**Proceso de análisis y diseño de solución Parcial #1.**

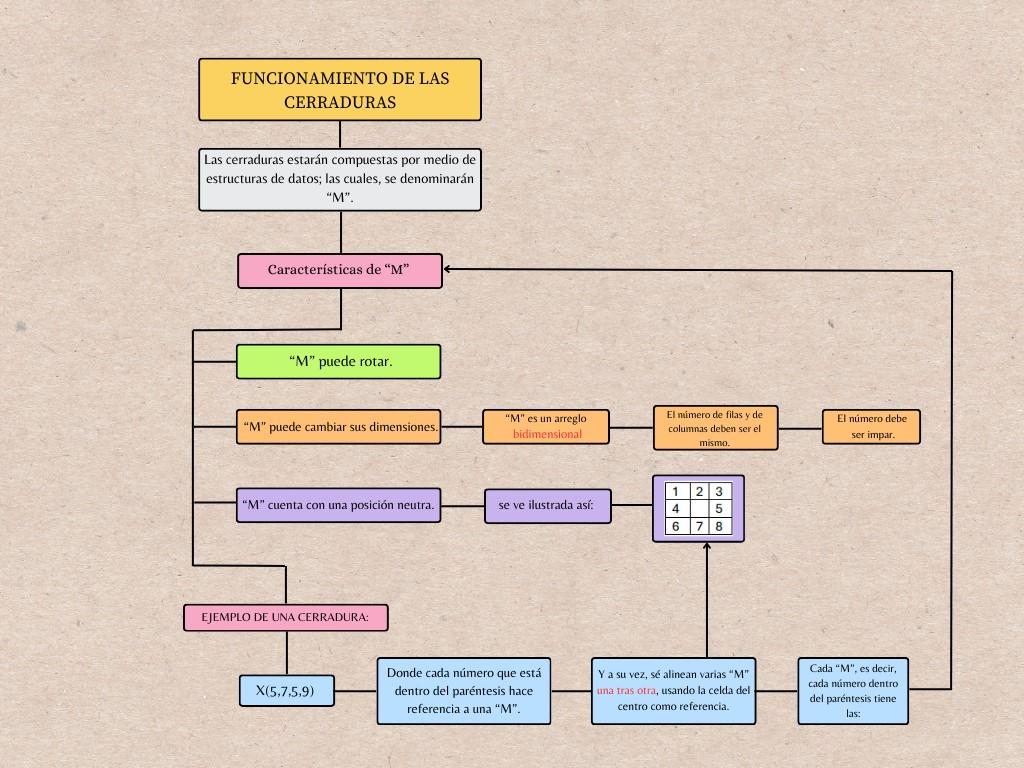
El problema planteado en este desafío consiste en la validación de una clave que será la “llave” para abrir cerraduras; tanto claves como cerraduras tienen ciertas características específicas que deben ser cumplidas y validadas.

Se nos ha pedido diseñar un programa en C++ que sirva como mecanismo de solución frente a las necesidades que la empresa “Informa2” posee. Además, sé nos ha proporcionado una serie de ítems que el programa debe de cumplir para ser completamente útil y funcional de acuerdo a las especificaciones de la empresa.

1. **Análisis del problema y consideraciones para la alternativa de solución propuesta.**

