	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: TISTA GARCÍA EDGAR

Asignatura: ESTRUCTURA DE DATOS Y ALGORITMOS I

Grupo: 1

No de Práctica(s): 1] APLICACIONES DE ARREGLOS

Integrante(s): MURRIETA VILLEGAS ALFONSO

Semestre: 2018 - 2

Fecha de entrega: 19 DE FEBRERO DE 2018

Observaciones:

CALIFICACIÓN: _____

APLIACIONES DE ARREGLOS

1] Objetivo de la práctica

- Utilizar arreglos unidimensionales y multidimensionales para dar solución a problemas computacionales.

2] Desarrollo

Introducción:

Un arreglo es un conjunto de variables o datos que se caracterizan por ser del mismo tipo, estos pueden ser unidimensionales o multidimensionales.

Los arreglos pueden también clasificarse como contiguos o ligados, se les conoce como **contiguos** aquellos que se crean desde el inicio del programa y permanecen estáticos durante toda la ejecución del mismo, es decir, no se pueden redimensionar.

Por otro lado, un **arreglo ligado** es aquel que se declara en tiempo de ejecución y bajo demanda, por lo tanto, es posible incrementar su tamaño durante la ejecución del programa,

2.1] Análisis de los problemas (Propuestos)

En el presente programa podemos destacar como mediante un algoritmo básico donde a través de arreglos y funciones podemos codificar un mensaje, así como lo hacían los griegos (En particular los espartanos) en el pasado con la famosa escitala espartana.

NOTAS:

Lo primero que podemos ver es que el código está de manera general partido en 3 funciones:

1) La función **main** se encarga de dar el menú al usuario para posteriormente este pueda escoger si cifrar o descifrar un mensaje

2) La función **crearMensaje** es llamada desde el *case 1* de la función **main**, esta se encarga de determinar el tamaño del mensaje mediante los datos ingresados por el usuario (Cantidad de renglones y columnas), de esta forma el usuario crea un arreglo ligado para así posteriormente pasar por otro arreglo el cual cifrará los datos.

3) La función **decifrarMensaje** se encarga precisamente de descifrar el mensaje, sin embargo, para poder descifrar el mensaje el usuario debe saber cuántos renglones y columnas tiene el mensaje cifrado además del texto ya cifrado, lo único que se encarga esta función es pasar el arreglo de caracteres por un arreglo que simplemente ordena los datos para que estos sean comprensibles.

A continuación, se muestran capturas de pantalla del programa propuesto (escitala espartana) donde primero vemos la parte del programa donde se está cifrando un mensaje y posteriormente la parte donde se descifra.

