**Informe sobre la Solución del Problema Planteado**

**Consideraciones Iniciales**

Se presenta una estructura de datos M con la capacidad de girar y producir resultados según las Figuras 1b - 1d. Además, M puede cambiar sus dimensiones, como se muestra en las Figuras 2a y 2b.

**Características del Sistema de Cerraduras X**

Varias M se alinean una tras otra, utilizando la celda del centro como referencia.

No hay restricción para la cantidad de M alineadas ni para sus tamaños.

El sistema de apertura de X funciona a partir de la validación de una regla K, que considera el valor de una celda específica, su posición y ubicación dentro de las diferentes estructuras alineadas.

**Objetivos**

Desarrollar un módulo para crear estructuras de datos M de tamaño variable.

Implementar funciones para realizar rotaciones a las estructuras M.

Desarrollar un módulo para configurar cerraduras X con diferentes tamaños y alineaciones.

Implementar funciones para validar una regla de apertura sobre una cerradura X.

Generar al menos una configuración de cerradura X a partir de una regla K.

**Requisitos Mínimos**

Generar un informe detallado del desarrollo del proyecto, incluyendo:

Análisis del problema y consideraciones para la solución.

Esquema de tareas definidas para el desarrollo de los algoritmos.

Implementar la solución en C++.

Incluir el uso de punteros, arreglos y memoria dinámica.

Crear un repositorio público con los archivos relacionados a la solución (informe, código fuente y otros).

Realizar commits regulares al repositorio.

Adjuntar un enlace al repositorio y otro al video de demostración.

Cumplir con las fechas de entrega establecidas.

Estructura del Informe

**Plan De Solución**

Regla k (4,3,1,-1,1)

análisis matrices

Primer paso

Mi programa empieza a crear una matriz desde 3x3 y va a ir sumando hasta un lugar donde exista la posición 4.3 que especifica la regla k teniendo en cuenta el sistema de matriz que hablamos sería este:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | 0 | 1 | 2 | 3 | 4 |
|  |  |  |  |  |  |  |
| 0 |  | 1 | 2 | 3 | 4 | 5 |
| 1 |  | 6 | 7 | 8 | 9 | 10 |
| 2 |  | 11 | 12 | 0 | 13 | 14 |
| 3 |  | 15 | 16 | 17 | 18 | 19 |
| 4 |  | 20 | 21 | 22 | 23 | 24 |

Donde el índice empieza en 0 por lo cual una de 3x3 no nos serviría

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | 0 | 1 | 2 |
|  |  |  |  |  |
| 0 |  | 1 | 2 | 3 |
| 1 |  | 4 | 0 | 5 |
| 2 |  | 6 | 7 | 8 |

Ya que solo alcanza hasta 2x2, e ahora nos toca crear una matriz que tiene que ser diferente a la anterior es decir ya no podemos utilizar una 5x5 que en su sombra guardemos su valor a que me refiero con su sombra una matriz que tenga sombra podría ser la 7x7 ya que como dijimos no puede ser igual a la interior donde su sombra va a ser así

Esta es la 5x5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 0 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 |
| 20 | 21 | 22 | 23 | 24 |

Esta va a ser la 7x7

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 0 | 25 | 26 | 27 |
| 28 | 29 | 30 | 31 | 32 | 33 | 34 |
| 35 | 36 | 37 | 38 | 39 | 40 | 41 |
| 42 | 43 | 44 | 45 | 46 | 47 | 48 |

La sombra se vería así

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 1 | 2 | 3 | 4 | 5 | 14 |
| 15 | 6 | 7 | 8 | 9 | 10 | 21 |
| 22 | 11 | 12 | 0 | 13 | 14 | 27 |
| 28 | 15 | 16 | 17 | 18 | 19 | 34 |
| 35 | 20 | 21 | 22 | 23 | 24 | 41 |
| 42 | 43 | 44 | 45 | 46 | 47 | 48 |

