	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:	Edgar Tista García
Asignatura:	Estructura de Datos y Algoritmos I
Grupo:	1
No de Práctica(s):	Práctica #1
Integrante(s):	Santiago Díaz Edwin Jaret
No. de Equipo de cómputo empleado:	Trabajo en casa
No. de lista de la	42
Semestre:	2021-2
Fecha de entrega:	12-Marzo-2021
Observaciones:	

CALIFICACIÓN: _____

APLICACIONES DE ARREGLOS

Objetivos

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

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

Desarrollo

Ejercicio de Guía

El programa recuperado del manual de prácticas realiza una escítala espartana con un mensaje que ingresa el usuario y con 2 opciones disponibles, cifrar el mensaje o descifrarla. El programa funciona de la siguiente manera:

1. Imprime el menú.
2. Se solicita qué opción desea realizar.
3. Si se desea cifrar un mensaje:
    a. Se le solicitará el usuario que ingrese el valor de los renglones y columnas, esto es para guardar un espacio para cifrar la palabra.
    b. Se ingresa la palabra.
    c. El programa regresa el mensaje cifrado
4. Si se desea descifrar un mensaje:
    a. Se le solicitara al usuario que ingrese el valor de los renglones y columnas, esto es para guardar un espacio para descifrar la palabra.
    b. Se ingresa la palabra.
    c. El programa regresa el mensaje descifrado.

Columnas

Renglones


```
       ***Escitala ESPARTANA***
Que desea realizar?
1) Crear mensaje cifrado.
2) Descifrar mensaje.
3) Salir.
1
Ingrese el tamano de la Iscitala:

Renglones:3
Columnas:2
Escriba el texto a cifrar:
Edwin
El texto en la tira queda de la siguiente manera:
Ewndi

       ***Escitala ESPARTANA***
Que desea realizar?
1) Crear mensaje cifrado.
2) Descifrar mensaje.
3) Salir.
2
Ingresar el tamano de la escitala:

Renglones:3
Columnas:2
Escriba el texto a descifrar:
Ewndi
El texto descifrado es:
Edwin
       ***Escitala ESPARTANA***
Que desea realizar?
1) Crear mensaje cifrado.
2) Descifrar mensaje.
3) Salir.
3
```

Para cifrar un mensaje, el programa crea un arreglo con datos ingresados de los espacios de renglones y columnas. El mensaje debe de ser del tamaño de la multiplicación de renglones por columnas.

Asigna cada letra a un espacio del arreglo.

E	d
w	i
n	

El mensaje cifrado es el valor de columna por columna, por lo tanto, este mensaje quedaría así:

Ewndi

E	d
w	i
n	

Para descifrar el mensaje, se ingresa el valor de renglones, de columnas y el mensaje. El programa guarda los valores de la misma manera que en el proceso anterior y el mensaje es el valor por renglones.

E	d
w	i
n	

Ejercicio 1

El funcionamiento del programa es imprimir los valores de un arreglo de 3 dimensiones utilizando 3 “ciclos for”. El arreglo tenía por dimensión los valores de [2][3][5].

```
Arreglo[0][0][0]: 2
Arreglo[0][0][1]: 4
Arreglo[0][0][2]: 6
Arreglo[0][0][3]: 8
Arreglo[0][0][4]: 10
Arreglo[0][1][0]: 12
Arreglo[0][1][1]: 14
Arreglo[0][1][2]: 16
Arreglo[0][1][3]: 18
Arreglo[0][1][4]: 20
Arreglo[0][2][0]: 22
Arreglo[0][2][1]: 24
Arreglo[0][2][2]: 26
Arreglo[0][2][3]: 28
Arreglo[0][2][4]: 30
Arreglo[1][0][0]: 32
Arreglo[1][0][1]: 34
Arreglo[1][0][2]: 36
Arreglo[1][0][3]: 38
Arreglo[1][0][4]: 40
Arreglo[1][1][0]: 42
Arreglo[1][1][1]: 44
Arreglo[1][1][2]: 46
Arreglo[1][1][3]: 48
Arreglo[1][1][4]: 50
Arreglo[1][2][0]: 52
Arreglo[1][2][1]: 54
Arreglo[1][2][2]: 56
Arreglo[1][2][3]: 58
Arreglo[1][2][4]: 60
```

El programa tenía un error en 2 “ciclos for”, en el primero y en el tercero.

```
for (i=0;i<5;i++){
    for(j=0;j<3;j++){
        for(k=0;k<2;k++){
```

Ya que el primer “ciclo for” debe de ser el el valor de la dimensión más externa, el cual en este caso es el 2. En el tercer “ciclo for” pertenece a la dimensión más interna del arreglo, el cual en este caso es el 5.