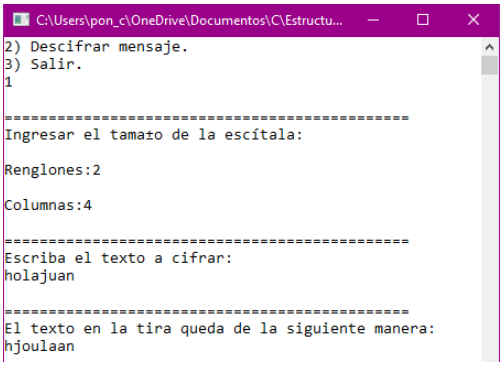


Imagen 1: Cifrado de mensaje

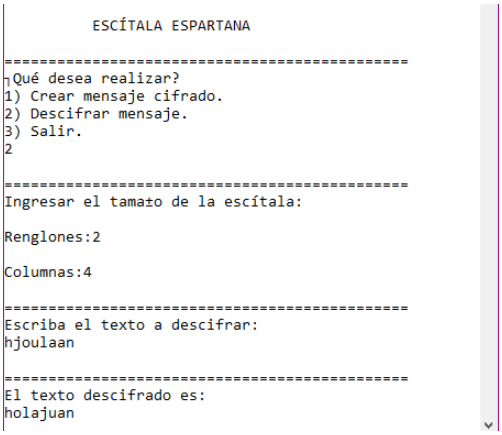
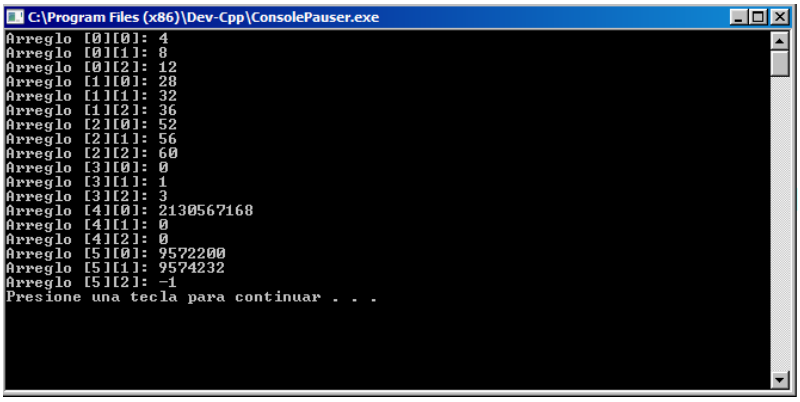


Imagen 2: Descifrado de mensaje

2.2] Análisis de los problemas (Práctica)

EJERCICIO 1

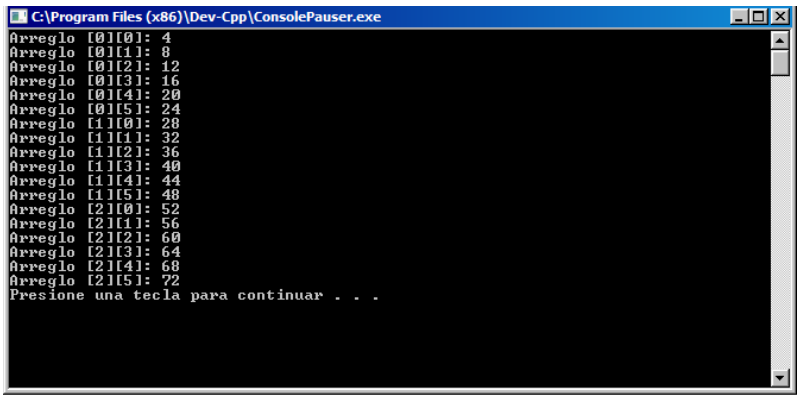
a) El primer ejercicio es una aplicación de un arreglo ya declarado donde la salida marca un pequeño error puesto al imprimirse se muestra de la siguiente forma (Se omiten algunos datos declarados en el código ya escrito)



La manera de solucionar este simple error es ir directamente al apartado donde se limitan los ciclos for.

```
for (i=0;i<3;i++){
    for(j=0;j<6;j++){
        printf("Arreglo [%d][%d]: %d  \n",i,j,arr[i][j] );
    }
```

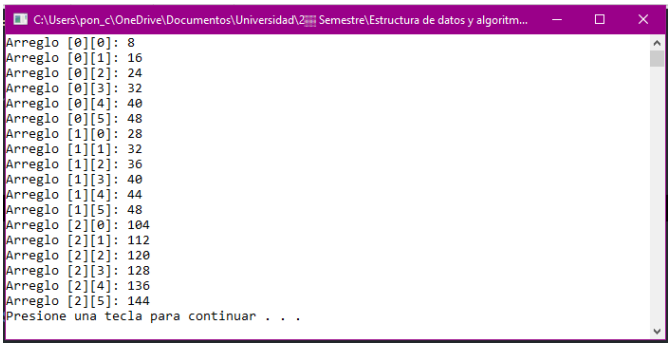
De esta forma ya imprimimos el arreglo de manera correcta



b) En la segunda parte del problema lo que hice fue duplicar los valores de los renglo nes impares que en este caso fueron el renglón 0 y 2, pese existen varias manera de abordar este problema, decidí condicionar de la siguiente forma, donde el primer *if* sirve para identificar los renglones a los que se les duplicará el valor, mientras que el *else* simplemente sirve para imprimir de manera normal el renglón que no corresponde a impar.

