Universidad de Antioquia.

Informe del parcial 1.

Giraldo Úsuga Mauricio.

Parra Osorio Rafael Ignacio.

Informática II.

Aníbal José Guerra Soler.

César Augusto Salazar Romaña.

Medellín.

2024.

Índice:

Introducción:

En el presente informe se pretende detallar el proceso en el debido desarrollo del proyecto del parcial número uno del curso Informática dos, así como los problemas e inquietudes surgidos a partir del análisis del desafío, con sus propuestas y soluciones, además de la descripción de los algoritmos junto a su respectiva implementación.

Análisis del problema y consideraciones:

La regla K determinará la de qué dimensiones serán las matrices de la cerradura X, así que se necesitará definir una regla K pidiendo al usuario que ingrese los primeros dos valores mediante un ciclo '*for*' para determinar las coordenadas del elemento a comparar y mediante otro ciclo '*for*' los valores de las comparaciones ya que estas sólo tendrán valores enteros entre -1 y 1. Los valores de dicha regla irán en un arreglo, para esto se diseñaría una función con puntero dinámico con el fin de guardar allí el arreglo ingresado que ha de usarse después. Luego, implementar un sistema de matriz dinámica hecha de punteros, con el fin de verificar la regla K utilizando dichas posiciones de memoria. El número de comparaciones determinará el número de matrices que tendrá la cerradura X, por ejemplo, si la regla K tiene 3 comparaciones, la cerradura X deberá tener 4 matrices, ya que se hará la comparación de la primera con la segunda, la segunda con la tercera y la tercera con la cuarta, es decir, n comparaciones para n + 1 matrices, considerando esto, laso elementos que tendrá la cerradura X serán n - 1 comparados con la regla K.

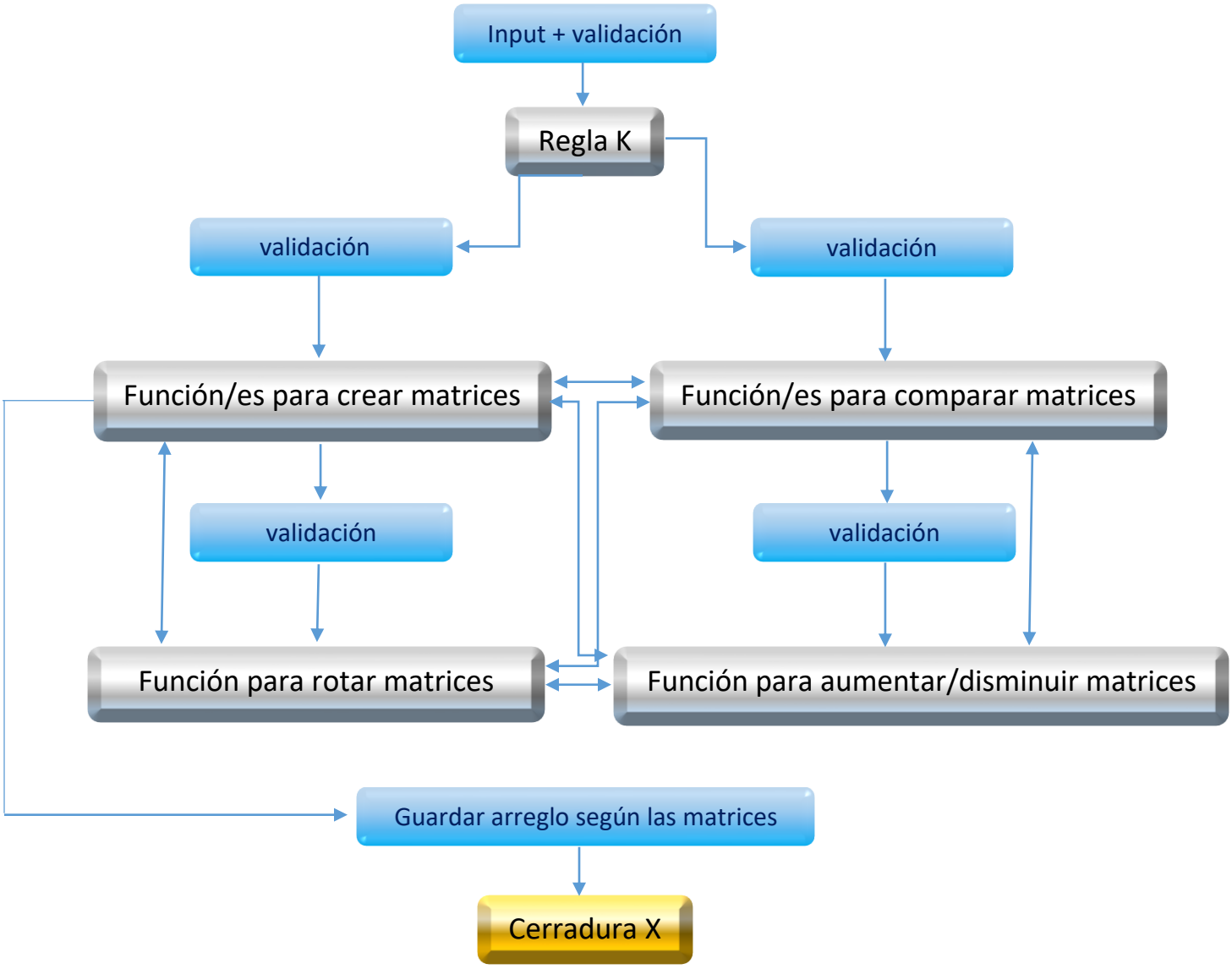
El método propuesto para llevar a cabo la creación de la cerradura X será el siguiente:

1) Una vez ingresada la regla K, se tomarán los dos primeros valores que representan la coordenada de comparación para así poner como matriz primitiva una matriz que cumpla con dicha coordenada, en este caso se implementaría la menor matriz posible, por ejemplo si la coordenada de comparación es la coordenada (5, 1…), entonces la primera matriz será una matriz de 5x5 en la posición base.
2) Esta matriz primitiva se replicará hasta completar el total de matrices requeridas por la regla, por ejemplo, si la regla es K (5, 1, 1, -1, 0), los valores de la cerradura X serían al principio (5, 5, 5, 5).
3) Las matrices sucesoras girarán en sentido anti-horario hasta encontrarse una combinación que satisfaga la comparación requerida por la regla K, si no es posible satisfacer la comparación girando, se procederá a aumentar o disminuir 1 vez en número impar la dimensión de la matriz sucesora.

Ahora, se deberá encontrar la manera de diseñar funciones para rotar y guardar las matrices que vallan surgiendo a partir de la regla K para llevar a cabo las ideas anteriormente puestas por el método.

En este punto se tendrán que diseñar acciones de validación de datos con el fin de evitar un aborto de ejecución por inputs con tipo de variable inválido, así como funciones de comparación de datos para poder desarrollar el programa y robustecerlo en pro de cumplir con los requerimientos del desafío.

Esquema de las tareas en el desarrollo de algoritmos:



Algoritmos Implementados:

A continuación se mostrarán los algoritmos con sus respectivos tipos y parámetros:

```
extern vector<int> dimensiones;
extern vector<int> contadorDeGiros;

void inicializarContadorDeGiros(int dimension);
void recorrerVectorG(const vector<int>& vec);
void recorrerVectorD(const vector<int>& vec);
int** crearMatrizImpar(int n);
void rotateMatrix(int** mat, int n);
int contarFilas(int indice);
vector<int**> crearMatricesImpares(int n, int d);
void compararElemento(vector<int**>& matrices, int fila, int columna, int* relacion);
int validar_numero(int* a);
int validar_filas_columnas(int* a);
int* regla_k(int* n);
```

Problemas de desarrollo afrontados:

El primer problema ha sido encontrar una forma de hacer que las matrices que se vayan creando a partir de la regla K queden guardadas.

Otro problema ha sido pensar cómo aumentar la dimensión de una matriz para hacer una comparación.

Uno de los principales problemas ha sido la situación en donde, si el usuario implementa una regla K con demasiadas comparaciones mayores seguidas se llega a un punto en que se acaban las opciones, es decir, la matriz no puede encontrar un valor mayor al elemento de la misma posición de la matriz que la antecede por más que se gire, debido a que el

elemento de dicha posición es el mayor en esa coordenada comparada con sus giros y además las matrices de menor dimensión no poseen la coordenada requerida, es decir habría un vacío. Para explicar mejor esta situación se ilustrará con el siguiente ejemplo:

Supóngase que se tiene una regla K que sea (1, 5, 1, 1, 1), entonces se llegara hasta la matriz de 5x5:

1	2	3	4	5
6	7	8	9	10
11	12		13	14
15	16	17	18	19
20	21	22	23	24

Luego de tres rotaciones quedaría así:

20	15	11	6	1
21	16	12	7	2
22	17		8	3
23	18	13	9	4
24	19	14	10	5

El 1 es el menor valor de la matriz 5x5 de los elementos que se rotan hacia esa posición ({1, 5, 20, 24}), si se pone una matriz más grande como sucesora no puede cumplir porque no se puede encontrar un número menor a uno para que la antecesora sea mayor y una matriz más pequeña como una de 3x3 no posee elementos en la coordenada 1x5 porque su límite es 3 para las columnas y filas.

Así que la regla K propuesta al inicio se vería truncada y quedaría como:

K = (1, 5, 1) arrojando una combinación X = (5, 5).

Evolución en la solución y consideraciones para la implementación:

La solución para el problema de guardar las matrices ha sido implementar un vector contenedor.

Para aumentar la dimensión de una matriz se ha resuelto llamando una función que crea las matrices dentro de la función comparación cambiando el parámetro de la dimensión sumando un 2.

En cuanto al problema de las comparaciones mayores seguidas que hacen llegar el algoritmo a un punto muerto donde no se puede encontrar una solución, una de las posibles soluciones planteadas sería que la regla K quede truncada al ser imposible que la comparación coincida con lo pedido por el usuario, arrojando como resultado la cerradura X con la regla K truncada al ser lo más cercano y posible a lo que solicitó. La segunda solución y más viable a opinión del equipo, ha sido devolver un mensaje al usuario comunicándole que no es posible crear una cerradura X a partir de la regla K que ingresó.

Conclusión:

Los resultados obtenidos han sido satisfactorios y han cumplido con las expectativas de los resultados esperados. La regla K ha demostrado tener limitaciones de uso en sus comparaciones mayores, lo que imposibilita hallar siempre una solución que satisfaga la petición ingresada por el usuario, sin embargo, esta limitación no provocará ningún fallo en el programa e igualmente se le dará respuesta al usuario a través de un comunicado que le informe que su petición no es posible debido a la no coincidencia con la lógica de la regla K.

1	2	3	4	5
6	7	8	9	10
11	12		13	14
15	16	17	18	19
20	21	22	23	24

Matriz 5x5.

Punto (1, 1).

Matriz 3x3.

Punto (1, 1).

1	2	3
4		5
6	7	8

Los puntos no se alinean.

No se puede hallar valor para que 1 sea mayor.

1	2	3	4	5
6	1	2	3	10
11	4		5	14
15	6	7	8	19
20	21	22	23	24