El Código corregido queda de la siguiente manera.

```
for (i = 0; i < 2; i++)
{
    for (j = 0; j < 3; j++)
    {
        for (k = 0; k < 5; k++)
        {
```

Ejercicio 2

El programa trabaja sobre un arreglo de 8 elementos.

Al iniciar el programa se imprime el menú en donde se encuentran 5 opciones a utilizar y trabajar sobre el arreglo de 8 elementos.

```
** Menú **

1) Ingresar o modificar elementos del arreglo
2) Mostrar la Suma de los elementos
3) Realizar la multiplicación de los elementos
4) Realizar la suma sólo de los elementos divisibles entre 3
5) Multiplicar por 3 cada elemento del arreglo
1
Ingrese el valor del elemento [1]1
Ingrese el valor del elemento [2]2
Ingrese el valor del elemento [3]3
Ingrese el valor del elemento [4]4
Ingrese el valor del elemento [5]5
Ingrese el valor del elemento [6]6
Ingrese el valor del elemento [7]7
Ingrese el valor del elemento [8]8

El array Ingresado es [1, 2, 3, 4, 5, 6, 7, 8]
```

En la **opción 1**, el programa solicitará los valores para cada espacio, para esto utiliza un “ciclo for” y se guarda en una variable de arreglo. Además, esta variable se guarda de manera global para que pueda ser utilizada por todas las funciones.

Al final imprime el arreglo ingresado por el usuario.

En la **opción 2**, utiliza un “ciclo for”, recorriendo todo el arreglo, y el valor de cada elemento será sumado a una variable.

Después imprimirá el arreglo con el que estamos trabajando.

```
1) Ingresar o modificar elementos del arreglo
2) Mostrar la Suma de los elementos
3) Realizar la multiplicación de los elementos
4) Realizar la suma sólo de los elementos divisibles entre 3
5) Multiplicar por 3 cada elemento del arreglo
2
La suma de los elementos del arreglo da: 72

El array original es [1, 2, 3, 4, 5, 6, 7, 8]
```

```
Suma = 0 (Inicia en 0)
Suma = suma + arr[0]   // suma + 1
Suma = suma + arr[1] … // suma + 2
```

Arreglo	**1 [0]**	**2 [1]**	**3 [2]**	**4 [3]**	**5 [4]**	**6 [5]**	**7 [6]**	**8 [7]**

En la **opción 3**, se utiliza un “ciclo for”, para recorrer todo el arreglo. El valor de cada elemento será multiplicado por el valor de una variable iniciada en 1, y este resultado será guardado en la misma variable.

```
1) Ingresar o modificar elementos del arreglo
2) Mostrar la Suma de los elementos
3) Realizar la multiplicación de los elementos
4) Realizar la suma sólo de los elementos divisibles entre 3
5) Multiplicar por 3 cada elemento del arreglo
3
La multiplicación del array es: 40320

El array original es [1, 2, 3, 4, 5, 6, 7, 8]
```

```
Suma = 1 (inicia en 1)
Suma = suma * arr[0]  // suma * 1
Suma = suma * arr[1] … // suma * 2
```

Arreglo	**1 [0]**	**2 [1]**	**3 [2]**	**4 [3]**	**5 [4]**	**6 [5]**	**7 [6]**	**8 [7]**

En la **opción 4** ocupamos un “ciclo for”, para recorrer cada elemento del arreglo. Para cada valor del elemento se dividirá entre 3, y si este da de residuo 0, el número de ese elemento es divisible entre 3.

```
1) Ingresar o modificar elementos del arreglo
2) Mostrar la Suma de los elementos
3) Realizar la multiplicación de los elementos
4) Realizar la suma sólo de los elementos divisibles entre 3
5) Multiplicar por 3 cada elemento del arreglo
4
Los suma de los elementos divisibles entre 3 da cómo resultado: 9

El array original es [1, 2, 3, 4, 5, 6, 7, 8]
```

