Laboratorio 3

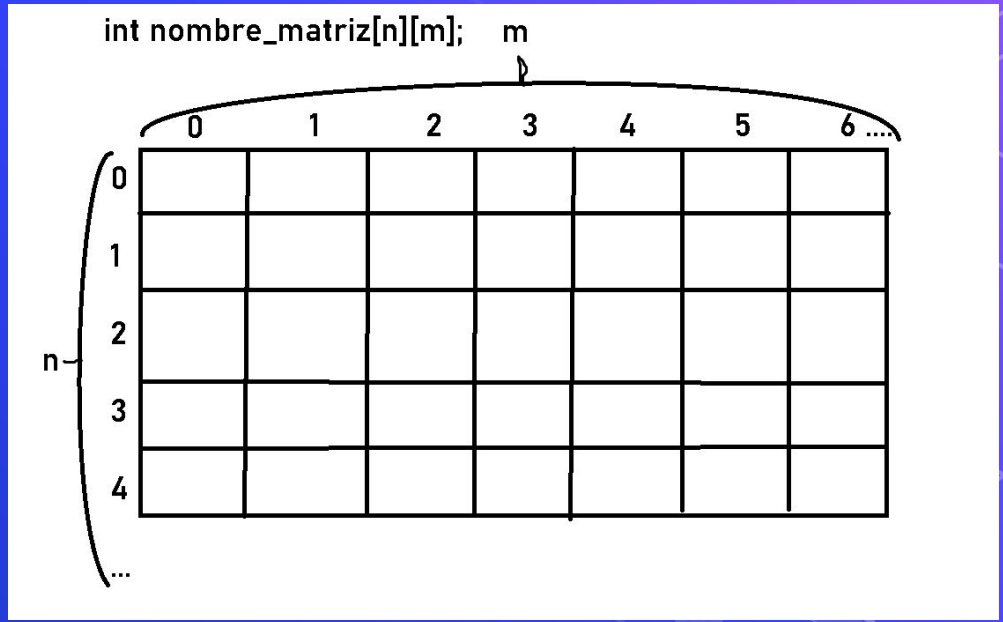
Programación de Computadores (2023-2)

Temas del Laboratorio 3

- Matrices
- Biblioteca matemática ***math.h***
- Ejercicios

Matrices

Las matrices en C se pueden entender como un arreglo de arreglos, se representa de arriba a abajo y de izquierda a derecha (como la lectura de texto) y ocupan un espacio continuo en memoria.



Declaración de matrices en C

Al igual que los array, se declaran especificando el tipo de dato de los elementos y cuantos podrá contener, pero en este caso se especifica tanto las filas como las columnas

```
int matrizInt[3][3];
double matrizFloat[6][7];
char matrizChar[4][2];
```

También será posible crear matrices de más dimensiones, por ejemplo una matriz 3D podría decirse que es un arreglo de arreglos de arreglos (y así sucesivamente con más dimensiones).

```
int matriz3D[4][2][7];
int matriz4D[2][6][9][5];
int matriz8D[3][1][2][6][7][6][10][9];
```

Matrices en C

Se pueden declarar y rellenar de la siguiente forma:

```
int matriz[4][5] = {{ 0, 1, 2, 3, 4},
                    { 5, 6, 7, 8, 9},
                    {10,11,12,13,14},
                    {15,16,17,18,19} };
```

Se accede a sus valores de la siguiente forma:

```
matriz[0][0];
matriz[1][3];
matriz[3][4];
```

Acceso a matrices en C

```c
printf("funcionamiento correcto:\n");
printf("matriz[0][4]: %d\n", matriz[0][4]);
printf("matriz[1][2]: %d\n", matriz[1][2]);
printf("matriz[3][4]: %d\n", matriz[3][4]);

printf("funcionamiento incorrecto:\n");
printf("matriz[0][5]: %d\n", matriz[0][5]);
printf("matriz[1][14]: %d\n", matriz[1][14]);
printf("matriz[3][5]: %d\n", matriz[3][5]);
```

```
funcionamiento correcto:
matriz[0][4]: 4
matriz[1][2]: 7
matriz[3][4]: 19
funcionamiento incorrecto:
matriz[0][5]: 5
matriz[1][14]: 19
matriz[3][5]: 609995040
```

Biblioteca *math.h* de C

La biblioteca math.h en C permite utilizar funciones y constantes matemáticas predefinidas para la realización de operaciones matemáticas avanzadas, como funciones trigonométricas, exponenciales, logarítmicas entre otras.

