

**UNIVERSIDAD AUTONOMA DE BAJA CALIFORNIA**

**FACULTAD DE INGENIERIA, ARQUITECTURA Y DISEÑO**

**INGENIERIA EN SOFTWARE Y TECNOLOGIAS EMERGENTES**

**GRUPO: 932**

**MATERIA: Lenguaje C**

**MAESTRA: Yulith Vanessa Altamirano Flores**

**TITULO:**

**Práctica 7: Punteros**

**ALUMNA: *Teresa Rivas Gómez***

**MATRICULA: *372565***

**CÓDIGO:**

***Solo partes que considero importantes:***

- Función: **ARGUMENTOS EN LA LINEA DE COMANDOS**

    - Descripción: Toma dos números enteros como argumentos desde la linea

    de comandos y procede a realizar operaciones aritméticas.

    - Parámetros: int argc, numero de cadenas en la matriz y char \*argv[],

    matriz de cadena de argumentos en la linea de comandos.

- Valor de retorno: 0, si se ejecuta sin errores.

*int* actividad1(*int* *argc*, *char* \**argv*[])

{

    if (*argc* != 3)

    {

        printf("INTRODUCE 2 NUMEROS ENTEROS.\n");

        return 1;

    }

    /\*------------------------------------------------------------------------------------------\*/

    // Convertir argumentos en numero entero con atoi

*int* num1 = atoi(*argv*[1]);

*int* num2 = atoi(*argv*[2]);

    /\*------------------------------------------------------------------------------------------\*/

    // Operaciones

    printf("SUMA DE %d Y %d ES: %d\n", num1, num2, num1 + num2);

    printf("RESTA DE %d Y %d ES: %d\n", num1, num2, num1 - num2);

    printf("MULTIPLICACION DE %d Y %d ES: %d\n", num1, num2, num1 \* num2);

    printf("DIVISION DE %d Y %d ES: %d\n", num1, num2, num1 / num2);

    /\*------------------------------------------------------------------------------------------\*/

    return 0;

}

- Función: **ORDENACION DE PALABRAS**

    - Descripción: Arreglo de apuntadores a caracteres,

    donde cada apuntador apunta a una palabra, se ordenan alfabéticamente y se imprimen.

    - Parámetros:

    + En la funcion void ordenar: int argc, numero de cadenas en la matriz y char \*argv[],

    matriz de cadena de argumentos en la linea de comandos.

    + En la función int actividad2: no tiene.

    - Valor de retorno:

    + En la función void ordenar: no tiene.

+ En la función int actividad2: a 0.

*void* ordenar(*char* \*\**palabras*, *int* *n*)

{

*int* i, j;

    // Puntero temporal

*char* \*temp;

    for (i = 0; i < *n* - 1; i++)

    {

        for (j = 0; j < *n* - i - 1; j++)

        {

            // Compara dos palabras y las intercambia si estan fuera de orden

            if (strcmp(*palabras*[j], *palabras*[j + 1]) > 0)

            {

                // Intercambia los punteros de las palabras para reordenarlas

                temp = *palabras*[j];

*palabras*[j] = *palabras*[j + 1];

*palabras*[j + 1] = temp;

            }

        }

    }

}

*int* actividad2()

{

    // Arreglo de palabras

*char* \*palabras[] = {"FRESA", "KIWI", "MANGO", "FRAMBUESA", "MANZANA", "PLATANO"};

    // Para calcular el numero de palabras que tiene el arreglo

*int* n = sizeof(palabras) / sizeof(palabras[0]);

    /\*------------------------------------------------------------------------------------------\*/

    // Ordena las palabras antes de imprimirlas

    ordenar(palabras, n);

    /\*------------------------------------------------------------------------------------------\*/

    printf("\nCON ORDEN ALFABETICO:\n");

    // Ciclo para imprimir las palabras

    for (*int* i = 0; i < n; i++)

    {

        printf("%s\n", palabras[i]);

    }

    /\*------------------------------------------------------------------------------------------\*/

    return 0;

}

- Función: **MATRIZ MULTIDIMENSIONAL**

    - Descripción: Matriz bidimensional de enteros, por ejemplo, 3x3, con algunos

    valores. Calcula la suma de todos los elementos de la matriz y encuentra el elemento

    mas grande en la matriz y muestra su valor y posición.