Aquellos que fueron divisibles entre 3 serán sumados entre si y guardados en una variable.

```
cicatrices[en]% 3 == 0
        Suma = 0
        Suma = suma + Arr[i]
```

```
   1
3⌋ 3
   0
```

```
1) Ingresar o modificar elementos del arreglo
2) Mostrar la Suma de los elementos
3) Realizar la multiplicación de los elementos
4) Realizar la suma sólo de los elementos divisibles entre 3
5) Multiplicar por 3 cada elemento del arreglo
5

El arreglo multiplicado por 3 queda de la siguiente manera: [3, 6, 9, 12, 15, 18, 21, 24]
El array original es [1, 2, 3, 4, 5, 6, 7, 8]
```

En la **opción 5** se utiliza un "ciclo for", pasando por cada valor del arreglo y multiplicarlo por 3, este resultado se guarda en otro arreglo.

Arreglo	1 * 3 [0]	2*3 [1]	3*3 [2]	4*3 [3]	5*3 [4]	6*3 [5]	7*3 [6]	8*3 [7]

Arreglo2	3 [0]	6 [1]	9 [2]	12 [3]	15 [4]	18 [5]	21 [6]	24 [7]

Ejercicio 3

El programa registra un arreglo de 9 espacios, el cual solo puede ser ingresados los valores de 0 y 1 por el usuario, estos números son bits.

Se registra a través de un "ciclo for".

Una vez ingresados los valores, se evalúa con un "ciclo for" recorriendo cada elemento [i] del arreglo, y con una "sentencia if" para que solo se acepte los números 0 o 1, de lo contrario, se le asigna al valor de i como "-1" para que inicia desde cero el "ciclo for".

Después, ya imprime el menú a con las opciones disponibles.

En la **primera opción** convierte los 9 bits a binario puro.

Se utiliza un "ciclo for" que inicia desde el valor 0 hasta el valor 9, esto representa al valor del espacio del arreglo ([0], [1], [2]…).

Si en el elemento hay un valor de 1, se le restará al 8 el valor del espacio del elemento.

```
** Ingrese los valores de 0 o 1 en el lugar [0] **  1
** Ingrese los valores de 0 o 1 en el lugar [1] **  0
** Ingrese los valores de 0 o 1 en el lugar [2] **  1
** Ingrese los valores de 0 o 1 en el lugar [3] **  0
** Ingrese los valores de 0 o 1 en el lugar [4] **  0
** Ingrese los valores de 0 o 1 en el lugar [5] **  0
** Ingrese los valores de 0 o 1 en el lugar [6] **  1
** Ingrese los valores de 0 o 1 en el lugar [7] **  1
** Ingrese los valores de 0 o 1 en el lugar [8] **  1

1) Binario Puro
2) Punto fijo (6 bit entero y 3 bit decimal)
3) Representación de complemeto a 2
1
El código Binario representado en número da: 327
```

El resultado será el valor del exponente a elevar al 2.

El resultado de elevar a un número será sumado a una variable.

Ejemplo:

0	1	0	0	0	0	0	0	0
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Si se ingresa 1 en el espacio 1, el programa le restará al 8 el valor del espacio (1) y el resultado que es 7, será el exponente por elevar al número 2.

```
Exponente = 8 – 1 // Exponente = 7
Cuadrado = pow(2, exponente)
Suma += Cuadrado
```

El resultado de elevar el 2, se sumará a una variable.

En la **segunda opció**n, el programa utiliza “ciclo for”, desde el 0 hasta el 9 (sin incluir al 9).

El “ciclo for” trabaja de la misma manera que en la opción 1, la diferencia es el número al que se le resta, esto es, cambiar el 8 por el 5, esto genera exponentes negativos para obtener los decimales.

```
** Ingrese los valores de 0 o 1 en el lugar [0] **  1
** Ingrese los valores de 0 o 1 en el lugar [1] **  0
** Ingrese los valores de 0 o 1 en el lugar [2] **  1
** Ingrese los valores de 0 o 1 en el lugar [3] **  0
** Ingrese los valores de 0 o 1 en el lugar [4] **  0
** Ingrese los valores de 0 o 1 en el lugar [5] **  0
** Ingrese los valores de 0 o 1 en el lugar [6] **  1
** Ingrese los valores de 0 o 1 en el lugar [7] **  1
** Ingrese los valores de 0 o 1 en el lugar [8] **  1

1) Binario Puro
2) Punto fijo (6 bit entero y 3 bit decimal)
3) Representación de complemeto a 2
2
El número es: 40.875
```

Ejemplo:

El valor del espacio es 1 y 6, este se le restará al 5, dando como valor 4 y -1 (valor del exponente).

