	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Edgar Tista G

Asignatura: Estructuras de Datos y Algoritmos I

Grupo: 2

No. de práctica(s): 1

Integrante(s): Álvarez López Carlos Manuel

No. de lista o brigada: 2

Semestre: 2023-2

Fecha de entrega: Jueves 2 de Marzo del 2023

Observaciones:

CALIFICACIÓN: _____

Objetivo Lab: Utilizar arreglos unidimensionales y multidimensionales para dar solución a problemas computacionales

Objetivo Clase: El alumno realizará aplicaciones en las que se utilicen arreglos de tipos datos primitivos. Implementará apropiadamente el manejo de los índices en arreglos multidimensionales

Descripción de los ejemplos

- Escitala espartana

En el ejemplo se nos muestra un programa que hace uso de arreglos unidimensionales y multidimensionales, para encriptar y desencriptar un mensaje en un proceso similar a una escítala.

El código cuenta con un menú y dos funciones que serán las que permiten encriptar y desencriptar el mensaje.

```
void crearMensaje();
void descifrarMensaje();

int main(){
    short opcion=0;

    while (1){
        printf("\n\t*** ESCÍTALA ESPARTANA ***\n");
        printf("¿Qué desea realizar?\n");
        printf("1) Crear mensaje cifrado.\n");
        printf("2) Descifrar mensaje.\n");
        printf("3) Salir.\n");
        scanf("%d", &opcion);
        switch(opcion){
            case 1:
                crearMensaje();
                break;
            case 2:
                descifrarMensaje();
                break;
            case 3:
                return 0;
            default:
                printf("Opción no válida.\n");
        }
    }
    return 0;
}
```

El código aprovecha los arreglos multidimensionales para organizar el mensaje en forma de matriz, llenando la misma por renglones. Seguido de esto, muestra el mensaje cifrado transponiendo la matriz.

```
printf("Escriba el texto a descifrar:\n");
scanf("%s", texto);

for (i=0 ; i<col ; i++)
    for (j=0 ; j<ren ; j++)
        escitala[j][i] = texto[k++];

printf("El texto descifrado es:\n");
for (i=0 ; i<ren ; i++)
    for (j=0 ; j<col ; j++)
        printf("%c", escitala[i][j]);
```

Para descifrar un mensaje, se vuelve a tomar el texto y se hace un proceso inverso al realizado anteriormente, ya que se llena de nuevo una matriz empezando por las columnas, y volviendo a transponer la matriz para que esta quede lista para su lectura antes de la encriptación.

Ejercicio 1

Línea 5: Hace falta un “;” al final de la línea para indicar el final de la declaración del arreglo a1.

Línea 6: El tamaño del arreglo es insuficiente para guardar toda la cadena, si se deja así solo se guardaría “Algor”, y se ignorará el resto de la cadena.

Línea 7: Para que el arreglo se despliegue correctamente en el segundo parámetro del for, debería de estar “i<2”.

Línea 8: No se inicializa la variable j.

Línea 9: Para que el arreglo se despliegue correctamente en el segundo parámetro del for, debería de estar “k<5”.

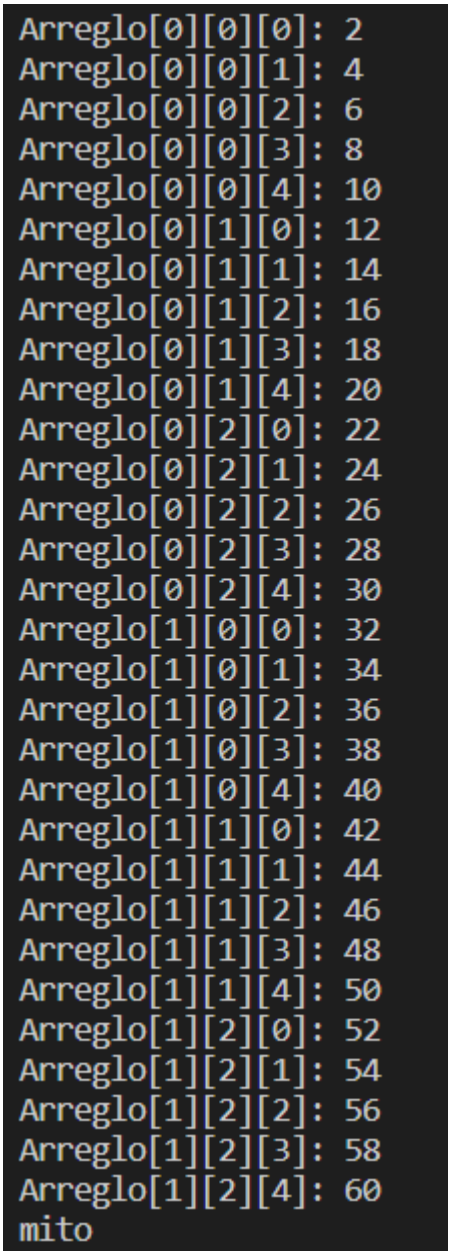
Línea 15: No hay un arreglo que se llame “a”, el arreglo declarado arriba es “a1”.