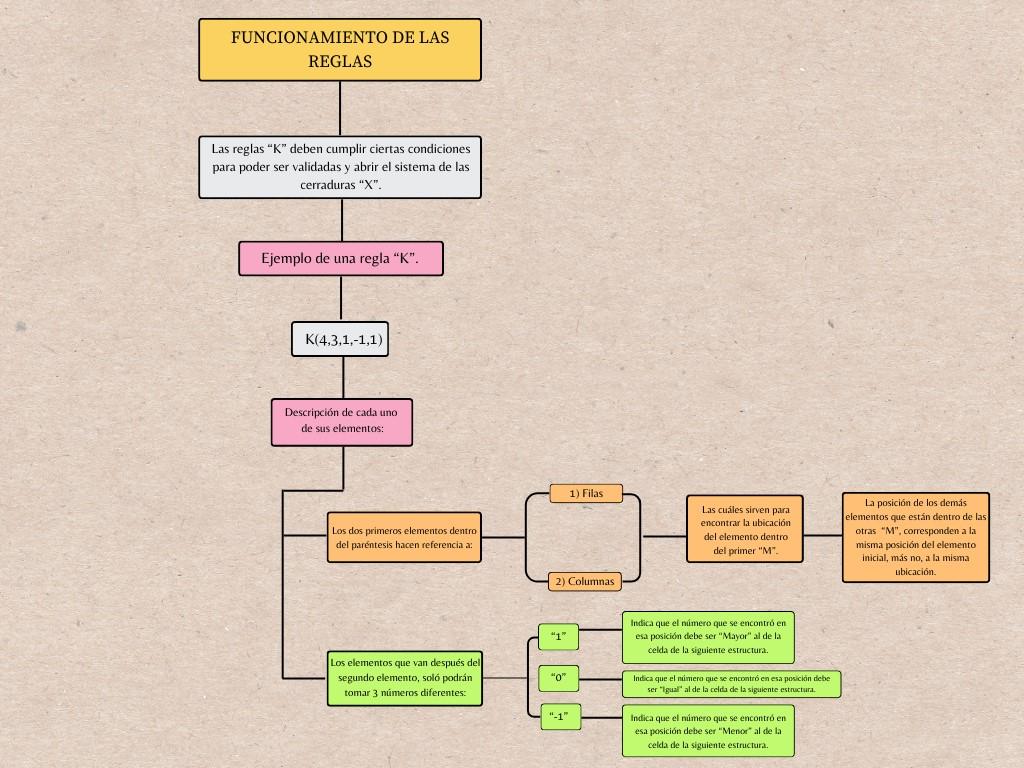
* **Análisis del problema:**

**Funcionamiento de las cerraduras fabricadas por la empresa “Informa2”:** A continuación, se planteará un esquema que contiene información referente a las características específicas de las cerraduras **“X”**; con el fin de tener una mejor legibilidad y organización sobre las características de las cerraduras.



**Funcionamiento de las reglas “K”:**

A continuación, se planteará un esquema que contiene información referente a las características específicas de las reglas **“K”**; con el fin de tener una mejor legibilidad y organización sobre las características de las claves.



* **Consideraciones para la alternativa de solución propuesta:**

Teniendo en cuenta los dos esquemas anteriores y las características descritas sobre cada elemento (cerradura y clave); hemos generado una serie de ideas sobre cómo resolver el problema planteado por la empresa “Informa2”.

Se pondrán las ideas a manera de ítems; en donde cada uno de ellos, corresponderá a la descripción de las estrategias que se utilizarán para resolver cada una de las tareas.

**Comportamiento de las cerraduras:**

1. **“M” puede cambiar sus dimensiones:** Frente a esta característica en particular, se planea hacer uso de memoria dinámica; pues es una herramienta facilitadora qué permitirá ajustar el tamaño del arreglo según sea necesario; además de esto, el tamaño del arreglo no es un dato conocido antes de la ejecución, el tamaño del arreglo se conocerá durante esta; ya que la clave que ingresa el usuario por pantalla, es lo que brindará “la pauta” para determinar la cantidad de “M” que tendrá la cerradura **“X”**.
2. **“M” puede rotar:** Para darle solución a esta característica, se planea desarrollar una función que reciba la dimensión “n” de cada arreglo “M” en la posición neutra, esta se transpone y luego invierte sus elementos en términos de columnas. Es decir, cada “i” (filas) de cada “M” va a tomar la posición de cada “j”(columnas) de cada “M”, después se recorren al revés las “i” de “M”, y de esta manera me queda mi arreglo “M” en una **posición 1** que es igual a 90°.
3. **Matriz en posición neutral**

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

1. **Matriz en posición neutral transpuesta**

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

1. **Matriz en posición 1 (90°)**

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

El numeral “B” y “C” se repiten dos veces más, permitiendo que se creen dos nuevas posiciones para el arreglo, las cuales corresponden a 180° y 270° respectivamente.

1. **“M” cuenta con una posición neutra:** Tal cómo se describió en el esquema anterior, cada “M” que haga parte de la cerradura “X” contará con una posición neutra, la cual servirá como referencia para que cada una de las “M” se alineen una detrás de la otra, usando la celda del centro como referencia.

Es por esto, que se planea hacer una función que calcule para cada uno de los “M” la posición del centro; esto se podría llevar a cabo dividiendo el número de filas y columnas entre 2 **(teniendo en cuenta la indexación basada en cero),** es decir, que los índices de las filas y columnas comienzan desde 0.