Por ende ahora guardamos la posición del número 39 que es donde está la sombra del 23 , la regla k (4,3,1,-1,1) nos dice que la posición del número 23 tiene que tener un número mayor al de la posición de la segunda matriz por ende utilizaremos la función que hicimos de rotar y verificar hasta encontrar que en esa posición haya un número menor que si la rotamos una vez quería el número 15 que efectivamente es menor ahora vemos que en la regla k hay un -1 lo cual nos dice que nuestro 15 tiene que ser un número menor al de la siguiente matriz recordando que no podemos usar ahora la 7x7 podemos probar nuevamente la 5x5 ya que quedaría nuevamente la posición con el número 23 que cumple con la regla por ultimo tenemos la regla del número 1 que nos dice que el número que tenemos debe ser mayor a la siguiente matriz como ya utilizamos la 5x5 vamos a utilizar o la 7z7 o la 9x9 para el ejemplo utilizamos 9x9

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 |
| 34 | 38 | 39 | 40 | 0 | 41 | 42 | 43 | 44 |
| 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 |
| 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 |
| 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 |
| 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 |

Como podemos ver tenemos que saber dónde está la sombra

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| 19 | 20 | 1 | 2 | 3 | 4 | 5 | 26 | 27 |
| 28 | 29 | 6 | 7 | 8 | 9 | 10 | 35 | 36 |
| 34 | 38 | 11 | 12 | 0 | 13 | 14 | 43 | 44 |
| 45 | 46 | 15 | 16 | 17 | 18 | 19 | 52 | 53 |
| 54 | 55 | 20 | 21 | 22 | 23 | 24 | 61 | 62 |
| 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 |
| 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 |

Que sería en el numero 59 pero 59 es mayor que 23 por ende aquí también rotaremos la matriz hasta que quede el número 22 que cumple al final mostraremos las matrices con sus respectivas rotaciones y el código que en este caso seria 5,7,5,9 que hace referencia a 5x5 7x7 5x5 9x9

Crear Matriz: Lo primero que necesitamos es una función para generar nuestra matriz M. Esta matriz tiene un tamaño variable, pero hay una regla importante: siempre debe tener un número impar de filas y columnas para que podamos tener un centro claro. Además, queremos que el centro esté lleno de ceros y que el resto de la matriz esté llena con números consecutivos, Esto nos ayudará a tener una estructura sólida para nuestras cerraduras.

Rotar Matriz: Una vez que tengamos nuestra matriz, necesitaremos una forma de girarla. Esto será útil más adelante cuando estemos jugando con las cerraduras X. Imagina que estamos girando las diferentes partes de la cerradura hasta que todas estén alineadas correctamente.

Configurar Cerradura: Ahora, para construir nuestras cerraduras X, necesitaremos alinear varias matrices M una tras otra,

Generar Configuración de Cerradura: Una vez que tengamos nuestras reglas K, necesitaremos una forma de generar configuraciones de cerraduras X basadas en esas reglas. Esto implica jugar un poco con las matrices y asegurarnos de que estén alineadas correctamente según las reglas que hemos establecido.

Imprimir Matriz: Por supuesto, necesitaremos una forma de ver nuestras matrices. Es como tomar una foto de nuestras cerraduras X y ver cómo quedaron. Esto nos ayudará a depurar y entender lo que está sucediendo en nuestro programa.

Alinear Estructuras: Y finalmente, necesitaremos una forma de alinear nuestras matrices correctamente. Esto será clave para validar nuestras reglas K y abrir nuestras cerraduras X. Imagina que estamos ajustando las diferentes partes de la cerradura hasta que encajen perfectamente.

**Funciones**

Funciones de manipulación de matrices M:

* Crear matriz: Función para crear una matriz M de tamaño variable. **( int\*\* crearMatriz(int dimension);**
* Liberar matriz: libera el espacio de memoria de la **matriz (void liberarMatriz(int\*\* matriz, int dimension);**
* Rotar matriz: Función para rotar una matriz M en sentido horario. ( void **rotarMatriz(int\*\* matriz, int dimension, int veces);**
* Imprimir matriz: Función para imprimir una matriz en la consola o en un archivo**. ( void imprimirMatriz(int\*\* matriz, int dimension);**

Funciones de configuración de cerraduras X:

* Configurar cerradura: Esta función será responsable de crear la cerradura X inicial con las dimensiones y alineaciones de matrices M especificadas, Almacena la dimensión de la estructura de la cerradura **X. int max\_val(int a, int b);**
* Configurar cerraduras: la función nos permitirá crear matrices que diferente a la anterior planteade y rotarla para cumplir con la solicitud de **k(int\*\* siguienteanillo(int predimension, int valor, int\* x, int\* y, int condicion,int\* tamano);**