Línea 18: Debido a que no se guardó correctamente la cadena en el arreglo “letra”, se imprimirá lo que hubiera en ese espacio de memoria antes de la ejecución del programa. Debido al “%d”, no se imprimirá el carácter indicado en la cadena de la línea 6.

Línea 19: Debido a que no se guardó correctamente la cadena en el arreglo “letra”, se imprimirá lo que hubiera en ese espacio de memoria antes de la ejecución del programa. Debido al “%d”, no se imprimirá el carácter indicado en la cadena de la línea 6.

Línea 20: Debido a que no se guardó correctamente la cadena en el arreglo “letra”, se imprimirá lo que hubiera en ese espacio de memoria antes de la ejecución del programa. Debido al “%d”, no se imprimirá el carácter indicado en la cadena de la línea 6.

Línea 21: Debido al “%d”, no se imprimirá el carácter indicado en la cadena de la línea 6.



Este programa hace uso de arreglos para almacenar varios elementos del mismo tipo de dato, y un ciclos for para recorrer cada uno de estos elementos. A su vez, se hace uso un arreglo de caracteres para guardar unas palabras, y después mediante los índices sólo tomar algunas letras.

Al momento de corregir el código de este ejercicio no se presentaron dificultades significativas, ya que dentro de lo que cabe, no era un código demasiado largo y complicado. Bastó con leer algunas veces el código, y tratar de compilarlo otras tantas para dar con todos los errores.

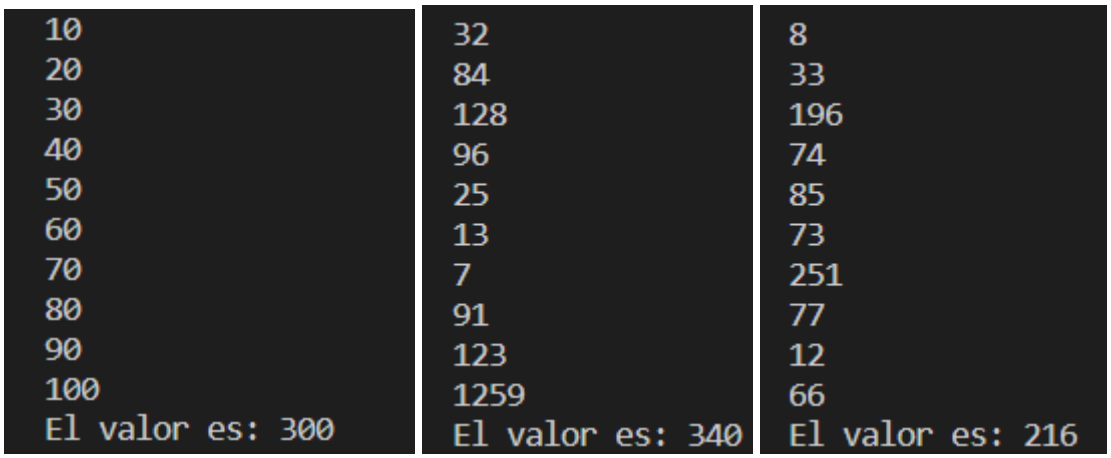
Este programa ayuda a visualizar la sintaxis de un arreglo multidimensional, así como el uso posible a un arreglo de caracteres.

Ejercicio 2

La salida son los valores del arreglo “lista”, y un mensaje donde indica cual es la suma de todos los valores del arreglo.

El programa declara el arreglo de 10 elementos con los enteros dados en el código, también crea dos variables de tipo entero, una llamada “valor”(para sumar los elementos del arreglo), y una llamada “i” (se usará como contador). Seguido de eso mediante un do{}while(), se imprimen los valores del arreglo y se van sumando conforme se imprimen a la variable valor, aumentado en 1 con cada elemento, hasta iterar 10 veces, lo cual recorrería todo el arreglo. Finalmente se imprime “El valor es: 550”, ya que 550 es la variable “valor” que fue sumando todos los elementos del arreglo.

Al seguir con el ejercicio, además del ciclo for, para recorrer el arreglo, es necesario añadir un if. Este último para que mientras se recorre el arreglo, solo se sumen aquellos elementos divisibles entre 4.



Ejercicio 3

Este programa se basa en realizar distintos procesos a un arreglo de 8 elementos dados por el usuario. El programa permite añadir o editar el arreglo, mostrar la suma de todos los elementos, la multiplicación de todos los elementos, la suma de todos los elementos divisibles entre 4, y finalmente multiplicar todos los elementos de un arreglo por un número dado por el usuario.

