Desafío 1 - Parcial 1

Informática II

Por:

Juan Felipe Higuita Perez

Semestre 2024-1

Universidad de Antioquia

2024

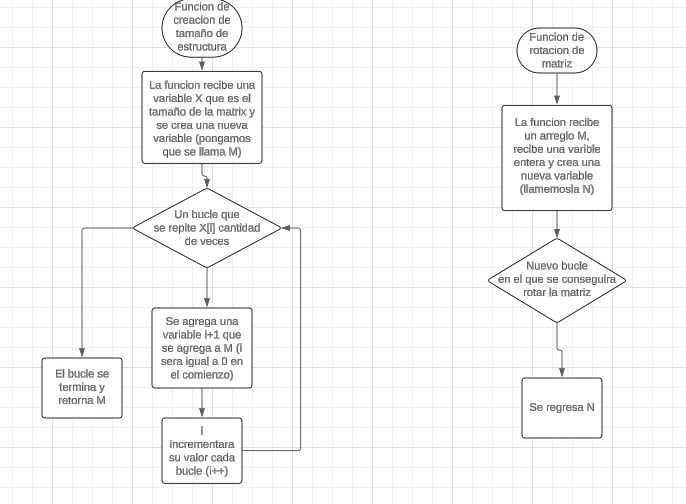
**Análisis del problema**

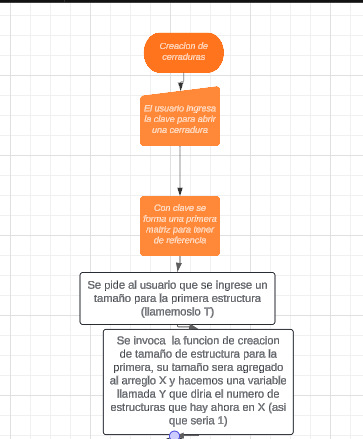
Se nos presenta una breve información sobre una estructura de datos que llamamos M que se trata de una matriz que tiene la característica de tener un número de filas columnas iguales e impares así siendo 3x3, 5x5, etc… También que su centro es neutro osea que no posee un número en esa celda, con las imágenes presentadas podemos observar que la estructura en la primera figura se puede ver que los números van de izquierda a derecha, tal que la primera una estructura 3x3 seria de la forma de la primera fila “1,2,3” la segunda fila “4, , 5” y la tercera “6, 7, 8”. Otra información que nos presentan es que tiene la capacidad de rotar y que su tamaño puede ser alterado siempre y cuando la regla de sea impar.

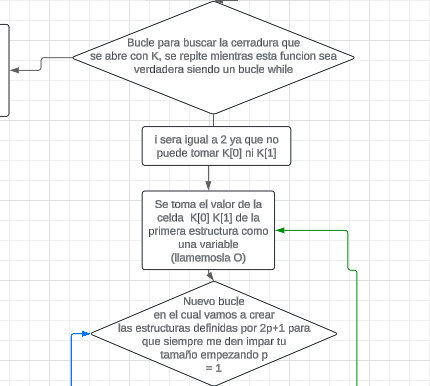
Una vez dándonos esta breve información nos comentan que una empresa de cerraduras usa esto con algunas especificaciones más como por ejemplo el tamaño de cada estructura de la cerradura sostenidas por un arreglo X y la forma con la que esta se abre es ingresada por el usuario siendo esta por medio una de clave K. Entonces lo que nos están pidiendo es hacer un código en el que se reciba una clave K ingresada por el usuario junto a X, así luego el programa deberá generar una estructura de matrices del tamaño que especifica el arreglo X y que sea validado con la clave K para que pueda abrirse, después se tendrá que generar un programa en que las matrices se puedan rotar de manera independiente para poder alinear las celdas y que la clave K sea verdadera.

