ISIS 1105 Diseño y Análisis de Algoritmos
Semestre 2023-20. Proyecto – PARTE 2
Entrega: viernes, noviembre 10 de 2023 6:00PM (por Bloque Neon)

0 OBJETIVOS

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

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

1 CONDICIONES GENERALES

El proyecto se divide en tres partes independientes entre sí. Este documento describe la PARTE II. Cada parte contiene un problema a resolver mediante soluciones implementadas en `Java` o `Python`.

Para cada problema se pide:
- Descripción de la solución.
- Análisis temporal y espacial.
- Una implementación en Java o Python

2 DESCRIPCIÓN DEL PROBLEMA

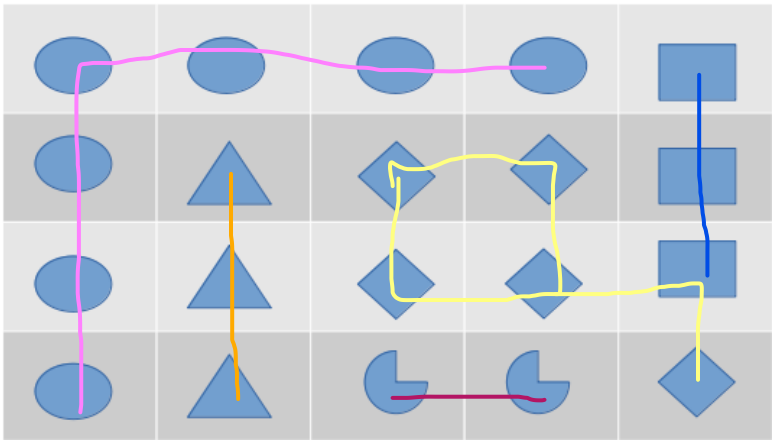
En medio del trabajo para resolver el código enigma, Alan Turing tuvo que resolver el siguiente acertijo. Se le entregó un mensaje oculto en una máquina rectangular de M filas y N columnas, en la que cada posición contiene una caja que encierra una palabra del mensaje oculto. Cada caja se puede abrir con una llave, pero varias cajas se pueden abrir con el mismo tipo de llave. Alan logra descifrar qué llave abre cada caja, pero debe abrir las cajas rápidamente porque las llaves cambian todos los días. Debido a esto, diseña una estructura que le permite abrir al mismo tiempo un subconjunto rectangular de las cajas utilizando un solo tipo de llave. En esta operación las cajas cerradas para las cuales la llave no coincida no se modifican de ninguna forma. Sin embargo, al intentar abrir algunas máquinas, descubre que si utiliza un tipo de llave con una caja que ya está abierta, la máquina se daña y el mensaje ya no se puede recuperar. Finalmente, solo alcanza a usar su estructura una vez por cada tipo de llave.

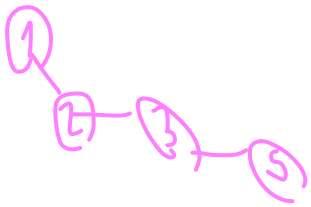
Problema

Dada una máquina de MxN con un mensaje oculto en el que k llaves abren todas las cajas, determinar si con una secuencia de aplicaciones de la estructura de Turing se puede descifrar el mensaje.

Ejemplo 1:

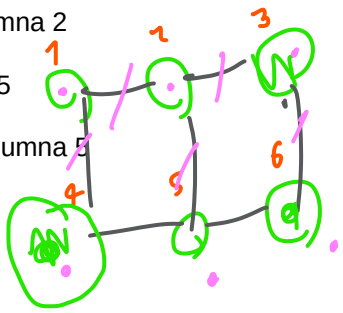
Dado M=4, N=5, k=5 y la máquina definida por la siguiente tabla:





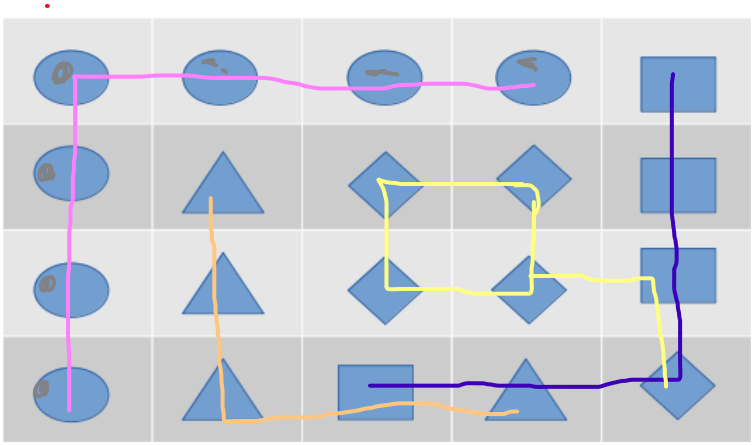
Se puede descifrar el mensaje con la siguiente secuencia de usos de la estructura de Turing:

1. Usar la llave circular de la fila 1 a la 4 y de la columna 1 a la 4
2. Usar la llave triangular de la fila 2 a la 4 y solamente en la columna 2
3. Usar la llave de rombo de la fila 2 a la 4 y de la columna 3 a la 5
4. Usar la llave rectangular de la fila 1 a la 3 y solamente en la columna 5
5. Usar la llave de Pacman en la fila 4 y d la columna 3 a la 4



Ejemplo 2:

Dado M=4, N=5, k=4 y la máquina definida por la siguiente tabla:



No se puede descifrar el mensaje sin dañar la máquina

3 ENTRADA Y 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, se establecen parámetros que definen su tamaño y formato de lectura de los datos, tanto de entrada como de salida.