Para este código además del arreglo de 8 elementos al que se le aplican todas las operaciones, se ocupan varios ciclos de repetición for, en cada una de las cosas que se le pueden hacer al arreglo, se involucra al menos un ciclo for para recorrer el arreglo.

Este programa no presentó un reto demasiado complicado. La mayor adversidad a la que me enfrente durante su realización, que el programa se repitiera hasta que el usuario quisiera, pero tan solo metí todo en un ciclo while, y modificar lo hacía que se cumpliera la condición del ciclo while cuando el usuario quisiera terminar.

La realización de este ejercicio me hizo pensar en algunas de las operaciones que se pueden realizar con o a un arreglo.

```
Seleccione la operaci├│n a realizar al arreglo de 8 elementos.
1)Ingresar o modificar elementos del arreglo.
2)Mostrar el resultado de la suma de los elementos.
3)Mostrar el resultado de la multiplicacion de los elementos.
4)Realizar la suma de los elementos divisibles entre 4.
5)Multiplicar cada elemento del arreglo por un n├│mero dado.
6)Salir
1

Arreglo actual:
Arreglo[0]= 0
Arreglo[1]= 0
Arreglo[2]= 0
Arreglo[3]= 0
Arreglo[4]= 0
Arreglo[5]= 0
Arreglo[6]= 0
Arreglo[7]= 0
Indique con que modificar los valores del arreglo.
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8

Arreglo actual:
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8
Para hacer algo m├ís ingrese 1, de lo contrario cualquier otro numero.1
Seleccione la operaci├│n a realizar al arreglo de 8 elementos.
1)Ingresar o modificar elementos del arreglo.
2)Mostrar el resultado de la suma de los elementos.
3)Mostrar el resultado de la multiplicacion de los elementos.
4)Realizar la suma de los elementos divisibles entre 4.
5)Multiplicar cada elemento del arreglo por un n├│mero dado.
6)Salir
2

Arreglo actual:
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8
```

```
Arreglo actual:
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8

La suma de todos los elementos es: 36
Arreglo actual:
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8
Para hacer algo m├ís ingrese 1, de lo contrario cualquier otro numero.1
Seleccione la operaci├│n a realizar al arreglo de 8 elementos.
1)Ingresar o modificar elementos del arreglo.
2)Mostrar el resultado de la suma de los elementos.
3)Mostrar el resultado de la multiplicacion de los elementos.
4)Realizar la suma de los elementos divisibles entre 4.
5)Multiplicar cada elemento del arreglo por un n├║mero dado.
6)Salir
3

Arreglo actual:
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8

La multiplicaci├│n de todos los elementos es: 40320
Arreglo actual:
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8
Para hacer algo m├ís ingrese 1, de lo contrario cualquier otro numero.1
Seleccione la operaci├│n a realizar al arreglo de 8 elementos.
1)Ingresar o modificar elementos del arreglo.
2)Mostrar el resultado de la suma de los elementos.
3)Mostrar el resultado de la multiplicacion de los elementos.
4)Realizar la suma de los elementos divisibles entre 4.
5)Multiplicar cada elemento del arreglo por un n├║mero dado.
6)Salir
4
```

```
Arreglo actual:
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8

La suma de todos los elementos divisibles entre 4 es: 12
Arreglo actual:
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8
Para hacer algo m├ís ingrese 1, de lo contrario cualquier otro numero.1
Seleccione la operaci├│n a realizar al arreglo de 8 elementos.
1)Ingresar o modificar elementos del arreglo.
2)Mostrar el resultado de la suma de los elementos.
3)Mostrar el resultado de la multiplicacion de los elementos.
4)Realizar la suma de los elementos divisibles entre 4.
5)Multiplicar cada elemento del arreglo por un n├│mero dado.
6)Salir
5

Arreglo actual:
Arreglo[0]= 1
Arreglo[1]= 2
Arreglo[2]= 3
Arreglo[3]= 4
Arreglo[4]= 5
Arreglo[5]= 6
Arreglo[6]= 7
Arreglo[7]= 8
Ingrese por cuanto quiere multiplicar los elementos del arreglo: 2

Arreglo actual:
Arreglo[0]= 2
Arreglo[1]= 4
Arreglo[2]= 6
Arreglo[3]= 8
Arreglo[4]= 10
Arreglo[5]= 12
Arreglo[6]= 14
Arreglo[7]= 16
Para hacer algo m├ís ingrese 1, de lo contrario cualquier otro numero.2
```

Ejercicio 4

El programa de este ejercicio le pide al usuario llenar un arreglo de 10 elementos con “0” o “1”, como si fuera un número binario; para después transformarlo a decimal, dando la opción de binario puro, punto fijo, o complemento a2.