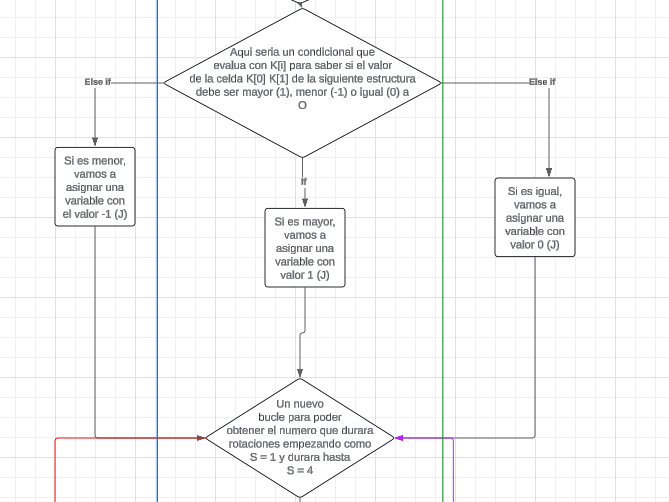
Entonces luego de pensarlo , sería hacer primero como dice la guia pedir que el usuario ingrese la clave K en una arregló para seguidamente hacer una función en la que se calcule la posición de la celda principal de la primera estructura con los primeros dos elementos del arreglo para que ubiquen del valor de la celda para tenerlo de referencia para asi con los otros tres elementos empiezan a formar las otras estructuras para tener la cerradura y su combinación hecha, para eso se deberá crear otras funciones para la creación de estas para que comparen sus valores con el de la primera estructura y también otra para comparar las rotaciones si contienen el mismo valor para que esa estructura pueda usarse con la clave K.

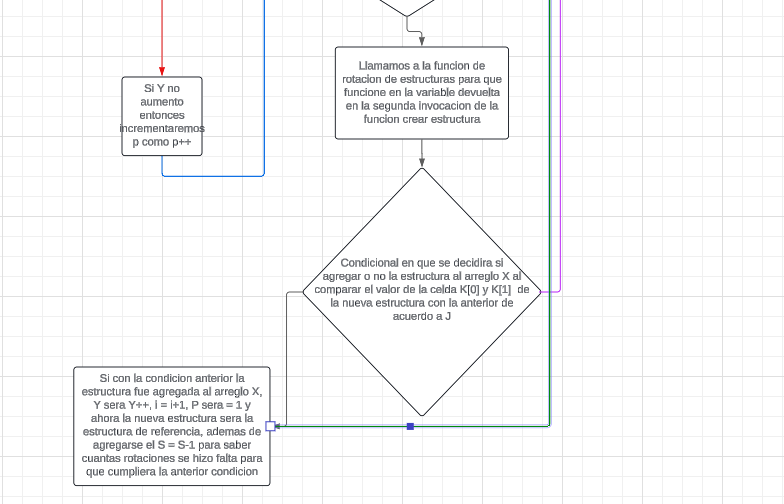
**Esquemas de diseño del algoritmo**

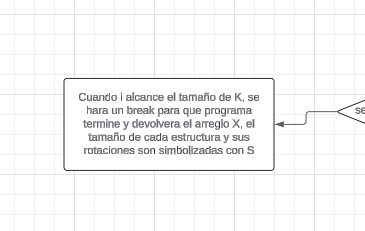












**Algoritmos implementados**

En el codigo separamos las funciones en dos modulos conocidos como “operaciones de matrices” donde se hara la creacion de matrices y la rotacion de estas, mientras que en la otra se llamara cerraduras donde se hara la creacion de la cerradura junto con la validación de las matrices si cumplen con las condiciones que pone la clave K.

-generarMatriz: esta función recibe un tamaño determinado para poder generar una matriz, además de generar su centro vacio.

-rotarMatrizContraManecillas: Función que rota la matriz que recibe, la razon por la que se llama asi es que las rotaciones que usa la guia es en contra de las manecillas del reloj.

-verificarCondicion y validarClave: Estas son funciones auxiliares que sirven para poder verificar que las matrices que entre cumplan las condiciones que tiene la clave K.

-generarCerraduras: Esta funcion invoca todas las demas funciones anteriormente mencionadas donde como su nombre indica genera la cerradura con las estructuras que generan generarMatriz y la funcion de rotacion que pasan de verificarCondicion y validarClave para agregar su tamaño a un arreglo X y su rotaciones para que se el dato de salida del programa y sea entregado al usuario.

