Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Introducción a la Programación y Computación 2

Inga. Claudia Liceth Rojas Morales
Ing. Marlon Antonio Pérez Türk
Ing. José Manuel Ruiz Juárez
Ing. Dennis Stanley Barrios Gonzalez
Ing. Edwin Estuardo Zapeta Gómez
Ing. Fernando José Paz González

Tutores de curso:
Dayana Alejandra Reyes Rodríguez
Andrea María Cabrera Rosito
Paula Gabriela García Reinoso
Piter Angel Esaú Valiente de León
Mario César Morán Porras
Denilson Florentín de León Aguilar

PROYECTO 1

OBJETIVO GENERAL

Desarrollar una solución integral que implemente tipos de datos abstractos (TDA) y visualización de datos (Graphviz) bajo el concepto de programación orientada a objetos (POO).

OBJETIVOS ESPECÍFICOS

- Implementar POO para el desarrollo de la solución a través de lenguaje Python.
- Utilizar estructuras de programación secuenciales, cíclicas y condicionales
- Visualizar TDA's por medio de la herramienta Graphviz.
- Utilizar archivos XML como insumos para la lógica y comportamiento de la solución.

ENUNCIADO

Este problema consiste en alojar objetos de bases de datos en sitios distribuidos, de manera que el costo total de la transmisión de datos para el procesamiento de todas las aplicaciones sea minimizado. Un objeto de base de datos es una entidad de una base de datos, esta entidad puede ser un atributo, un set de tuplas, una relación o un archivo. Los objetos de base de datos son unidades independientes que deben ser alojadas en los sitios de una red. Una definición formal del problema se presenta en la figura No. 1.

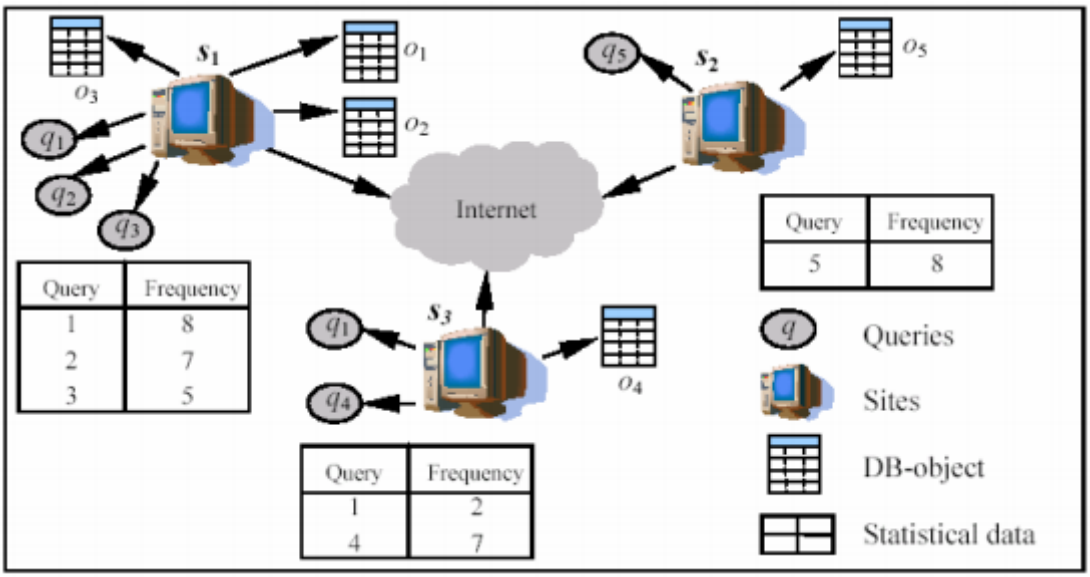


Figura No. 1 – Problema de diseño de distribución en una base de datos

La figura No. 1 muestra un set de objetos de bases de datos $O = \{o_1, o_2, \ldots, o_{no}\}$, una red de computadoras que consiste en un set de sitios $S = \{s_1, s_2, \ldots, s_{ns}\}$, donde un set de consultas $Q = \{q_1, q_2, \ldots q_{nq}\}$ son ejecutadas, los objetos de base de datos requeridos por cada consulta, un esquema inicial de alojamiento de objetos de bases de datos, y las frecuencias de acceso de cada consulta desde cada sitio en un período de tiempo. El problema consiste en obtener un nuevo esquema replicado de alojamiento que se adapte a un nuevo patrón de uso de la base de datos y minimice los costos de transmisión.

