**Informe Desarrollo Parcial I**

Manuel Esteban Orjuela Montealegre

Universidad De Antioquia

Facultad de Ingeniería

Departamento de Electrónica y Telecomunicaciones

Informática II

Semestre 2024 – 1

**Análisis del Problema y Consideraciones para la Alternativa de la Solución Propuesta**

**Primer Análisis.** Luego de una lectura detallada al documento proporcionado, encontré que se definieron ciertas especificaciones a tener en cuenta en mi solución. Entonces separé los problemas en 5 ramas señaladas.

***Creación de la Estructura de Datos.*** Podemos evidenciar el uso de estructuras bidimensionales dinámicas de tamaño NxN donde N es impar y el tipo de dato correspondiente a éstas es entero. Estas estructuras, poseen 4 estados, Neutro, Estado 1, Estado 2 y Estado 3.

***Neutro:*** Estado inicial, donde la matriz se rellene desde el numero 1 hasta (N\*N) – 1, de izquierda a derecha con la particularidad de que la posición [(N/2)][(N-2) ] es igual a ‘vacío’ – Su representación será el número 0.

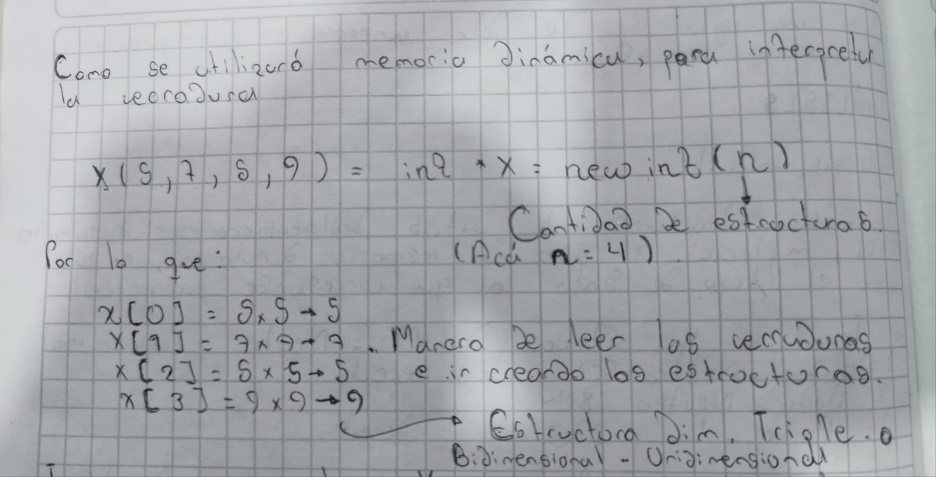
***Estado 1:*** Matriz neutro rotada 90 grados en sentido de las manecillas del reloj.

***Estado 2:*** Matriz neutro rotada 180 grados en sentido de las manecillas del reloj.

***Estado 3:*** Matriz neutro rotada 270 grados en sentido de las manecillas del reloj.

***Solución Propuesta.*** Para esta primera parte, lo más óptimo es utilizar un arreglo de arreglos, primero inicializar una matriz entero dinámica que apunte a N punteros y éstos reserven N espacios para así asignar los valores correspondientes. Para los estados habrá que jugar con las columnas y filas para intercambiar su posición, pero un apoyo para esto es utilizar los algoritmos implementados durante el ejercicio 15 la práctica 2. La implementación de los algoritmos la haré más adelante.

**Segundo Análisis.** Para este segundo apartado tuve en cuenta que se nos presentan estructuras en conjunto denominadas cerraduras. Esta agrupación se puede contemplar de la siguiente manera:

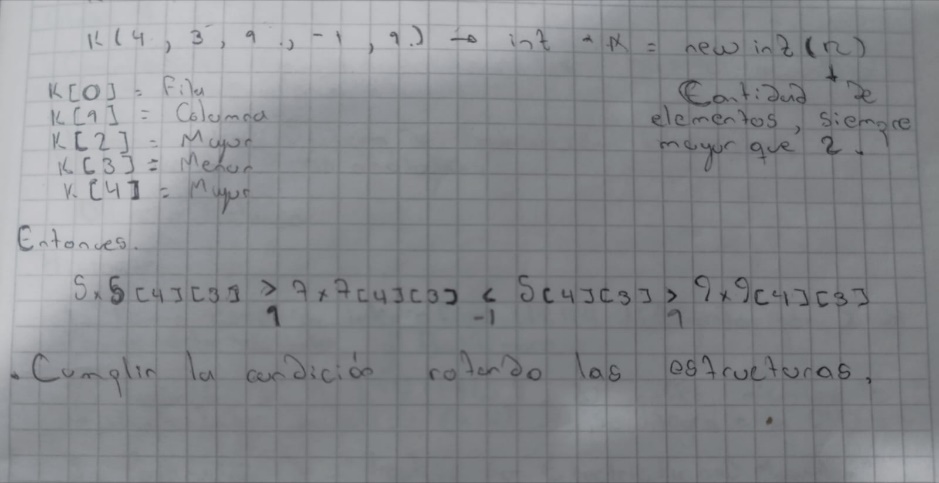


Al ser un conjunto con cantidad variable, también habrá que hacer uso de memoria dinámica.

***Primera opción.*** Podemos manejar estas cerraduras como un arreglo de una dimensión en el cual se irán leyendo sus posiciones para así crear los arreglos correspondientes dados su tamaño.

***Segunda opción.*** Hacer uso de un puntero triple, donde se almacenen directamente las matrices bidimensionales con un uso directo.

Sumado a esto, debemos tener en cuenta las claves, que de cierta forma nos indica qué parámetros deben cumplir las cerraduras para que se puedan abrir.



La llave corresponde a un arreglo unidimensional con las características expuestas en la imagen. De igual forma, se pueden tener lo siguiente a partir del arreglo de la llave.

***Cantidad Elementos Cerradura.*** La cantidad de elementos en el arreglo – 1, indican cuántas matrices hay en una cerradura.

***Dimensión Mínima de las Matrices de la Cerradura***. La posición 0 y 1 de la llave muestran que para que se cumplan las condiciones en todo el conjunto, todas las matrices deben tener una dimensión mayor o iguales al número mayor entre k [0] y k [1].

**Tercer Análisis. *Consideraciones para el funcionamiento de una llave respecto a una cerradura.***

**.** Los elementos correspondientes a k deben cumplir las siguientes condiciones:

- Cantidad elementos de K = Cantidad Elementos de X - 1.

**-** No existen valores nulos dentro de K.

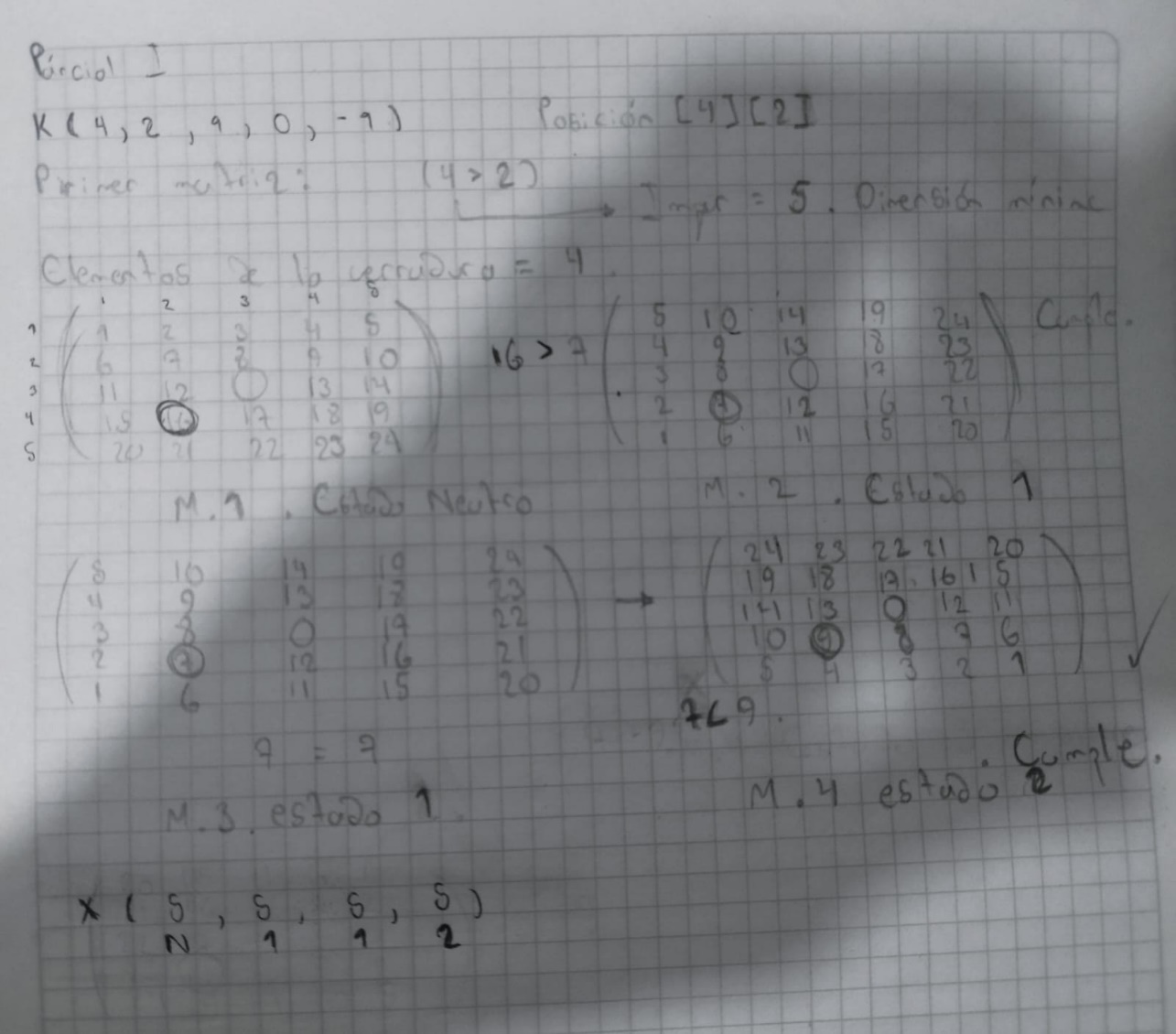
- Las posiciones que correspondientes a filas y columnas deben existir en cada una de las matrices de las cerraduras, de lo contrario, se evaluarían valores que no están presentes.