**Problemas de desarrollo**

Como todo proyecto, es normal que uno se encuentra algunas dificultades en las que pueda verse atorado y que tenga que probar muchos métodos o pensarlo asi que a continuación explicare los que he tenido hasta el momento:

* Tuve problemas al desarrollar las matrices ya que analizando los ejemplos que nos han mostrado en la guia, note que el centro que debe estar vacío siempre estaba definido por el tamaño de la matriz divido entre 2 mas 1, ya que si tomamos como ejemplo una matriz 3x3 el espacio donde debería ir el 5 es el centro de esta y si dividimos 9 entre 2 nos dará 4.5 que en c++ se aproxima automáticamente a 4 y si sumamos 1 nos dará 5. Entonces el problema viene cuando al hacer eso es que no sabía cómo hacer que la siguiente celda estuviera el número anterior, pues todo el rato me daba un número que no era y por lo tanto todo después del centro se me desorganizaba, así que después de varias pruebas pude hacer que si me hiciera bien la matriz como debería ser.
* Otro problema que se presento en el desarollo del codigo es que cada vez que intentaba cambiarlo el programa llegaba a crashear haciendo que tuviera que rehacer varias veces una misma función buscando la forma que el programa no colapsara.
* Junto al anterior problema, se me presento dos tipos de errores uno llamado \*\*\* stack smashing detected \*\*\*: terminated que era debido a que el programa estaba escribiendo mas alla de los límites de alguna matriz y corrompía la memoria y free(): double free detected in tcache 2 que indica que hay un problema al intentar liberar la misma área de memoria mas de una vez.

Algo que tengo que aclarar es que el problema de los crasheos es el problema que mas se presento en el codigo, si no llego a hacer que el codigo funcione, es debido a los crasheos tan continuos que me generaba el programa, intente cambiar que solo estuviera un limite del tamaño de las matrices para que no sobrepasan la memoria pero aun me estaban dando a menudo estas situaciones.

**Evolución de la solución**

Primero se empezó con lo mas sencillo que era hacer una función para crear matrices como indicaba la guia, una matriz de tamaño variable que podía debía tener su centro vacío, para eso se tuvo que encontrar el patron del centro vacío el cual pude deducir que era la aproximación de la mitad del tamaño total de la matriz porque por ejemplo si es una matriz 3x3 el centro es donde debe estar 5, entonces si dividimos (3x3)/2 da 4.5 que en c++ se aproxima automáticamente a 5 asi que cuando llega a esta celda puse un condicional para que restara 1 a los siguiente números de la matriz para que quedara como las estructuras mostradas en la guia.

Luego de esto se empezó a desarrollar la funcion para rotar matrices, hay que tener en cuenta que la matriz gira en un sentido contra las manecillas del reloj asi que por eso se le puse ese nombre, esta funcion recibe una matriz para que cambie de lugar sus variables y simule una rotacion, siendo para que si hace falta rotarla si no coincide con la regla en su forma inicial.

Como siguiente paso cree la función generar cerraduras, que básicamente fue pasar lo que habia hecho en el main en un principio a esta, recibe la clave K donde hace las matrices y las rotas un numero determinado de veces poniendo un maximo de 3 veces antes de volver a su forma inicial devolviendo las matrices con sus rotaciones.

Despues de esto vino lo complicado, que era que solo introdujera en un arreglo X el tamaño de las matrices y de volviere este junto con el numero de rotaciones que hizo falta para que cumpliera la condicion de la clave K, asi que tuve que crear un ciclo donde si la matriz no cumplia la condicion aumentara en 2 su tamaño para que vuelva a probar hasta que encontrara la cantidad determinada de matrices que hacia indicaba la clave K con sus rotaciones y que comparara sus valores que la celda que indicaba esta, siendo introducida las funciones de validacion aunque en el proceso con mucha dificultad no se pudo lograr que imprimiera con exito por culpa de los constantes crasheos del programa y de sus errores.