Para usar las funciones y constantes de esta biblioteca se debe incluir la biblioteca **math.h** en el código del programa.

```
#include <math.h>
```

Para compilar los programas que usen estas bibliotecas se debe agregar el argumento **`-lm`** al momento de usar gcc

```
gcc -o programa codigo.c -lm
```

Algunas funciones de math.h

Función	Significado	Ejemplo de uso
`ceil(x)`	Redondeo hacia arriba de x	`ceil(10.8) = 11`
`floor(x)`	Redondeo hacia abajo de x	`floor(10.8) = 10`
`fabs(x)`	Valor absoluto de x	`fabs(-45.6) = 45.6`
`sqrt(x)`	Raíz cuadrada de x	`sqrt(16) = 4`
`pow(x,y)`	x elevado a y	`pow(2,3) = 8`
`log(x)`	Logaritmo natural de x	`log(15) = 2.708`
`log10(x)`	Logaritmo decimal de x	`log10(15) = 1.176`

Algunas funciones trigonométricas

Función	Significado
`sin(x)`	Seno de x
`cos(x)`	Coseno de x
`tan(x)`	Tangente de x
`asin(y)`	Arcoseno de y
`acos(y)`	Arcocoseno de y
`atan(y)`	Arcotangente de y

x: Ángulo en radianes

Entrega el valor del ángulo en radianes

Ejemplo 1: Distancia euclidiana entre dos puntos

```c
double x1, y1, x2, y2;

printf("Ingrese las coordenadas del primer punto (x y): ");
scanf("%lf %lf", &x1, &y1);

printf("Ingrese las coordenadas del segundo punto (x y): ");
scanf("%lf %lf", &x2, &y2);

double distancia = sqrt(pow(x2 - x1, 2) + pow(y2 - y1, 2));

printf("La distancia entre los puntos (%.2lf, %.2lf) y
(%.2lf, %.2lf) es: %.2lf\n", x1, y1, x2, y2, distancia);
```

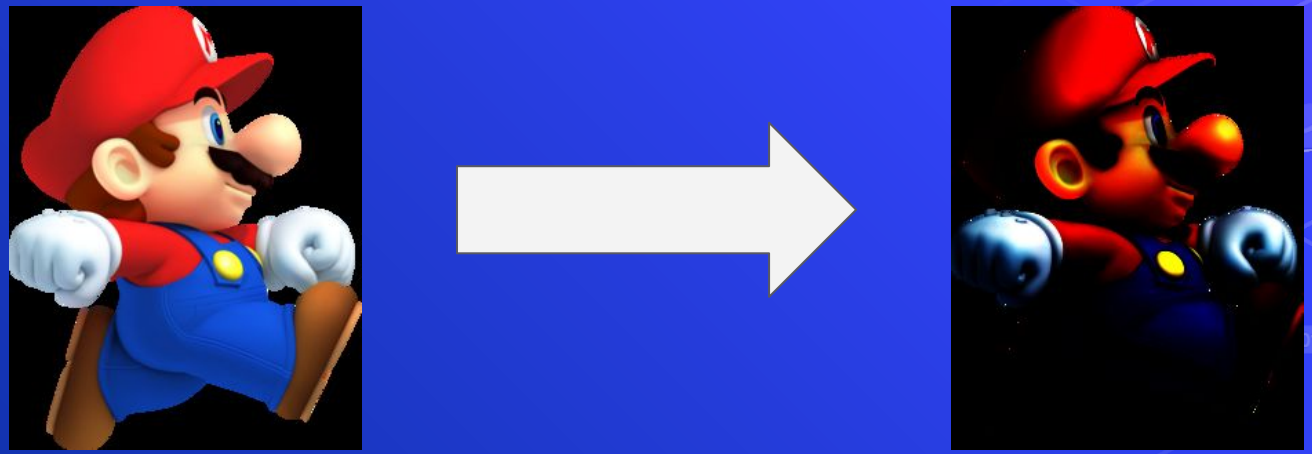
Ejemplo 2: Obtener un ángulo en grados a partir de la tangente

```c
double tangente;

printf("Ingrese el valor de la tangente: ");
scanf("%lf", &tangente);

// Calcular el ángulo en radianes usando la función trigonométrica atan
double angulo_radianes = atan(tangente);

// Convertir el ángulo de radianes a grados
double angulo_grados = angulo_radianes * 180.0 / M_PI;

printf("El ángulo correspondiente a la tangente %.2lf es %.2lf
grados\n", tangente, angulo_grados);
```

