1. Dada una matriz de 3x3, se necesita cargarla con valores que no sea repetidos, una vez cargada, debe finalizar la misma y poder mostrarla.

2. Imagina que estás desarrollando un software para controlar una pantalla LED de 8x8 píxeles. Cada píxel en la pantalla puede tener un color, y cada color se define por sus componentes Rojo (R), Verde (G) y Azul (B), conocidos como valores RGB.

3. En este ejercicio, deberás crear un programa en C que simule esta **pantalla de 8x8 píxeles**, asignando un color aleatorio a cada píxel. Los colores deben ser representados usando el formato RGB, donde cada componente (R, G, B) es un valor entre 0 y 255.
   Tareas:
   a. **Asignación de colores:** Implementa una función que asigne un color aleatorio a cada uno de los 64 píxeles de la pantalla. Recuerda que cada color está compuesto por tres valores: Rojo (R), Verde (G) y Azul (B), cada uno de ellos variando entre 0 y 255.
   b. **Visualización de la pantalla:** Implementa una función que muestre en la consola los colores asignados a cada píxel de la pantalla. La salida debería ser un formato legible que muestre claramente los valores RGB para cada píxel.
   c. **Estructura del Programa:** En el main, asegúrate de inicializar los valores necesarios, invocar la función de asignación de colores, y luego mostrar la pantalla en la consola.
   Consideraciones:
   d. Cada píxel de la pantalla debe ser independiente y tener un color único determinado por tres valores (R, G, B).
   e. La representación de los colores en la consola debe ser clara y organizada, de manera que se pueda identificar fácilmente el color de cada píxel.
   Una salida esperada seria la siguiente
   (123, 234, 45) (67, 89, 255) (12, 34, 56) ...
   (78, 90, 12) (45, 67, 89) (234, 123, 67) ...
   ... donde cada unes es R,G,B.

4) En este ejercicio, vas a simular las conexiones entre **usuarios en una red social**. Cada usuario en la red será representado por un nodo en una matriz de adyacencia, donde se indica si dos usuarios están conectados (es decir, si son amigos).

Requerimientos:

#define NODOS 10 // Número de nodos en la red (usuarios)

#define PROBABILIDAD_CONEXION 10 // Probabilidad de conexión en porcentaje o sea 10%

a) **Matriz de Conexiones**:
   i) Debes crear una matriz `redSocial[NODOS][NODOS]`, donde cada elemento `[i][j]` será:
      (1) `1` si el usuario `i` está conectado con el usuario `j` (es decir, son amigos).
      (2) `0` si no hay conexión entre `i` y `j`.
      (3) La red social tiene `NODOS` usuarios, donde `NODOS` es una constante definida en el código.

b) **Inicialización de la Matriz**:
   i) La matriz debe inicializarse con ceros, indicando que inicialmente no hay conexiones entre los usuarios.

c) **Conexiones Aleatorias**:
   i) Utiliza un generador de números aleatorios para decidir si debe existir una conexión entre dos usuarios.
   ii) La probabilidad de conexión entre cualquier par de usuarios viene dada por la constante `PROBABILIDAD_CONEXION`, que se expresa como un porcentaje (por ejemplo, 10% de probabilidad de que dos usuarios estén conectados).

d) **Bidireccionalidad**:
   i) Si el usuario `i` está conectado con el usuario `j`, la conexión debe ser bidireccional, es decir, la matriz debe reflejar esta relación en ambas direcciones (`matriz[i][j] = 1` y `matriz[j][i] = 1`).

e) **Visualización**:
   i) Finalmente, muestra la matriz resultante para visualizar las conexiones entre los usuarios.

5. Dada una matriz de 14x2 determinar por fila si ese par de números es amigo, si así lo fuese informarlo y determinar cuantos de ellos son amigos.
   Armar las funciones que se proponen
   // Prototipos de funciones

   int sumaDivisoresPropios(int num);
   void verificarNumerosAmigos(int matriz[14][2]);

   // Matriz de pares de números precargada

int Mat[14][2] = { {220, 284}, {10, 20}, {1184, 1210}, {21, 28}, {2620, 2924}, {30, 40}, {5020, 5564}, {6232, 6368}, {36, 45}, {10744, 10856}, {12285, 14595}, {17296, 18416}, {28520, 55272}, {6144, 6680} };

Marco Teórico para entenderlo

Un **número amigo** es un término que se refiere a un par de números naturales, donde cada número es la suma de los divisores propios del otro. Los divisores propios de un número son aquellos divisores menores que el número en sí.

Definición Formal

Dos números A y B se llaman números amigos si se cumplen las siguientes condiciones:

- La suma de los divisores propios de A es igual a B.
- La suma de los divisores propios de B es igual a A.

Ejemplo Clásico

El par de números amigos más famoso es **220** y **284**.

1. **Divisores propios de 220**:
    - Los divisores de 220 (**excluyendo 220**) son: 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110.
    - La suma de estos divisores es: estos recién mencionados
    - 1+2+4+5+10+11+20+22+44+55+110=**284**

Divisores propios de 284:

- Los divisores de 284 (**excluyendo 284**) son: 1, 2, 4, 71, 142.
    - La suma de estos divisores es: 1+2+4+71+142= **220**

Solo debes **verificar hasta la mitad de 220**, que es 110. Esto es porque un divisor mayor que la mitad de un número **no puede ser un divisor propio** (excepto el número mismo).

Ejemplo Práctico:

- Consideremos el número 220.
- Si intentamos verificar divisores mayores que 220/2 = 110, digamos 150. Para que 150 sea un divisor de 220, necesitaríamos que 220/150 sea un número entero. Pero 220/150 ≈ 1.472
- Esto demuestra que cualquier divisor mayor que 110 no puede ser un divisor propio de 220. Los divisores mayores que n/2 ya tienen sus pares multiplicativos menores que n/2