***Primera idea para la generación de una cerradura, dada una llave.***

**.** Podemos tomar en cuenta la relación anterior de las llaves para así determinar la cantidad de matrices en una cerradura. Cantidad Elementos X = Cantidad Elementos de k + 1.

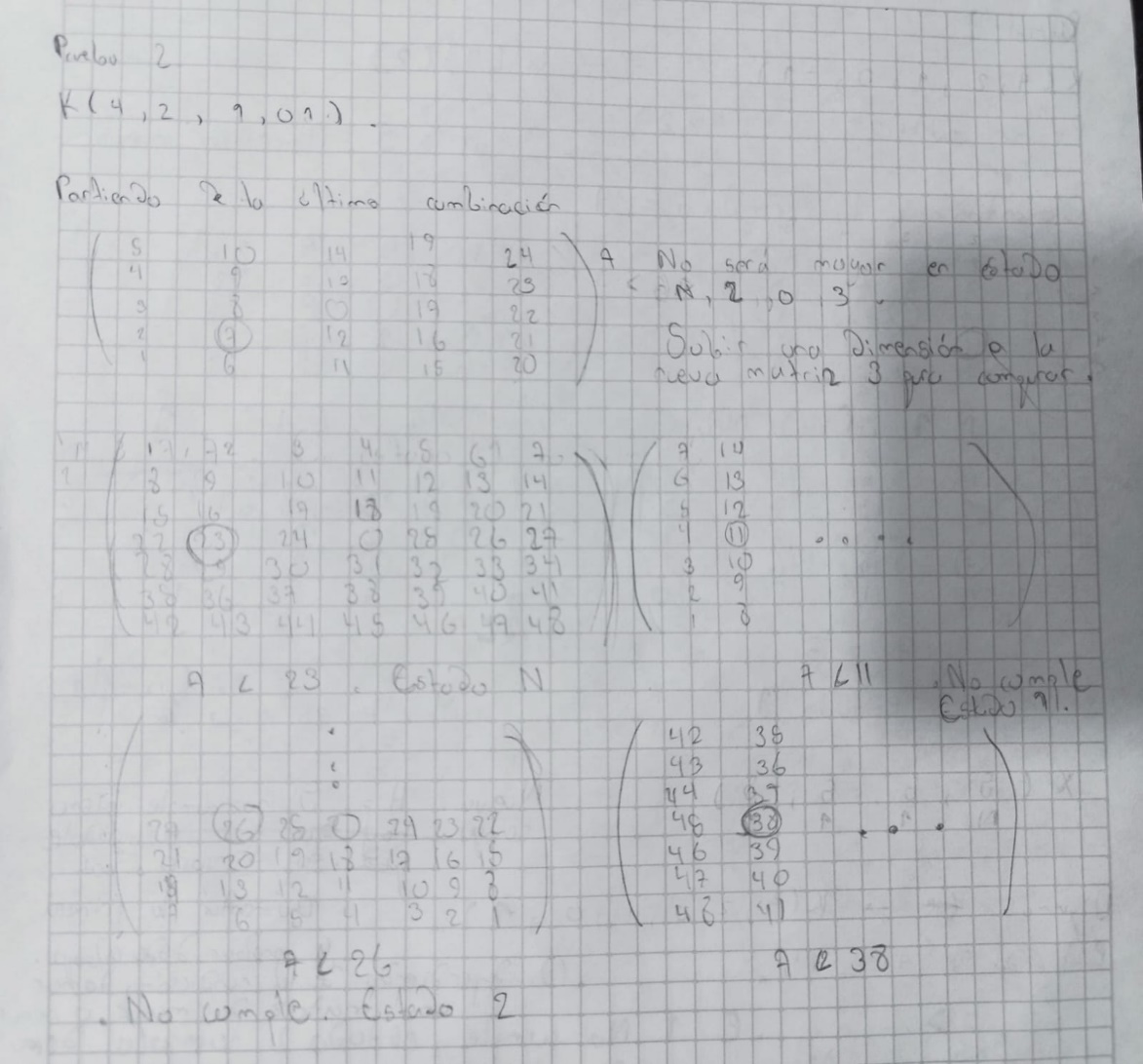
**.** La primera matriz se formará evaluando el número mayor entre k [0] y k [1]. Si resulta ser par, se seleccionará la opción impar más cercana donde se cumpla que: Numero impar = Incluye el número mayor entre k [0] y k [1].