**Ejemplo:** Tengo una estructura "M" de 5 filas y 5 columnas, y si tomamos en cuenta lo anterior, la posición neutra debería ser la celda que está ubicada en la fila 2 y la columna 2.

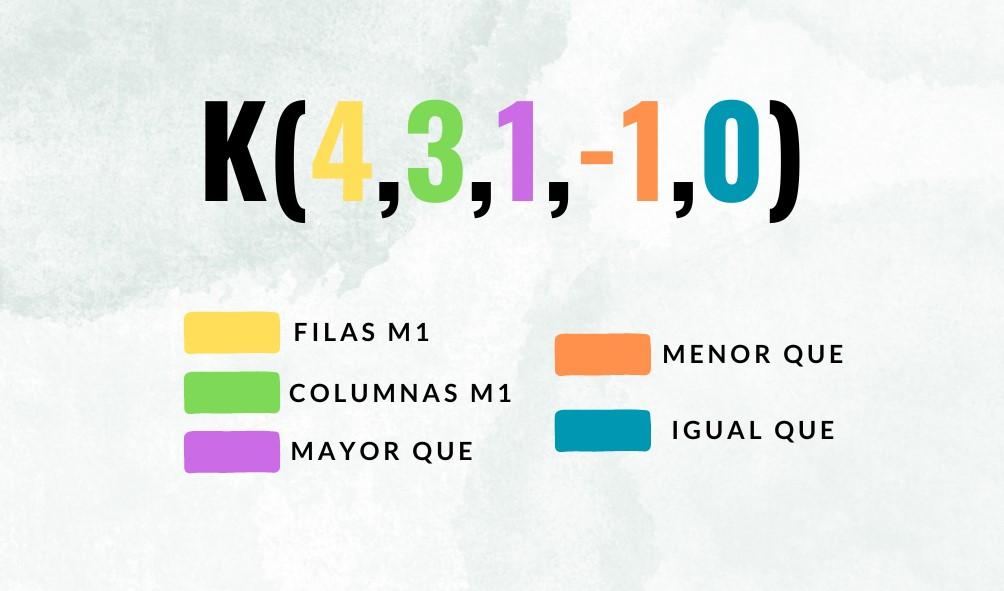
**Filas: 5/2=2.5 Columnas: 5/2=2.5**

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

De acuerdo a lo anterior, la celda encontrada en cada “M” sería la posición neutra, pues así lo plantean las características del problema que estamos solucionando.

**Comportamiento de las claves:**

**Análisis de los elementos que componen a “K”:**



De acuerdo a la imagen anterior, podemos evidenciar una serie de elementos que componen la clave, se ha empleado una ilustración para complementar el proceso de comprensión sobre cómo imprimir las cerraduras a partir de la clave que el usuario ingresa.

1. En primera instancia contamos la cantidad de números de la clave y le restamos 2 para obtener solo los números que están destinados a proporcionar las 3 clases de comparaciones, que han sido definidas como: **“<”**,**”>”** e **”=”**; y de esta manera poder saber el número de arreglos que va a tener nuestra cerradura “X”, el cuál queremos generalizar de la siguiente manera: **(num +1),** donde **“num”** representa la cantidad de comparaciones, y el número **“1”** sirve para completar el número total de arreglos de la cerradura “X”.
2. Luego, se verificará el número mayor entre los dos primeros dígitos y si se tiene en cuenta que estos corresponden a una posición en un arreglo, se deberá entregar una cerradura, en donde el primer elemento corresponda a matriz “M” de una dimensión más grande que el número mayor entre esos dos primeros dígitos de la clave.
3. Después de determinar el número mayor entre los dos primeros dígitos de la clave "K", se procederá a recorrer el primer arreglo "M" de la cerradura "X". Es esencial considerar que, aunque el valor "A" en el arreglo "M" se ubicará en una posición dada por los primeros dos dígitos de la clave "K", sus coordenadas no serán necesariamente iguales a las de los demás valores ubicados en cada una de las “M”.

