0 10 11 3 5 3 3 4 3 8 8 5 2 10 7 7 10 4 9 4 6 2 4 0 5 1 0 0 0 0 0	

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,C}`:

- 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. Y, si lo hace, asegúrese de incluir aserciones explicativas, fáciles de leer y de comprender.