0	1	0	0	0	0	1	0	0
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}

```
Exponente = 5 – 1                  // Exponente = 5 - 6
Cuadrado = pow(2, exponente)       // Cuadrado = pow (2, exponente)
Suma += Cuadrado                   // Suma += cuadrado
```

Se crea una variable en donde se guarda y suma el valor de elevar el 2 a exponente número.

La **tercera opción** del programa convierte los 9 bits a complemento A 2, el cual cambia los valores de 0 por 1 y de 1 por 0, al final, al último elemento [8] se le suma 1.

Se utiliza un “ciclo for” del 0 al 9 (sin incluir al 9), una “sentencia if”, para validar si es 0 o es 1.

Si es en el espacio [i], se le cambia el valor a 0, si no, el espacio [i] equivale 0 y se le cambia el valor a 1.

```
** Ingrese los valores de 0 o 1 en el lugar [0] **  1
** Ingrese los valores de 0 o 1 en el lugar [1] **  0
** Ingrese los valores de 0 o 1 en el lugar [2] **  1
** Ingrese los valores de 0 o 1 en el lugar [3] **  0
** Ingrese los valores de 0 o 1 en el lugar [4] **  0
** Ingrese los valores de 0 o 1 en el lugar [5] **  0
** Ingrese los valores de 0 o 1 en el lugar [6] **  1
** Ingrese los valores de 0 o 1 en el lugar [7] **  1
** Ingrese los valores de 0 o 1 en el lugar [8] **  1

1) Binario Puro
2) Punto fijo (6 bit entero y 3 bit decimal)
3) Representación de complemeto a 2
3

El complemento a 2 es: 010111001
```

Después, se le suma 1 al valor del elemento [8], pero ¿Qué pasa si el elemento [8] tiene 1? Si sucede eso, en vez de que el resultado sea 2 será 0, porque en bits no existe otro número distinto a 0 y 1, por lo tanto, lo convierte a 0, y le pasa el valor de 1 hacia el elemento menor [7]. Si el ya tiene 1, se convierte 0 y le pasa el valor de 1 al menor, así sucesivamente hasta que encuentre un elemento con 0.

¿qué pasa si no lo encuentra, si el usuario ingresa todo 0? El programa lo detecta y arroja un mensaje, diciendo que no es posible convertir esos 9 bits a Complemento A 2.

1 [0]	0 [1]	1 [2]	0 [3]	1 [4]	1 [5]	0 [6]	0 [7]	0 [8]

Bits ingresados

0 [0]	1 [1]	0 [2]	1 [3]	0 [4]	0 [5]	1 [6]	1 [7]	**1** + **1** [8]

Cambiar de 0 por 1 y 1 por 1
Sumar 1 al elemento [8]

1 [0]	0 [1]	1 [2]	0 [3]	0 [4]	1 [5]	0 [6]	0 [7]	0 [8]

Resultado de complemento A2

Conclusiones

Los arreglos se usan en todas partes por su fácil uso y manejo, la manera de trabajar con dimensiones logra ahorrar muchas líneas de código, variables y datos, pueden ser implementadas de múltiples maneras y, además, pueden guarda y trabaja con cualquier tipo de dato, abriendo un sinfín de posibilidades para implementar algoritmos.

Los arreglos son de suma importancia porque se pueden encontrar e implementar en todos lados.

Los objetivos de la práctica se cumplieron por implementar arreglos de 1 dimensión, de 2 y de 3 dimensiones. 3 programas fueron realizados por el alumno y esto provocó que el alumno fuera razonando, generando e implementando maneras de trabajar con arreglos.

La práctica es laboriosa pero adecuada, genera al alumno razonamientos y que vaya más allá de sus propios conocimientos haciéndolo investigar, leer libros, ver videos, genera preguntas y maneras más cortas para obtener un gran programa.