ISIS 1105 Diseño y Análisis de Algoritmos
Semestre 2023-10. Proyecto – PARTE 1
Entrega: viernes, marzo 31 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 I. 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

A Asignación equitativa de beneficiarios

Uno de los principales retos de la reforma a la salud 2023 propuesta por el gobierno actual son los denominados Centros de Atención Primaria (CAD). A cada CAD en el país serán asignados los beneficiarios y sus familias según cercanía y disponibilidad. Para que las finanzas de estos centros no se vean afectadas por una mala distribución de beneficiarios y para garantizar el mejor servicio, es necesario diseñar una estrategia de asignación equitativa de beneficiarios.

Problema

Suponga que para un atender la demanda de salud en un determinado municipio de Colombia se habilitaran k diferentes CADs a los cuales se deberán asignar m familias diferentes. La familia i-esima cuenta con f_i miembros diferentes. Se le encarga la tarea de identificar si existe una manera en que todos los CAD puedan quedar con la misma cantidad de beneficiarios, con la restricción que TODOS los miembros de un mismo grupo familiar sean asignados a un mismo CAD.

Ejemplo:

Dado $k = 3$, $m = 5$, $f_1 = 8, f_2 = 7, f_3 = 3, f_4 = 4, y f_5 = 11$.

Una forma de asignación equitativa de beneficiarios seria:

$$CAD_1 = (f_1, f_3)$$
$$CAD_2 = (f_2, f_4)$$
$$CAD_3 = (f_5)$$

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 en una línea con la especificación de k, m y los f_i. Los dos primeros números serán los valores de k y m, seguidos de los diferentes valores de f_i.

$$2 \leq k \leq 3$$

$$2 \leq m \leq 10^4$$

$$1 \leq f_i \leq 50$$

Descripción de la salida

Para cada caso de prueba, imprimir si es posible una distribución equitativa, y en caso de que si lo sea la distribución impresa como una lista de conjuntos.

Ejemplos de entrada / salida

Caso descrito arriba

Entrada	Salida
1 3 5 8 7 3 4 11	True [(8,3),(7,4),(11)]

Ejemplo con tres casos de prueba:

Entrada	Salida
3 3 5 8 7 3 4 11 2 3 4 7 9 2 4 5 7 8 4	True [(8,3),(7,4),(11)] False True [(5,7)(8,4)]

Nota: Se van a diseñar casos de prueba para valores de k, y m muchos 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: Suponga ahora que no solo deben ser equitativos en el número de beneficiarios, si no también en el número de familias.

ESCENARIO 2: Beneficiarios mayores de 60 años y menores de 5 años pueden estar asignados a todos los CAD de un municipio.

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 `proyectoDalgoP1.zip`. Este archivo es una carpeta de nombre `proyectoDalgoP1`, comprimida en formato `.zip`, dentro de la cual hay archivos fuente de soluciones propuestas y archivos que documentan cada una de las soluciones.

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 `ProblemaP1.java` o `ProblemaP1.py` el archivo de la solución que se presente.

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

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 (`ProblemaP1.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. Se recomienda generar al menos una gráfica para apoyar la explicación del algoritmo. Se recomienda no pegar código fuente en el informe como parte de la explicación del algoritmo

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.

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.