**.** Para la segunda matriz, partimos del mismo principio del punto anterior. En caso de que aún realizando las rotaciones y evaluando el valor de la fila y columna de la llave no se cumpla la condición, aumentamos la matriz una dimensión impar más.

 ***Prueba de mi hipótesis (Caso ideal).***

Tras realizar la prueba surgen las incógnitas:

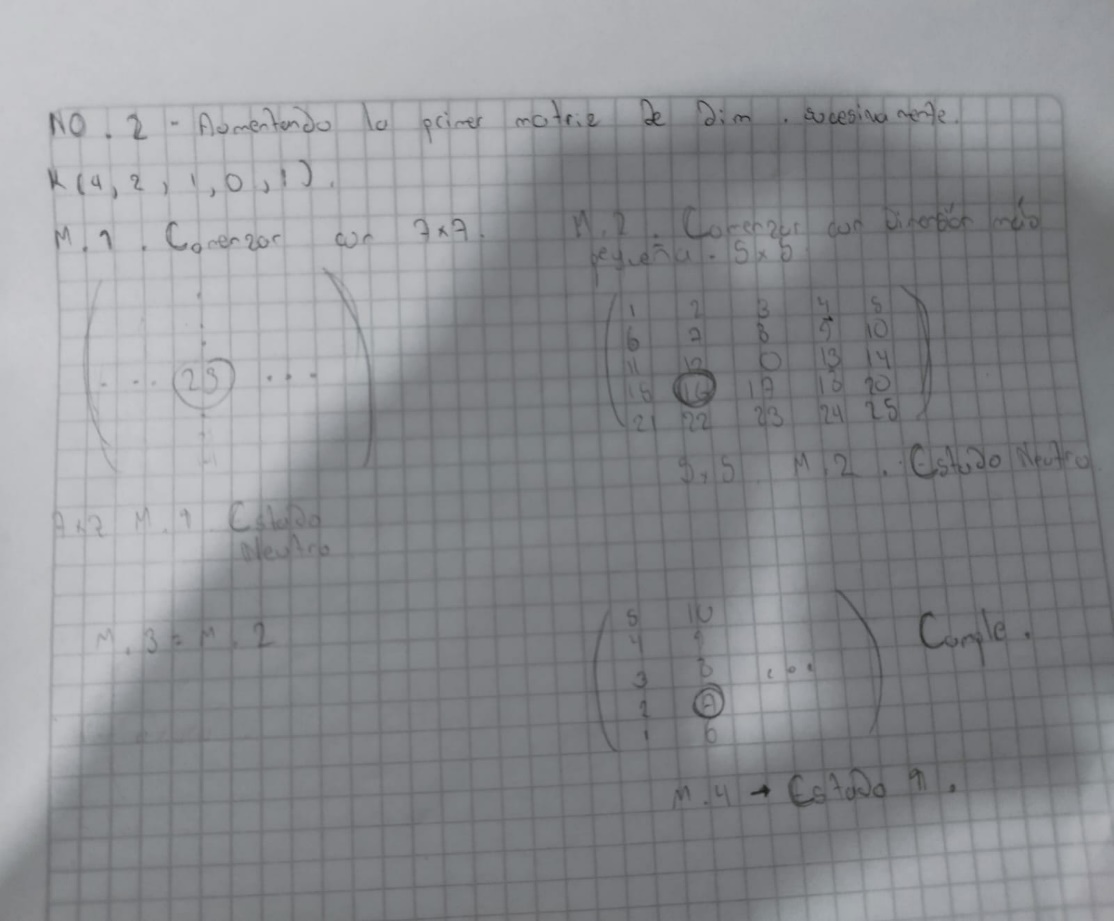
**.** ¿Qué sucede si no se cumple la regla de la llave aún si se aumenta la dimensión?, ¿Habrá que aumentar las dimensiones iniciales y probar nuevamente?, ¿Existe un método en el que pueda saber que sin importar si aumento la dimensión, la regla no se cumpla?