El programa ocupa un arreglo para almacenar el número binario del usuario, y otro para guardar el complemento a2, y así no modificar el número del usuario por si quiere hacer alguna otra operación. En caso del complemento a2 fue el más complicado de realizar, ya que se tuvo que primero recorrer el arreglo del usuario de derecha a izquierda(el número binario se guardó de izquierda a derecha), pasando los elemento tal cual estan al arreglo del complemento, igual de derecha a izquierda; hasta encontrar un 1, el cual también se pasaría al arreglo del complemento.

```
for(i = 9; i >= 0; i--){
    if(arr[i] == 1){
        complemento[i] = arr[i];
        break;
    } else {
        if(arr[i] == 0){
            complemento[i] = 0;
        } else {
            complemento[i] = 1;
        }
    }
```

Después tan solo quedó seguir recorriendo el arreglo, pero en lugar de pasar directamente lo que estaba en el arreglo del usuario al arreglo del complemento, se pasaba lo contrario; si había un “0” en el arreglo, se pasaba un “1” y viceversa. Una vez que se tenía el arreglo del complemento, se ocupaba como parámetro en la función para pasar a decimal por binario puro.

Para el binario puro, y el punto fijo se siguió la estrategia enseñada por el profesor. Apoyándonos en el ciclo for, para que respecto al índice del arreglo, se usará como potencia de 2, y dependiendo si el bit era “0” o “1”, se ignoraba o se sumaba al resultado respectivamente. En el caso del punto fijo, se tomaban 1 bit para el signo, 4 para la parte entera, y 5 para la mantisa; esta última funcionaba de manera similar a la parte entera, pero con potencias negativas, que iban en aumento mientras mayor fuera el índice.

```
Introduzca los bits del numero
Bit 1: 1
Bit 2: 0
Bit 3: 1
Bit 4: 1
Bit 5: 0
Bit 6: 0
Bit 7: 1
Bit 8: 1
Bit 9: 0
Bit 10: 1
Para continuar cambiar el arreglo introduzca 1.2

Elija la como transformar el arreglo a decimal.
 1. Binario Puro
2. Punto fijo(5 bit entero, y 5 bit decimal)
3. Representaci├│n complemento a2.
4. Salir
1
Numero= -205
Para continuar cambiar el arreglo introduzca 1.2

Elija la como transformar el arreglo a decimal.
 1. Binario Puro
2. Punto fijo(5 bit entero, y 5 bit decimal)
3. Representaci├│n complemento a2.
4. Salir
2
Numero= -6.406250
Para continuar cambiar el arreglo introduzca 1.2

Elija la como transformar el arreglo a decimal.
 1. Binario Puro
2. Punto fijo(5 bit entero, y 5 bit decimal)
3. Representaci├│n complemento a2.
4. Salir
3
El complemento a 2 en binario es:0 1 0 0 1 1 0 0 1 1
El complenemto a2 en decimal es: 307
Para continuar cambiar el arreglo introduzca 1.2

Elija la como transformar el arreglo a decimal.
 1. Binario Puro
2. Punto fijo(5 bit entero, y 5 bit decimal)
3. Representaci├│n complemento a2.
4. Salir
4
```

Conclusiones

Al haber concluido con todos los ejercicios respectivos de la práctica, puedo afirmar que los objetivos de la misma se cumplieron, ya que los ejercicios nos hicieron analizar cómo implementar correctamente arreglos, así como darles distintas aplicaciones. A lo largo de los ejercicios se mostraron y se nos dio la oportunidad de probar bastantes aspectos de los arreglos, empezando por su sintaxis y errores comunes en el manejo de sus índices en la aplicación, así como la manera de recorrer un arreglo, las posibilidades de las operaciones que se le pueden realizar, hasta algunas aplicaciones que se les pueden dar.

Por lo antes mencionado considero que los ejercicios son apropiados para aprender arreglos de ellos, ya que estas cubren varios aspectos importantes para el uso correcto de los arreglos. Esta es una práctica de una importancia considerable, dado que los arreglos nos ofrecen una gran herramienta para organizar, manipular y almacenar varios datos del mismo tipo, añadiendo que no es un elemento difícil de usar o implementar.

La única modificación que propondría sería referente al ejercicio 2, ya que lo que se solicita en el ejercicio es relativamente simple, y más adelante se incluye en el ejercicio siguiente. Pensaría en tal vez añadir algún otro parámetro a considerar en el ejercicio, como que se tuviera que tener en cuenta el sumar los múltiplos de 4, pero no los de 6; no sería un gran cambio, y diferenciaría un poco los ejercicios, además de cumplir con el propósito de presentar esta posibilidad al momentos de recorrer el arreglo.