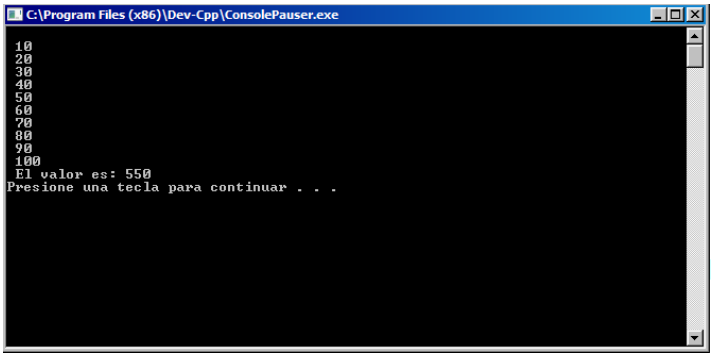
```
if (i % 2 == 0)
//if(i== 0 || i == 2) version alternativa
{
    for(j=0;j<6;j++){

        arr[i][j]= arr[i][j]*2;
        printf("Arreglo [%d][%d]: %d  \n",i,j,arr[i][j] );
    }
}
else
{
    for(j=0;j<6;j++){
        printf("Arreglo [%d][%d]: %d  \n",i,j,arr[i][j] );
        //Esta imprime los renglones pares (o sea el 1 para la pc)
    }
}
```

A continuación, se muestra la pantalla de ejecución del programa donde podemos notar como solo los valores del renglón 0 y 2 son los que duplicaron su valor.



EJERCICIO 2

a) El programa del ejercicio 2 mediante el uso de un arreglo bidimensional ya declarado lo que hace es imprimir todos los valores guardados en el arreglo y posteriormente imprimir la suma de todos los valores.



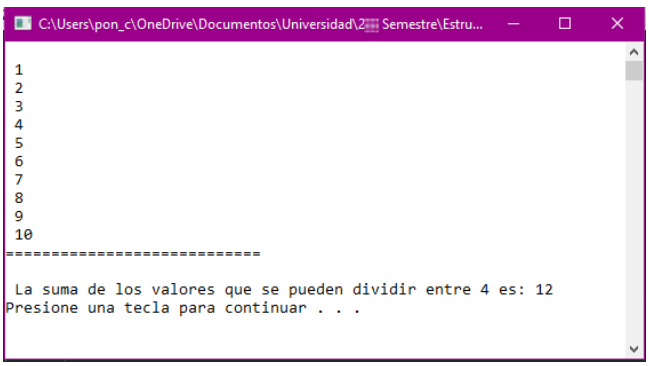
b) La segunda parte de este ejercicio lo que se plantea es que haga lo mismo solamente que esta vez solo se sumaran los valores que sean divisibles entre 4.

1) Lo primero que hice fue cambiar los valores declarados del arreglo, realmente esto no era necesario puesto no afectaba en nada al código, solo los resultados que si era divisibles entre 4 iban a ser impresos, pero pese a ello aun así decidí cambiar los valores.

2) Posteriormente condicioné las impresiones del arreglo mediante un simple *if* con un módulo de 4

```
    if ( lista[i]%4 == 0 )
    {
    /*  printf("\n %d", lista[i]);  si el print lo ponemos dentro del if solo si imprimirán
     los valores que se dividen entre 4*/

        valor += lista[i];
    }
    i++;// Recuerda sacar el iterador para que de esta forma vaya sumando después del if
```

A continuación, se muestra una captura de pantalla de la salida del programa con el siguiente arreglo declarado **lista[C]= {1,2,3,4,5,6,7,8,9,10};**



NOTAS:

El ejercicio pedía que en la suma solo se sumaran los números que se pueden dividir entre 4, sin embargo, no mencionaba que no imprimiéramos todo el arreglo ya declarado, por lo tanto, agrego en el código como puedo hacer que solo se imprima los valores de los números divisibles entre 4.