***Segunda prueba de mi hipótesis.***

Para comprobar mi hipótesis, forcé al ejercicio a que no se deba cumplir la condición, pero al parecer solo funciona cuando se necesita un número menor, para números mayores, no.

Entonces planteé lo siguiente:

***En vez de aumentar la dimensión de la última matriz, empezar aumentando la primera matriz, a partir de la segunda empezar desde la mínima y así sucesivamente.***



En este caso funciona la llave aplicando el principio propuesto. De primer vistazo (Aún sin comprobar) podría concluir que, si no se cumple la regla de la llave aplicando todas las rotaciones, existen estas dos soluciones:

**.**  Si se busca que el valor k [0] y k [1] sea MAYOR, la dimensión de la matriz siguiente debe ser mayor.

**.**  Si se busca que el valor k [0] y k [1] sea MENOR, la dimensión de la matriz siguiente debe ser menor.

**Tareas Definidas para el Desarrollo de Algoritmos**

Para la implementación de mi solución separé el código en los siguientes bloques con sus respectivas tareas definidas separas por funciones.

**Creación de estructuras de datos variables:**

. Crear matriz bidimensional de manera mecánica

. Crear los elementos internos de la matriz bidimensional

. Eliminar matriz bidimensional

. Rellenar matriz con elementos del estado neutro

. Mostrar matriz bidimensional (Apoyo visual)

**Rotación de estructuras**

. Matriz estado 1

. Matriz estado 2

. Matriz estado 3

**Creación de Cerraduras**

. Crear Cerradura

. Eliminar Cerradura

. Mostrar Cerradura (Apoyo visual)

**Validaciones**

. Tamaño impar para las matrices

**Algoritmos Implementados**

**Creación Mecánica de Las Estructura**

**Evolución de la Solución y Consideraciones**

**29/03/2024 – Primer día en la ejecución de la implementación propuesta**

Los apartados correspondientes a la creación de las estructuras de datos y sus rotaciones no tuvieron mayor cambio a lo propuesto en el análisis. Cuando enfrenté la sección de crear la cerradura tuve que tomar una decisión, entre un usar un arreglo bidimensional y unidimensional o un arreglo tridimensional, me resultó más práctico la segunda opción así que definí a la cerradura como un elemento de ese tipo. En el momento de liberar la memoria, caí en cuenta que, las matrices al tener tamaño variable, no iban a tener toda la misma dimensión, entonces, decidí crear un arreglo que almacene las dimensiones y que cada vez que se cree una matriz y se almacene en la cerradura, la dimensión ingresada se guarde en el arreglo de las dimensiones, luego, con un puntero, me podría apoyar para la liberación de la memoria.

.