# Preparación de tarea

La siguiente tabla representa a la matriz espiral, y se genera moviéndose desde la posición central de la misma (la que aparece en fondo negro marcado en rojo) hacia la derecha, realizando un movimiento en el sentido de las manecillas del reloj.

| 43 | 44 | 45 | 46 | 47 | 48 | 49 |
| --- | --- | --- | --- | --- | --- | --- |
| 42 | 21 | 22 | 23 | 24 | 25 | 26 |
| 41 | 20 | 7 | 8 | 9 | 10 | 27 |
| 40 | 19 | 6 | 1 | 2 | 11 | 28 |
| 39 | 18 | 5 | 4 | 3 | 12 | 29 |
| 38 | 17 | 16 | 15 | 14 | 13 | 30 |
| 37 | 36 | 35 | 34 | 33 | 32 | 31 |

Una matriz se puede representar en forma de arreglo bi-dimensional. Por ejemplo, el siguiente snippet de código declara una matriz de 7 x 7, y la inicializa con ceros en la misma línea:

*int* matrix[**7**][**7**] = {0};

Notar que las posiciones de la matriz se manejan por coordenadas. En la representación a través de arreglos bi-dimensionales, las coordenadas serán los números entre paréntesis cuadrados. Por ejemplo, en el caso de la matriz espiral, sus valores pueden ser accedidos a través de coordenadas de la siguiente manera:

printf("%d", matrix[**1**][**1**]); // 21

printf("%d", matrix[**1**][**3**]); // 23

printf("%d", matrix[**3**][**1**]); // 19

printf("%d", matrix[**5**][**2**]); // 16

printf("%d", matrix[**2**][**5**]); // 10

printf("%d", matrix[**6**][**0**]); // 37

La primera coordenada es la fila, y la segunda coordenada es la columna. **Nota**: Recordar que los arreglos están indexados desde 0 hasta N - 1. Si nuestro N es 7, entonces, los índices para nuestra matriz espiral se manejarán en el rango desde 0 hasta 6. Las posiciones matrix[**7**][**x**] y matrix[**x**][**7**], con ‘*x’* siendo un número cualquiera entre 0 y 6, **NO son accesibles**.

Supongamos ahora que queremos imprimir todos los elementos de la fila 5. ¿Cómo lo haríamos? La fila es estática, por lo que simplemente iteramos por todas las columnas pertenecientes a la fila. El código resultante quedaría de la forma siguiente:

*for* (*int* j = 0; j < **N**; j++) {

printf("%d ", matrix[5][j]);

}

El resultado sería el siguiente:

| 38 17 16 15 14 13 30 |
| --- |

¿Y si quisiésemos imprimir todos los elementos de la columna 5? La columna es estática, por lo que simplemente iteramos por todas las filas pertenecientes a la columna:

*for* (*int* i = 0; i < **N**; i++) {

printf("%d ", matrix[i][5]);

}

El resultado sería el siguiente:

| 48 25 10 11 12 13 32 |
| --- |

# Postulado de tarea

Diríjase al servidor de Discord de la carrera, y descargue el código que se subió al canal #tutoria-intro-material. En el código, aparecen unos métodos ya implementados, como por ejemplo, cómo rellenar la matriz mostrada en la sección anterior. **NO intente entender el código implementado**, su labor no es esa. Lo que le compete a usted es proveer una implementación para los métodos que han sido declarados, pero que carecen de implementación. A continuación, procederemos a documentar lo que hacen los métodos, para que tenga una idea clara de lo que debe hacer.

// Imprime por consola el contenido de la fila especificada como parámetro.

Argumentos:

matrix: La matriz, previamente rellenada.

row: La fila que se desea imprimir.

*void* showRow(*int* matrix[][**N**], *int* row);

**Ejemplo de uso**:

showRow(matrix, 3);

**Output**:

| 40 19 6 1 2 11 28 |
| --- |

// Imprime por consola el contenido de la columna especificada como parámetro.

Argumentos:

matrix: La matriz, previamente rellenada.

col: La columna que se desea imprimir.

*void* showCol(*int* matrix[][**N**], *int* col);

**Ejemplo de uso**:

showCol(matrix, 1);

**Output**:

| 44 21 20 19 18 17 36 |
| --- |

// Imprime por consola EN ORDEN INVERSO el contenido de la fila especificada como parámetro.

Argumentos:

matrix: La matriz, previamente rellenada.

row: La fila que se desea imprimir.

*void* showRowReversed(*int* matrix[][**N**], *int* row);

**Ejemplo de uso**:

showRowReversed(matrix, 4);

**Output**:

| 29 12 3 4 5 18 39 |
| --- |

// Imprime por consola EN ORDEN INVERSO el contenido de la columna especificada como parámetro.

Argumentos:

matrix: La matriz, previamente rellenada.

col: La columna que se desea imprimir.

*void* showColReversed(*int* matrix[][**N**], *int* col);

**Ejemplo de uso**:

showColReversed(matrix, 6);

**Output**:

| 31 30 29 28 27 26 49 |
| --- |

// Imprime fila a fila el contenido de toda la matriz.

Argumentos:

matrix: La matriz, previamente rellenada.

*void* showMatrixByRow(*int* matrix[][**N**]);

**Ejemplo de uso**:

showMatrixByRow(matrix);

**Output**:

| 43 44 45 46 47 48 49  42 21 22 23 24 25 26  41 20 7 8 9 10 27  40 19 6 1 2 11 28  39 18 5 4 3 12 29  38 17 16 15 14 13 30  37 36 35 34 33 32 31 |
| --- |

// Imprime columna a columna el contenido de toda la matriz.

Argumentos:

matrix: La matriz, previamente rellenada.

*void* showMatrixByCol(*int* matrix[][**N**]);

**Ejemplo de uso**:

showMatrixByCol(matrix);

**Output**:

| 43 42 41 40 39 38 37  44 21 20 19 18 17 36  45 22 7 6 5 16 35  46 23 8 1 4 15 34  47 24 9 2 3 14 33  48 25 10 11 12 13 32  49 26 27 28 29 30 31 |
| --- |

// Imprime fila a fila el contenido EN ORDEN INVERSO de toda la matriz.

Argumentos:

matrix: La matriz, previamente rellenada.

*void* showMatrixByRowReversed(*int* matrix[][**N**]);

**Ejemplo de uso**:

showMatrixByRowReversed(matrix);

**Output**:

| 49 48 47 46 45 44 43  26 25 24 23 22 21 42  27 10 9 8 7 20 41  28 11 2 1 6 19 40  29 12 3 4 5 18 39  30 13 14 15 16 17 38  31 32 33 34 35 36 37 |
| --- |

// Imprime columna a columna el contenido EN ORDEN INVERSO de toda la matriz.

Argumentos:

matrix: La matriz, previamente rellenada.

*void* showMatrixByColReversed(*int* matrix[][**N**]);

**Ejemplo de uso**:

showMatrixByColReversed(matrix);

**Output**:

| 37 38 39 40 41 42 43  36 17 18 19 20 21 44  35 16 5 6 7 22 45  34 15 4 1 8 23 46  33 14 3 2 9 24 47  32 13 12 11 10 25 48  31 30 29 28 27 26 49 |
| --- |