Debemos conocer el valor de la resta entre dimM1 y dimM2:

**Ejemplo:**

**Dim 7x7 n=7 Dim 5x5 n=5**

**7-5= 2 (i+1,j+1) ó (i-1,j-1)**

**Ejemplo:**

**Dim 9x9 n=9 Dim 5x5 n=5**

**9-5= 4 (i+2,j+2) ó (i-2,j-2)**

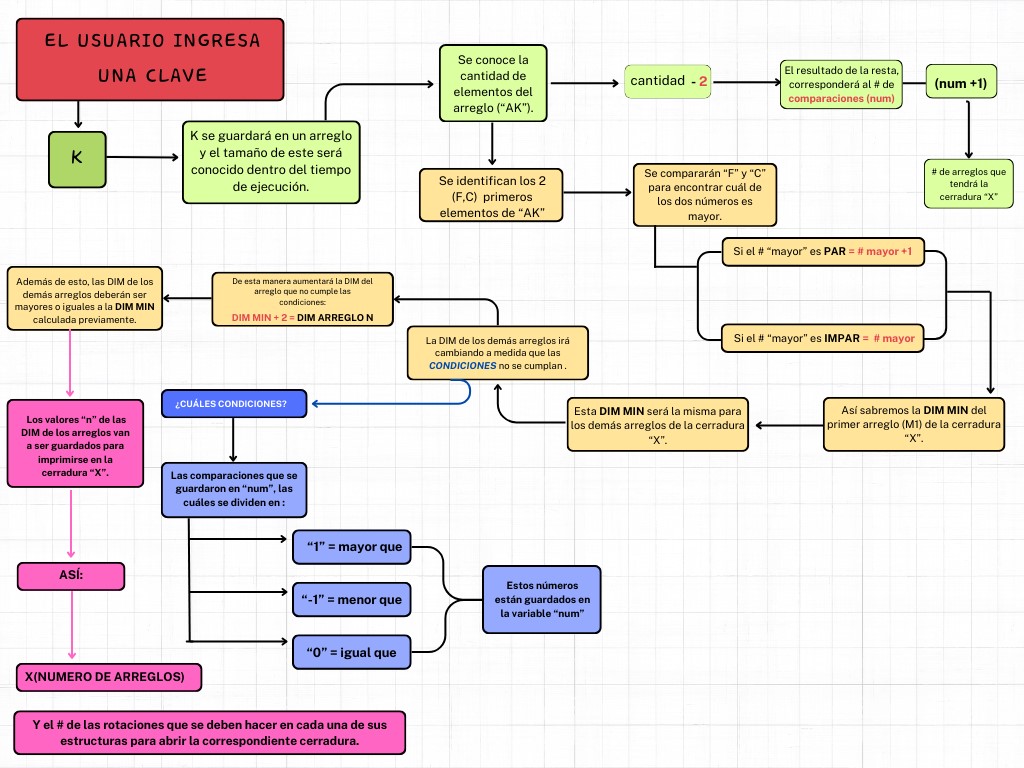
Para averiguar la posición resultante independientemente de la dimensión tendremos que saber si la dimensión siguiente es mayor o menor, si es mayor (dimM1> dimM2) las coordenadas de B serán dadas por (i-s,j-s) de lo contrario; (dimM1<dimM2) las coordenadas de B serán dadas por (i-s,j-s); donde s= el número que está rojo del ejemplo anterior.

**4)** Una forma de averiguar las dimensiones restantes de mi cerradura "X” puede ser por tanteo, donde puedo crear matrices de orden impar hasta que alguna, junto con sus respectivas rotaciones me cumpla las condiciones de comparaciones propuestas en mi clave "K" teniendo en cuenta la posición especial que pudimos encontrar.

**5)** Los valores “n” de mis dimensiones van a ser guardados para emplearlos en la impresión final de mi cerradura "X", lo mismo que las rotaciones de cada matriz a partir de un contador, donde cont = 1 será la posición neutra, cont =2 será la posición 1 y así sucesivamente.

-Todo se imprime y felices los cuatro 🙂

1. **Esquema donde describa las tareas que usted definió en el desarrollo de los algoritmos.**



**d) Problemas de desarrollo que afrontamos:**

1. Entender una estructura multidimensional, plantear el funcionamiento aprender a recorrer dichos arreglos.
2. Comprender el uso de la memoria dinámica y cómo emplearlo en el programa, cómo creo una estructura bidimensional de tamaño variable.
3. Tener en cuenta la utilidad y la sintaxis de los punteros.