**PROYECTO INFORMATICA 2**

* **Análisis del problema:**

Se requiere hacer un programa que reciba como entrada un arreglo unidimensional (regla) la cual tiene números en la primera(A) y segunda posición(B) que se refieren a las “coordenadas” del número que se quiere comparar de un arreglo bidimensional determinado por estos dos primeros números(A,B), por lo que primero se buscará el número mayor de estos dos, luego se determinará si es un número par o impar, si resulta ser un número impar(A), entonces la dimensión del primer arreglo bidimensional resultara ser de orden AxA, pero si el número mayor resulta ser par(B), entonces la dimensión del arreglo será de orden BxB. Luego de identificar el orden del primer arreglo (base), se tomarán los valores de la “regla” que estén en la posición siguiente a los números A y B que habíamos usado antes y los llamaremos “C”, con ello se buscará crear arreglos bidimensionales de dimensiones cuadrados que puedan ser alineados unos detrás de otros, a través de un punto llamado “centro”. Para estos nuevos arreglos que se irán creando se deberán cumplir ciertas condiciones dependiendo del valor de “C”, estos valores pueden ser:

1. (C = -1): esto quiere decir en primera instancia (partiendo desde el arreglo “base”) que el número(N) que este en la ubicación A,B debe ser menor al número(Z) de un arreglo bidimensional que esté alineado con nuestro arreglo Base a través del centro, una vez alineados los centros de los dos arreglos, se hará la comparación entre el número Z y el número del nuevo arreglo que esté alineado directamente. Si no se cumple la condición, se deberá “rotar” el nuevo arreglo tres veces más y si aún no se cumple la condición se deberá intentar los mismo con un arreglo diferente que sea 2 dimensiones mayor a la del arreglo que habíamos probado antes, así hasta que se cumpla la condición y al arreglo que cumpla esta condición lo llamaremos “base2”. Si se cumple la condición entonces se deberá tomar otro valor a C y dependiendo de su valor hacer la respectiva comparación con un número de un arreglo bidimensional nuevo comenzando con el mismo orden del arreglo Base y aumentando de 2 en 2 las dimensiones, así hasta que C sea el último valor de la “regla”.
2. (C = 0): partiendo del arreglo “base” , se busca crear un arreglo bidimensional (base2) cuadrado que se pueda alinear su “centro” con el “centro” de nuestro arreglo “base”, para luego ubicarnos en el número que se encuentre en la posición A,B y hacer la comparación con el número que esté alineado de el nuevo arreglo (base2) a la posición A,B de nuestra arreglo “base”, una vez determinados los dos números a comparar, se debe cumplir condición de que el número en nuestro arreglo “Base2” debe ser igual al número alineado del nuevo arreglo (base2), si no se cumple la condición se deberá “rotar” 3 veces el nuevo arreglo(base2) hasta que se cumpla la condición, luego se deberá cambiar el valor de C a la siguiente posición de la “regla” y generar un nuevo arreglo bidimensional el cual debe partir desde la dimensión de la “Base” inicial e ir aumentando dimensiones de 2 en 2 hasta que cumpla la condición de C aplicando el mismo método ya descrito anteriormente hasta que C sea el último valor de la “regla”.
3. (C = 1): estando en el arreglo “Base” se busca alinear nuevos arreglos bidimensionales(base2) cuadrados que al alinear su “centro” con el “centro” del arreglo ”base”, se compare el número que se encuentre en la ubicación A,B del arreglo “Base” con el número que esté alineado del nuevo arreglo(base2) y se debe cumplir la condición de que el número del primer arreglo debe ser mayor al número que esté alineado del nuevo arreglo(base2), si no se cumple la condición se deberá “rotar” 3 veces el nuevo arreglo(base2) hasta que se cumpla la condición, luego se deberá cambiar el valor de C a la siguiente posición de la “regla” y generar un nuevo arreglo bidimensional el cual debe partir desde la dimensión de la “Base” inicial e ir aumentando dimensiones de 2 en 2 hasta que cumpla la condición de C aplicando el mismo método ya descrito anteriormente hasta que C sea el último valor de la “regla”.

Para la solución del problema se tiene previsto:

* + Pedir al usuario que ingrese la “regla” y eso se hará mediante una función que valide las entradas y se asegure que sean datos validos con los que se pueda trabajar.
  + Se obtendrá la posición del número ubicado en el primer arreglo usando los dos primeros datos de la “regla” y con esto se determinará el tamaño del primer arreglo bidimensional.
  + se hará una función para crear arreglos bidimensionales que contengan valores enteros comenzando con el número uno.
  + Luego de crear el primer arreglo y asegurar el valor del entero en la ubicación dada, se usarán los datos siguientes a los dos primeros de la “regla” para saber las condiciones de comparación, esto se hará iterando sobre esta regla cada vez que se encuentre un arreglo que cumpla con la condición dada (-1,0,1).
  + Se creará una función para “rotar” un arreglo en contra de las manecillas del reloj.
  + Se comparará el numero del primer arreglo con un número del siguiente arreglo creado comenzando desde la misma dimensión del primer arreglo. si las dimensiones del primer arreglo y del siguiente a comparar son las mismas, entonces se comparará un numero en la misma posición de ambos arreglos y se “rotará” el ultimo arreglo hasta que se cumpla la condición (-1,0,1). Si no se cumple la condición incluso “rotando” el arreglo completo, se aumentará 2 veces la dimensión de este ultimo y así mismo se aumentará dos veces las posiciones para comparar los enteros, pero ahora con este nuevo arreglo.
  + Se deberá guardar el orden del arreglo que cumpla con las condiciones dadas por la “regla”. Estas dimensiones estarán en un arreglo que lo llamaremos “cerradura”
* **Consideraciones para solución propuesta**

Se deberán utilizar arreglos dinámicos para facilitar la creación de las estructuras de datos bidimensionales de tamaño variable, además de que se debe validar correctamente la entrada de la “regla” para generar la certeza de trabajar con valores dentro de lo acordado y que el programa no genere inconvenientes, sumado a esto se debe tener presente que, al trabajar con memoria dinámica, también se debe ser responsable y saber “liberarla” cuando sea necesario.

* **Esquema de tareas definidas para el desarrollo de los algoritmos**
* **Algoritmos implementados**
  + rotarM( int \*\*, int): esta función tiene como parámetro un doble puntero a un arreglo bidimensional de tipo entero y un entero que equivaldría a la dimensión del arreglo. Esta función retorna un doble puntero a un arreglo bidimensional que seria igual al arreglo original pero rotado 90 grados en contra de las manecillas del reloj.
  + Validar\_Regla(): esta función se encarga de pedir como entrada cada dato de la “regla” y comprobar si son validos o no, por ultimo retorna un puntero a un arreglo unidimensional de enteros que sería la “regla”.
* **Problemas en el desarrollo**
* **Evolución de la solución y consideraciones para tener en cuenta en la implementación**