    - Parámetros: no tiene.

- Valor de retorno: 0, si se ejecuta sin errores

*int* actividad3()

{

*int* i;

*int* matriz[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

    // Declaramos un puntero y le damos la dirección de la primera posición en la matriz

*int* \*p = &matriz[0][0];

    /\*------------------------------------------------------------------------------------------\*/

    // Imprimir la matriz

    for (*int* i = 0; i < 3; i++)

    {

        for (*int* j = 0; j < 3; j++)

        {

            printf("%d ", matriz[i][j]);

        }

        printf("\n");

    }

    /\*------------------------------------------------------------------------------------------\*/

    // Suma de todos los valores, utilizamos un ciclo para recorrerla.

*int* suma = 0;

    for (i = 0; i < 3 \* 3; i++)

    {

        suma += \*(p + i);

    }

    printf("La suma de TODOS los elementos de la matriz es: %d\n", suma);

    /\*------------------------------------------------------------------------------------------\*/

    // Para encontrar el elemento mas grande de la matriz, que valor tiene y donde esta ubicado

*int* max = \*p;

    // Para proporcionar la ubicacion

*int* fila = 0;

*int* columna = 0;

    // Ciclo que recorre la matriz

    for (i = 0; i < 3 \* 3; i++)

    {

        // Accedemos a los valores de la matriz y si el elemento es mayor al maximo se actualiza

        if (\*(p + i) > max)

        {

            // Y se almacena en la variable max junto con su posicion

            max = \*(p + i);

            fila = i / 3;

            columna = i % 3;

        }

    }

    printf("El elemento MAS GRANDE de la matriz es: %d\n", max);

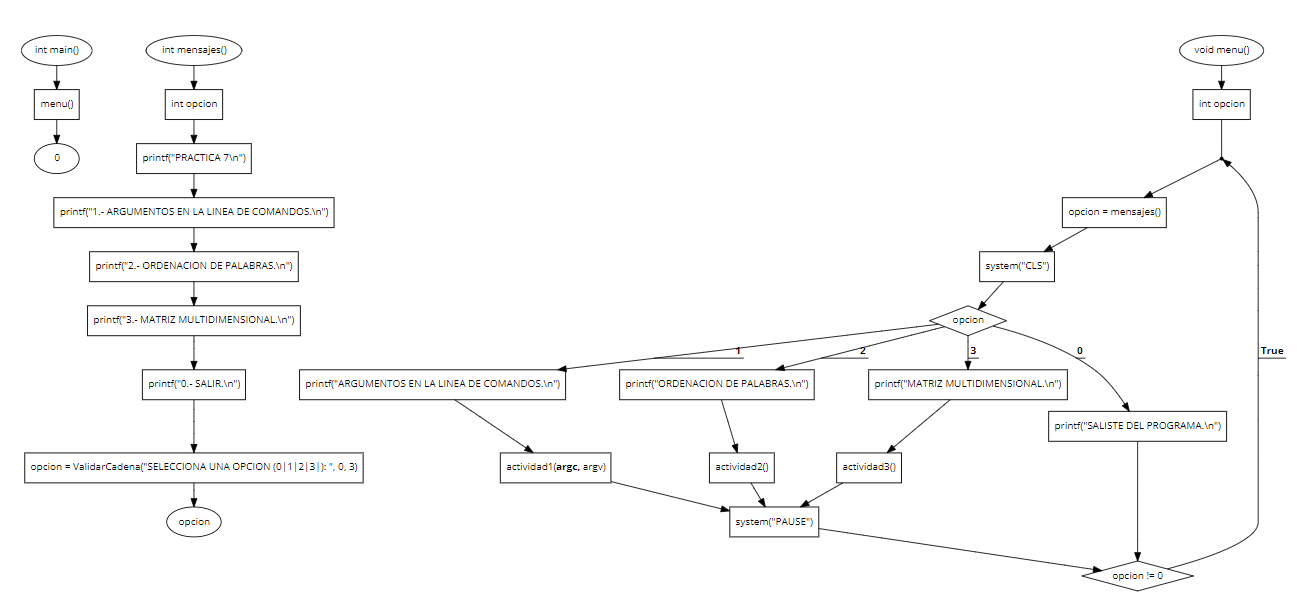
    printf("En la fila %d y columna %d\n", fila, columna);