Descripción de la entrada

La primera línea de entrada especifica el número de casos de prueba que contiene el archivo. El programa debe terminar su ejecución, una vez termine de resolver la cantidad de casos de prueba dados por este número.

Cada caso está representado por una línea con la especificación de M, N y k, seguida de M líneas, cada una con N números entre 1 y k separados por espacios. Cada número representa la información de cual llave abre cada caja: $2 \leq M \leq 1000$, $2 \leq N \leq 1000$, $1 \leq k \leq 10^5$

Descripción de la salida

Para cada caso de prueba, si se puede descifrar el mensaje, imprimir k líneas cada una con 5 números se parados por espacios. Cada línea debe indicar la llave a usar, de qué fila a qué fila y de qué columna a qué columna se debe usar. Las líneas deben estar ordenadas de acuerdo con la secuencia de pasos que se necesita seguir para descifrar el mensaje. En caso de no ser posible abrir el mensaje, se debe imprimir una sola línea con el mensaje “NO SE PUEDE”

Ejemplo de entrada / salida

Para los ejemplos descritos arriba:

Entrada	Salida
2 4 5 5 2 2 2 2 1 2 3 5 5 1 2 3 5 5 1 2 3 4 4 5 4 5 4 2 2 2 2 1 2 3 4 4 1 2 3 4 4 1 2 3 1 3 4	2 1 4 1 4 3 2 4 2 2 5 2 4 3 5 1 1 3 5 5 4 4 4 3 4 NO SE PUEDE

Ejemplo con tres casos de prueba:

Entrada	Salida
3 4 5 5 2 2 2 2 1 2 3 5 5 1 2 3 5 5 1 2 3 4 4 5 4 5 4 2 2 2 2 1 2 3 4 4 1 2 3 4 4 1 2 3 1 3 4 2 2 2 1 1 1 2	2 1 4 1 4 3 2 4 2 2 5 2 4 3 5 1 1 3 5 5 4 4 4 3 4 NO SE PUEDE 1 1 2 1 2 2 2 2 2 2

Nota: Se van a diseñar casos de prueba para valores de M, N y k mucho más grandes y dentro de los valores establecidos en el enunciado. Los casos mostrados en este documento son demostrativos de la estructura de entrada/salida esperada.

4 COMPRENSIÓN DE PROBLEMAS ALGORITMICOS

A continuación, se presentan un conjunto de escenarios hipotéticos que cambian el problema original. Para cada escenario debe contestar[1]: (i) que nuevos retos presupone este nuevo escenario -si aplica-?, y (ii) que cambios -si aplica- le tendría que realizar a su solución para que se adapte a este nuevo escenario?

ESCENARIO 1: Se puede aplicar la misma llave hasta dos veces

ESCENARIO 2: Se pueden diseñar estructuras en forma de ele (L).

Nota: Los escenarios son independientes entre sí.

5 ENTREGABLES

El proyecto puede desarrollarse por grupos de hasta tres estudiantes de la misma sección. La entrega se hace por bloque neon (una sola entrega por grupo de trabajo).

El grupo debe entregar, por bloque neon, un archivo de nombre `proyectoDalgoP2.zip`. Este archivo es una carpeta de nombre `proyectoDalgoP2`, comprimida en formato `.zip`, dentro de la cual hay archivos fuente de soluciones propuestas y archivos que documentan cada una de las soluciones.

[1] NO tiene que implementar la solución a estos escenarios, el propósito es meramente analítico.

5.1 Archivos fuente de soluciones propuestas

Todos los programas implementados en *Java* o en Python

Para el problema:

- Entregar un archivo de código fuente en *Java* (`.java`) o python (`.py`) 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 `ProblemaP2.java` o `ProblemaP2.py` el archivo de la solución que se presente.

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

5.2 Archivos que documentan la solución propuesta

La solución al problema debe acompañarse de un archivo de máximo 3 páginas que la documente, con extensión `.pdf`. El nombre del archivo debe ser el mismo del código correspondiente (`ProblemaP2.pdf`).

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. Generar al menos una gráfica que apoye la explicación del algoritmo implementado. No se debe copiar y pegar código fuente como parte de la explicación del algoritmo. Argumentar si el algoritmo planteado resuelve perfectamente el problema.

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

3 Respuestas a los escenarios de comprensión de problemas algorítmicos.
Respuesta a las preguntas establecidas en cada escenario. No tiene que implementar la solución a estos escenarios, el propósito es meramente analítico.

La nota del informe corresponde a un 50% de la nota total de la entrega del proyecto, solamente si se entrega el código fuente de la solución implementada. En caso de no entregar el código fuente, no se hará evaluación del informe y la nota del proyecto será cero.

Además de la pertinencia del texto como explicación de la solución implementada, se evaluará la calidad en la redacción del texto y en el diseño de las gráficas. Se evaluará también que la explicación del algoritmo integre los conceptos relacionados con las técnicas de diseño de algoritmos cubiertas en el curso. 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.