El problema de diseño de distribución consiste en determinar el alojamiento de datos de forma que los costos de acceso y comunicación son minimizados. Como muchos otros problemas reales, es un problema combinatorio NP-Hard. Algunas de las situaciones comunes que hemos observado cuando se resuelven instancias muy grandes de un problema NP-Hard son: Fuerte requerimiento de tiempo y fuerte demanda de recursos de memoria. Un método propuesto para resolver este tipo de problemas consiste en aplicar una metodología de agrupamiento.

Para “nt” tuplas y “ns” sitios, el método consiste en tener la matriz de frecuencia de acceso en los sitios F[nt][ns] de la instancia objetivo, transformarla en una matriz de patrones de acceso y agrupar las tuplas con el mismo patrón.

El patrón de acceso para una tupla es el vector binario indicando desde cuál sitio la tupla es accedida.

Por ejemplo:

Para la siguiente matriz de frecuencia de acceso:

$$\begin{bmatrix} 2 & 3 & 0 & 4 \\ 0 & 0 & 6 & 3 \\ 3 & 4 & 0 & 2 \\ 1 & 0 & 1 & 5 \\ 0 & 0 & 3 & 1 \end{bmatrix}$$

Entonces, su correspondiente matriz de patrones de acceso es

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Se puede observar que las filas 1 y 3 tienen los mismos patrones de acceso, así como también las filas 2 y 5. Entonces, habrá tres grupos considerando el tercer grupo formado simplemente por la fila 4. La cardinalidad de un grupo es definida por el número de tuplas incluidas en él.

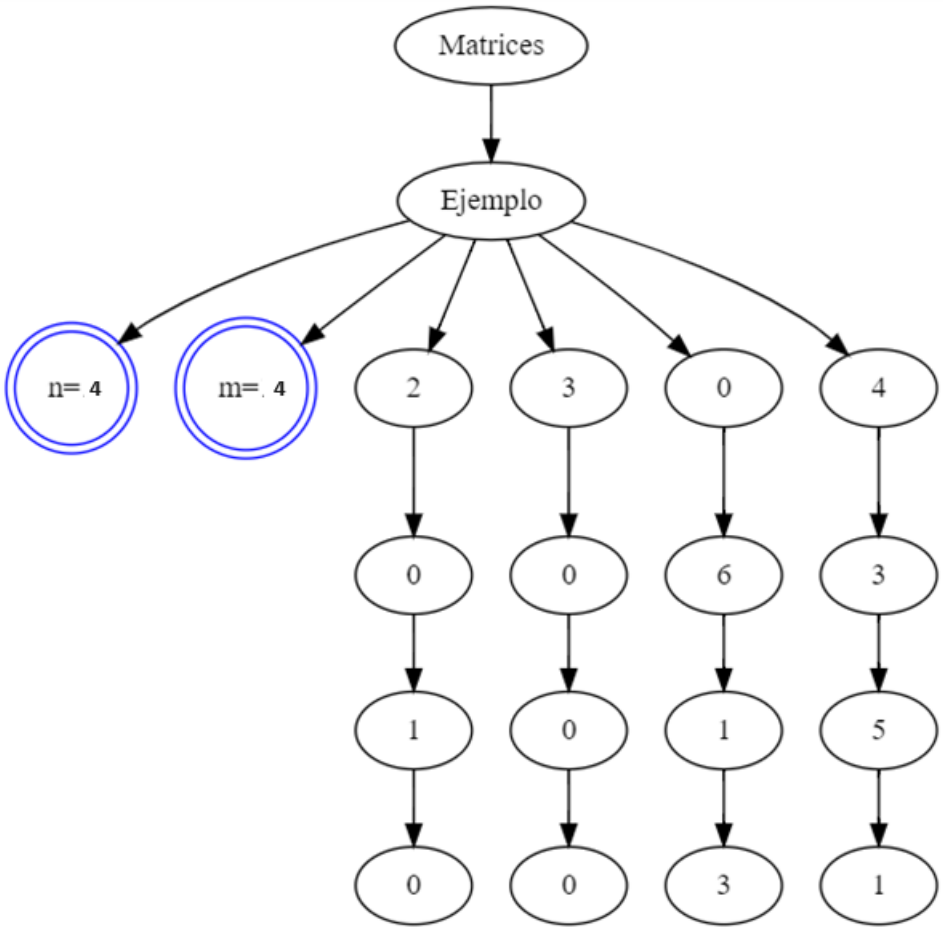
La matriz reducida de frecuencia de accesos obtenida de la suma de las tuplas en los grupos será:

$$\begin{bmatrix} 5 & 7 & 0 & 6 \\ 0 & 0 & 9 & 4 \\ 1 & 0 & 1 & 5 \end{bmatrix}$$

Se debe diseñar un programa que acepte “n” matrices de frecuencia de accesos y por cada una obtener los grupos formados por las tuplas con el mismo patrón de acceso. Finalmente, se debe obtener la matriz de frecuencia de acceso reducida.