    /\*------------------------------------------------------------------------------------------\*/

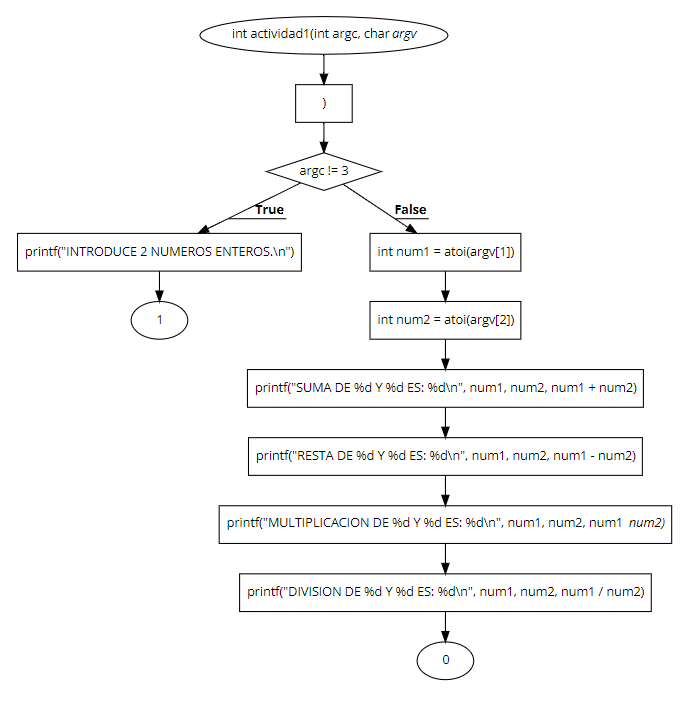
    return 0;

}

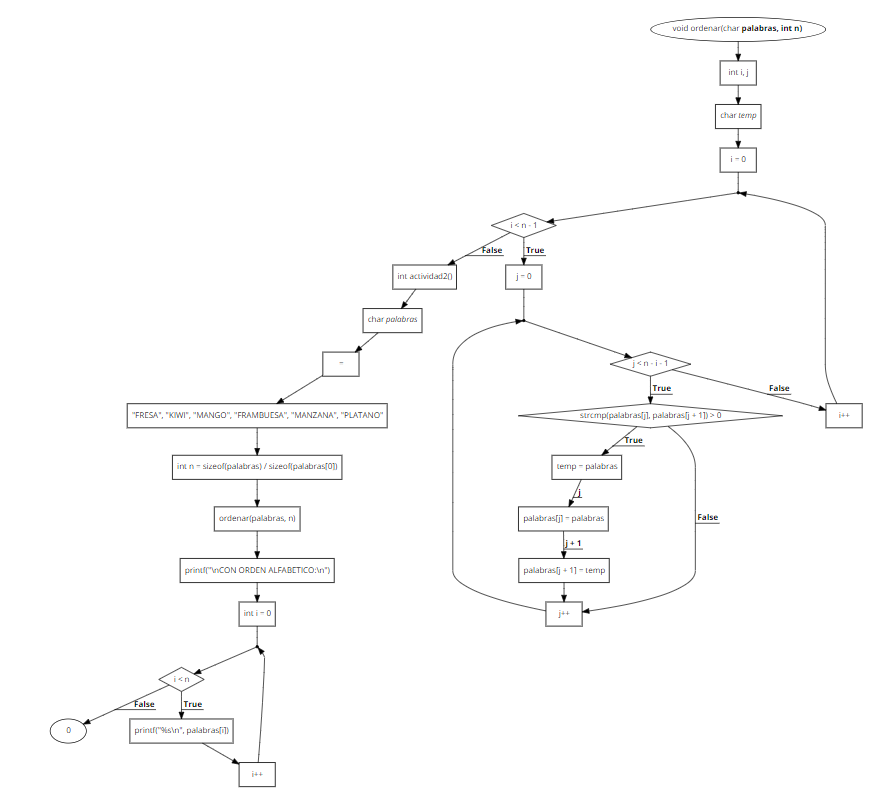
**DIAGRAMA DE FLUJO:**

***Solo partes que considero importantes:***

***ACTIVIDAD 1:***



***ACTIVIDAD 2:***



***ACTIVIDAD 3:***