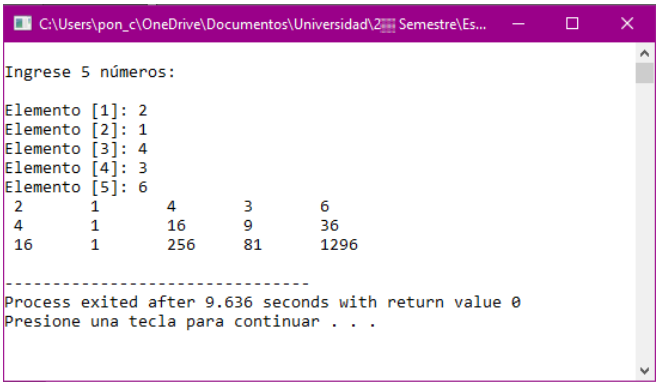
EJERCICIO 3

3) Por último, en este programa se declaró un arreglo bidimensional de 3x5 donde solamente el primer renglón almacena datos dados por el usuario para posteriormente en los siguientes renglones imprimir el cuadrado del número anterior a su columna.

    1) En la primera parte lo que hice fue anidar 2 ciclos *for* donde a través de condiciones primero está la parte de ingresar los datos dados por el usuario y posteriormente la parte que se encarga de elevar cada valor declarado al cuadrado.

    2) Nuevamente tuve que anidar dos ciclos *for* para imprimir el arreglo ya guardado, debido a que tuve algunos inconvenientes al momento de imprimir el arreglo usando el conjunto de *for* anidados que se encargaba de guardar el arreglo

A continuación, se muestra una captura de pantalla donde se muestra la salida de los valores ingresados por el usuario al cuadrado en los siguientes renglones:



3.1) ¿Qué pasaría si el usuario introduce 500 como uno de los números que guardará en el arreglo?

Cuando el usuario ingrese al programa 500 lo que pasará es que desbordará la capacidad permitida por el tipo de arreglo declarado, ya que elevar 500 al cuadrado nos daría 250000 el cual, si podría ser impreso, sin embargo, ese valor elevado al cuadrado sobrepasaría la capacidad del tipo de variable utilizada.

2.3 Relación de los ejercicios con el tema

Todos los ejercicios realizados en la presente práctica son sin duda algunos casos donde no hay manera de resolver más que con arreglos ya que si fueran abordados mediante variables tendrían que usarse una cantidad enorme además de lo poco optimizado que estaría el código con tantas variables implicadas. Los arreglos de cierta forma se pueden aplicar con el principal propósito de optimizar e incluso tener ordenadas nuestras variables.

3] Conclusiones

En la presente práctica mediante el uso de arreglos abordamos distintos tipos de problemas, mediante la abstracción y razonamiento de los problemas planteados uno como programador debe plantearse cómo se puede llevar acabo la solución del problema.

En el caso del ejercicio 1 a través de un código ya hecho debíamos corregir un pequeño error que este tenía, es ahí donde conceptos básicos de cómo se imprime un arreglo sirven para descubrir que el error estaba en los límites de los renglones y columnas, también es el caso del ejercicio 2 donde a través de la suma de todos los elementos del arreglo podíamos obtener un valor escalar del arreglo, sin embargo, el apartado b de ese ejercicio pedía que solo fueran sumados los elementos divisibles entre 4, por lo tanto, uno debía hacer uso de condiciones y operadores aritméticos para poder hacer esto posible.

Por otro parte, lamentablemente durante la clase entre los nervios debido a que estaba condicionado a un tiempo determinado y además de haber olvidado conceptos muy básicos de arreglos me fue imposible terminar los apartados b de cada ejercicio, dado la situación que se presentó el día de hoy espero y trataré de que no se vuelva a repetir, aunque para ser sincero siempre me ha gustado trabajar con tiempo y sin presión.

Para concluir también quiero mencionar que sin duda esta práctica me hizo poder ver algunos ejemplo y ejercicios de cómo se aplican y usan los arreglos, sin duda algo necesario para posteriores temas.

4] Referencias

1) Curso de Programación C7 C++. Javier Ceballos, tercera edición. USA, Ra – Ma. Educación 2007

2) El lenguaje de programación C. Brian W. Kernighan, Dennis M. Ritchie, segunda edición, USA, Pearson Educación 1991.