REPORTES

Se deberá utilizar la herramienta Graphviz para crear un grafo que muestre de manera gráfica la estructura del archivo de entrada procesado. Se debe mostrar de una manera similar a la siguiente. En este ejemplo, “Ejemplo” será el nombre de la matriz, “n” y “m” serán las dimensiones y en la parte derecha se mostrarán los valores que contiene la matriz.



Es una imagen representativa, deberán crear su propio gráfico lo más detallado posible.

Archivos de Entrada y Salida

Los archivos de entrada y salida consistirán en archivos con extensión y estructura xml en el cual se limitará a utilizar únicamente las etiquetas:

- **matrices**: este será necesario para la lectura inicial del archivo, ya que será la etiqueta padre de todo.

- **matriz**: esta etiqueta será la que indica que una nueva matriz de frecuencia de accesos será creada para su respectivo análisis y únicamente puede estar dentro de la etiqueta **matrices** y puede tener los atributos:
  - **nombre**: este contendrá el identificador de la matriz leída (se deberá validar la existencia de matrices con el mismo nombre, para mantener la consistencia de los datos).
  - **n**: representa el número de filas que tendrá la matriz, si los datos dentro de ella son mayores a este atributo o menor a 1 será error.
  - **m**: representa el número de columnas que tendrá la matriz, si los datos dentro de ella son mayores a este atributo o menor a 1 será error.
- **dato**: esta etiqueta únicamente podrá estar dentro de la etiqueta **matriz** y contendrán los valores respectivos a cada celda de la matriz, esta etiqueta puede tener los siguientes atributos.
  - **x**: será la fila de la matriz y no puede ser mayor al atributo **n** de la matriz ni menor a 1
  - **y**: será la columna de la matriz y no puede ser mayor al atributo **m** de la matriz ni menor a 1
- **frecuencia**: esta etiqueta representa la frecuencia de los grupos de registros utilizados para crear la matriz reducida.

Ejemplo de Entrada:

```
<matrices>
  <matriz nombre="Ejemplo" n=4 m=4>
    <dato x=1 y=1>2</dato>
    <dato x=1 y=2>3</dato>
    <dato x=1 y=3>0</dato>
    <dato x=1 y=4>4</dato>
    <dato x=2 y=1>0</dato>
    <dato x=2 y=2>0</dato>
    <dato x=2 y=3>6</dato>
    <dato x=2 y=4>3</dato>
    <dato x=3 y=1>3</dato>
    <dato x=3 y=2>4</dato>
    <dato x=3 y=3>0</dato>
    <dato x=3 y=4>2</dato>
    <dato x=4 y=1>1</dato>
    <dato x=4 y=2>0</dato>
    <dato x=4 y=3>1</dato>
    <dato x=4 y=4>5</dato>
    <dato x=5 y=1>0</dato>
    <dato x=5 y=2>0</dato>
    <dato x=5 y=3>3</dato>
    <dato x=5 y=4>1</dato>
  </matriz>
```

```
   …
</matrices>
```

Ejemplo de Salida:

Donde n=fila, y=columnas y g=grupos de la matriz reducida

```
<matriz nombre="Ejemplo_Salida" n=3 m=4 g=3>
   <dato x=1 y=1>5</dato>
   <dato x=1 y=2>7</dato>
   <dato x=1 y=3>0</dato>
   <dato x=1 y=4>6</dato>
   <dato x=2 y=1>0</dato>
   <dato x=2 y=2>0</dato>
   <dato x=2 y=3>9</dato>
   <dato x=2 y=4>4</dato>
   <dato x=3 y=1>1</dato>
   <dato x=3 y=2>0</dato>
   <dato x=3 y=3>1</dato>
   <dato x=3 y=4>5</dato>
   <frecuencia g=1>2</frecuencia>
   <frecuencia g=2>2</frecuencia>
   <frecuencia g=4>1</frecuencia>
</matriz>
```

