Pregunta 2

Implementación:

En un comienzo las funciones entregadas por los profesores para este programa se encargarán de pedir al usuario el tamaño de la matriz que debe ser resultado de una potencia de 2, luego la función **crearMatriz** se encarga de crear una matriz cuadrada con el tamaño indicado por el usuario y asigna valores al azar hasta llenar la matriz.

La resolución de este problema se hará utilizando el método de división y conquista, aprovechando el hecho de que la matriz creada tiene una cantidad de filas y columnas igual a una potencia de 2, dividiremos la matriz original en múltiples matrices de 2x2, de tal manera de reducir el problema a calcular el mayor numero dentro de una matriz de 2x2.

Para realizar este método utilizaremos la función **buscarMayor**, esta función recibirá como argumentos: la matriz, el tamaño de la matriz (tamaño = filas = columnas), **inicio_i** que será el índice de la fila para el "pivote" de la matriz (valor en la esquina superior izquierda), **final_i** que representa hasta que índice abarca la matriz que estamos revisando, **inicio_j** que al igual que **inicio_i** será el índice de la columna para el pivote de la matriz y **final_j** que indica hasta que columna abarca la matriz que estamos revisando.

Primero establecemos el caso base en donde el tamaño de la matriz es de 2 (matriz cuadrada de 2x2), se crea la variable **mayor** en la cual almacenaremos el mayor numero dentro de la matriz actual, luego mediante 4 sentencias, de los cuales la primera sirve para establecer el primer numero de la matriz como el mayor, luego en los 3 siguientes iremos comparando el número mayor local con el resto de los números de la matriz 2x2, si el numero en la posición a revisar es mayor que el numero anterior, este ocupara el lugar del nuevo numero mayor local; una vez revisado los 4 números de la matriz, la función retornará la variable **mayor** que contiene el valor del numero mayor dentro de la matriz 2x2.

En el caso base calculamos el mayor valor de entre cuatro números, en el caso general veremos los cuatro números como cuatro matrices, de esta manera para saber el valor de cada matriz se hará un llamado recursivo para calcular el numero mayor dentro de cada matriz, así luego de calcular el numero mayor de cada matriz, se calculará el numero mayor de entre los cuatro números.

Al igual que en el caso base, crearemos la variable **mayor** para almacenar el valor mayor local, luego mediante 4 if buscaremos el valor mayor de entre las 4 matrices, es en este momento que ocurre la llamada recursiva, llamaremos nuevamente a la función **buscarMayor**, pero ahora lo haremos con la mitad del tamaño de la matriz de esta manera, por ejemplo: de tener una matriz de 4 filas y 4 columnas, ahora tendremos 4 matrices de 2 filas y 2 columnas cada una, esto se repetirá hasta llegar a una matriz de 2x2 entrando al caso base de la recursión. Además, en cada caso iremos variando el **inicio_i, final_i, inici_j, final_j** de esta manera sabremos donde se encuentra el "pivote" de cada pequeña matriz que generaremos.

Al final luego de terminar los llamados recursivos, terminaremos con 4 valores, estos al igual que el caso base, se compararán e irán reemplazando al valor de mayor cuando sean mayores que el valor anterior. Luego de comparar los cuatro valores de las cuatro matrices originales, la función retorna la variable **mayor** que contiene el mayor valor dentro de la matriz original saliendo de la función.

El valor de la búsqueda en la matriz será almacenado en la variable **resultado**, para luego ser mostrada por pantalla junto con la matriz en donde se realizó esta búsqueda. Finalmente se libera la memoria utilizada por el arreglo en dos dimensiones de la matriz y se retorna cero terminado el programa.