Constante con valor aproximado de π

Matrices en imágenes

Sabemos que una imagen es una matriz de pixeles, donde cada pixel está definido en RBG por 3 valores (R,G,B) que varían entre 0 y 255, de manera que cada pixel es una tupla con 3 valores, de esta forma podemos variar los valores en cada pixel para modificar la imagen a nuestra conveniencia, como rotar, escalar, cambiar tonalidad o ajustar el gamma de la imagen.



Transformación logarítmica

Es una técnica de procesamiento digital que aplica una función logarítmica a los valores RGB de cada pixel, aumentando la visibilidad de zonas oscuras y disminuyendo la saturación en zonas muy claras.

Definida por :

$$M = c * \ln(1 + r)$$

M —---> Nuevo valor

c —----> Constante de escala que asegura que el resultado esté entre 0 - 255

r —----> Valor original del pixel

Nota: Definiremos $c = 255 / \ln(1+255)$ para mantenernos en el rango pedido.

Corrección gamma

La corrección gamma es una transformación no lineal que ajusta el brillo de los pixeles para adaptarla a distintos tipos de visualización, está dada por la siguiente fórmula:

$$M = c * ((r /255) ^ b)$$

M —---> Nuevo valor

c —----> Constante de escala (para nosotros será 255)

r —---> Valor original del pixel

b —---> Exponente

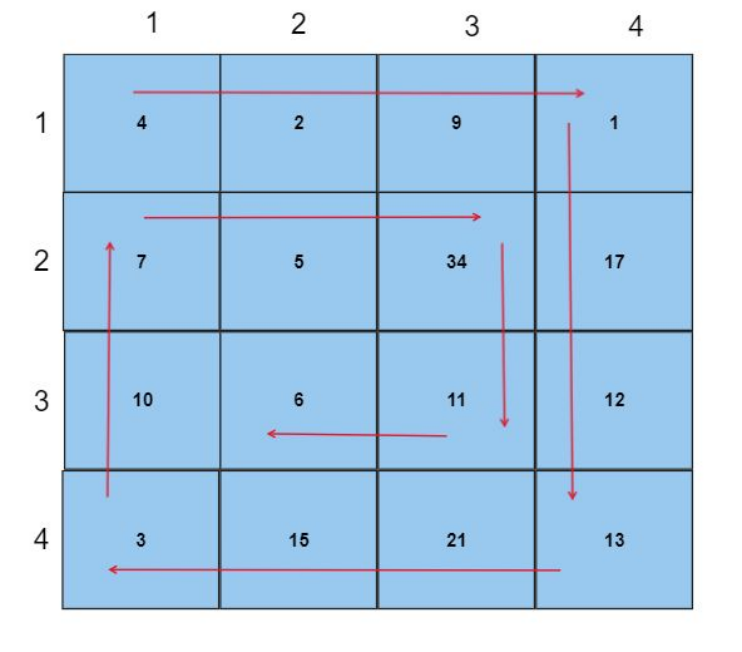
Con b > 1 la imagen se oscurece, con 0 < b < 1 se aclara y con b = 1 no cambia

Qué pasa con b = 0? O con b < 0?

Ejercicio 1

Escribir un programa que pida un entero “n”, crear una matriz “n x n” , rellenar la matriz y luego imprimir los valores en orden de espiral

Ejemplo:



Output:

4 2 9 1 17 12 13 21 15 3 10 7 5 34 11 6

Ejercicio 2

En el repositorio del curso, en la pestaña de laboratorios, encontrará una carpeta con material complementario, dentro de ella verá un par de archivos .c y .h, abra el archivo llamado lab3.c dentro de él, verá (Fuera del main) código que no es necesario que entienda, el resto de los archivos contienen código que genera una imagen mediante valores RGB, una vez ejecutado lab3.c, observará que se crean 2 imágenes dentro de la carpeta, su objetivo es modificar el main de lab3.c para realizar lo siguiente:

1. Aplicar transformación logarítmica a la imagen de salida
2. Aplicar corrección gamma a la imagen de salida, con posibilidad de seleccionar el exponente a utilizar

   Recomendación: En 2, pase a float la operación r /255