La aplicación deberá contar con un menú en consola, con las siguientes opciones

```
Menú principal:
        1. Cargar archivo
        2. Procesar archivo
        3. Escribir archivo salida
        4. Mostrar datos del estudiante
        5. Generar gráfica
        6. Salida
```

1. **Cargar Archivo:** Esta Opción solicitará la ruta del archivo a cargar.

```
Opcion Cargar Archivo:
Ingrese la ruta del archivo:
```

2. **Procesar el Archivo:** Esta opción será la encargada de procesar la información cargada en memoria, durante el proceso se deben ir mostrando mensajes al usuario para tener el conocimiento de lo que está pasando en el sistema.

```
> Calculando la matriz binaria...
> Realizando suma de tuplas...
```

3. **Escribir Archivo de salida:** Esta opción será la encargada de escribir el archivo con la salida específica.

```
Escribir una ruta especifica: C:/escritorio/ipc2/final.xml
Se escribio el archivo satisfactorio
```

4. **Mostrar datos del estudiante:** Mostrar los datos del estudiante, carné, nombre, curso, carrera, semestre y enlace de acceso a documentación.

```
> Juan Pablo Osuna de Leon
> 201503911
> Introduccion a la Programación y computación 2 seccion "A"
> Ingenieria en Ciencias y Sistemas
> 4to Semestre
```

5. **Generar gráfica**: El programa deberá permitir que el usuario elija una de las matrices ingresadas en el archivo de entrada y mostrará la gráfica cómo la que se indica en la sección de reportes incluyendo una 2ª. gráfica con la matriz reducida.

CONSIDERACIONES

Se deberá implementar una lista circular simplemente enlazada, creada por el estudiante, creando una clase Nodo, y una clase lista, de tal manera que al recorrer la lista se pueda validar la existencia de los nombres de las matrices, y en base en esta lista poder realizar un reporte gráfico de la misma.

Debe utilizarse versionamiento para el desarrollo del proyecto. Se utilizará la plataforma **Github** en la cual se debe crear un repositorio en el que se gestionará el proyecto. Se deben realizar 4 releases o versiones del proyecto (se recomienda realizar una por semana del tiempo disponible). **Se deberá agregar a su respectivo auxiliar como colaborador del repositorio**. El último release será el release final y se deberá de realizar antes de entregar el proyecto en la fecha estipulada.

DOCUMENTACIÓN

Para que el proyecto sea calificado, el estudiante deberá entregar la documentación utilizando el formato de ensayo definido para el curso. En el caso del proyecto, el ensayo debe tener entre 4 y 7 páginas de contenido, este máximo no incluye los apéndices o anexos donde se pueden mostrar modelos y diseños utilizados para construir la solución. Es obligatorio incluir

el diagrama de clases que modela la solución de software presentada por el estudiante y los diagramas de actividades con los principales algoritmos implementados en la solución.

RESTRICCIONES

- Solo se permitirá la utilización de los IDEs discutidos en el laboratorio.
- Uso obligatorio de programación orientada a objetos (POO) desarrollada por completo por el estudiante. De no cumplir con la restricción, no se tendrá derecho a calificación.
- El nombre del repositorio debe de ser **IPC2_Proyecto1_#Carnet.**
- El estudiante debe entregar la documentación solicitada para poder optar a la calificación.
- Los archivos de entrada no podrán modificarse.
- Los archivos de salida deben llevar la estructura mostrada en el enunciado obligatoriamente.
- Deben existir 4 releases uno por cada semana, de esta manera se corrobora el avance continuo del proyecto. **Se definirá una penalización por cada release faltante**.
- Se calificará de los cambios realizados en el cuarto release. Los cambios realizados después de ese release no se tomarán en cuenta.
- Cualquier caso de copia parcial o total tendrá una nota de 0 y será reportada a Escuela de Ciencias y Sistemas.
- Para dudas concernientes al proyecto se utilizarán los foros en UEDI de manera que todos los estudiantes puedan ver las preguntas y las posteriores respuestas.
- **NO HABRÁ PRÓRROGA.**

ENTREGA

- La entrega será el **4 de septiembre** a las 11:59 pm como máximo.
- La entrega será por medio de la UEDI.
- La documentación debe estar subida en el repositorio en una carpeta separada.
- Para entregar el proyecto en UEDI se deberá subir un archivo de texto con el link del repositorio.