Carátula para entrega de prácticas	
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Alejandro Esteban Pimentel Alarcon

Asignatura: Fundamentos de programación

Grupo: 3

No de Práctica(s): 10

Integrante(s): Rivera Sosa Arlethe
Yuan Xiaojing

No. de Equipo de cómputo empleado:

No. de Lista o Brigada: 317083033
317693612

Semestre: 2020-1

Fecha de entrega: 28/octubre/19

Observaciones:

CALIFICACIÓN: _____

Arreglos

Los arreglos permiten almacenar vectores y matrices. Los arreglos unidimensionales sirven para manejar vectores y los arreglos bidimensionales para matrices. Sin embargo, las matrices también se pueden almacenar mediante arreglos unidimensionales y por medio de apuntadores a apuntadores, temas que se verán en el capítulo siguiente. La palabra unidimensional no indica que se trata de vectores en espacios de dimensión uno; indica que su manejo se hace mediante un subíndice. El manejo de los arreglos bidimensionales se hace mediante dos subíndices.

- Un arreglo unidimensional: es un tipo de datos estructurado que está formado por una colección finita y ordenada de datos del mismo tipo. Es la estructura natural para modelar listas de elementos iguales. Los datos que se guarden en los arreglos todos deben ser del mismo tipo. El tipo de acceso a los arreglos unidimensionales es el acceso directo, es decir, podemos acceder a cualquier elemento del arreglo sin tener que consultar a elementos anteriores o posteriores, esto mediante el uso de un índice para cada elemento del arreglo que nos da su posición relativa. Para implementar arreglos unidimensionales se debe reservar espacio en memoria. Los arreglos nos permiten hacer un conjunto de operaciones para manipular los datos guardados en ellos, estas operaciones son: ordenar, buscar, insertar, eliminar, modificar entre otras.
- Arreglos multidimensionales: un arreglo multidimensional es simplemente una extensión de un arreglo unidimensional. Más que almacenar una sola lista de elementos, piense en un arreglo multidimensional como el almacenamiento de múltiples listas de elementos. Por ejemplo, un arreglo bidimensional almacena listas en un formato de tabla de dos dimensiones de filas y columnas, en donde cada fila es una lista. Las filas proporcionan la dimensión vertical del arreglo, y las columnas dan la dimensión horizontal. Un arreglo de tres dimensiones almacena listas en un formato de tres dimensiones de filas, columnas y planos, en donde cada plano es un arreglo bidimensional. Las filas proporcionan la dimensión vertical; las columnas, la dimensión horizontal; y los planos, la dimensión de profundidad del arreglo.
- Arreglos bidimensionales: tiene dos dimensiones y es un caso particular de los arreglos multidimensionales. Los arreglos bidimensionales son tablas de valores. Cada elemento de un arreglo bidimensional está simultáneamente en una fila y en una columna. En matemáticas, a los arreglos bidimensionales se les llama matrices, y son muy utilizados en problemas de Ingeniería.

Objetivo: Reconocer la importancia y utilidad de los arreglos, en la elaboración de programas que resuelvan problemas que requieran agrupar datos del mismo tipo, así como trabajar con arreglos tanto unidimensionales como multidimensionales.

Actividad1: Hacer un programa que:

- Pida al usuario un número.
- Genere un arreglo de esa longitud.
- Pida al usuario números suficientes para llenar el arreglo.
- Muestre al usuario el número menor y el mayor de dicho arreglo.

```c
#include <stdio.h>
int main(){
    int n, i;
    printf("ingresar lista\n");
    scanf("%i", &n);
    int lista [n];
    for(int i=0; i<n; i++){
        printf("ingrese n numero:");
        scanf("%i", &lista[i]);
    }
    int valor1=lista[0];
    int valor2=lista[0];
    for(int i=0; i<n; i++){
        if(lista[i]>valor1){
            valor1=lista[i];
        }
        if(lista[i]<valor2){
            valor2=lista[i];
        }
    }
    printf("el mayor es: %i\n", valor1);
    printf("el menos es: %i\n", valor2);
    return 0;
}
```

```
Familia@DESKTOP-NE4QAFJ ~
$ gcc eje1.c -o eje

Familia@DESKTOP-NE4QAFJ ~
$ ./eje
ingresar lista
1 4 9 4 8 2 7 1 4 5
ingrese n numero:el mayor es: 1
el menos es: 9
```

Hacemos un programa para que vaya leyendo los datos uno por uno y elija el más grande de esos dos, luego lo guarda y vuelve a comparar con el siguiente número y así hasta que el arreglo acabe. Asimismo, lo hace, pero con el número más pequeño.

Actividad2: Hacer un programa que:

- Pida al usuario unos dos números N y M.
- Genere dos matrices de N × M.
- Pida al usuario números suficientes para llenar ambas matrices.
- Muestre al usuario la matriz resultado de sumar las dos de entrada.

```c
#include <stdio.h>
int main(){
    int N, M;
    printf("ingresa N:"); scanf("%i", &N);
    printf("ingresa M:"); scanf("%i", &M);
    int matriz[N][M];
    printf("Matriz 1:\n");
    for(int i=0; i<N; i++){
        for(int j=0; j<M; j++){
            scanf("%i", &matriz[i][j]);
        }
    }
    int matriz2[N][M];
    printf("Matriz 2:\n");
    for(int i=0; i<N; i++){
        for(int j=0; j<M; j++){
            scanf("%i", &matriz2[i][j]);
        }
    }
    int matrizR[N][M];
    printf("Matriz R:\n");
    for(int i=0; i<N; i++){
        for(int j=0; j<M; j++){
            matrizR[i][j]=matriz[i][j]+matriz2[i][j];
        }
    }
    for(int i=0; i<N; i++){
        for(int j=0; j<M; j++){
            printf("%i\n", matrizR[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

```
Familia@DESKTOP-NE4QAFJ ~
$ ./eje
ingresa N:1 3 5
ingresa M:Matriz 1:
1 3 5
Matriz 2:
1 3 5
Matriz R:
2
6
10
```

Hacemos un programa que sume dos matrices y nos de el resultado en una tercera, se suma el primer elemento de cada una y el resultado de la suma es el primer número de la tercera matriz.

Colocamos M1= (1, 3, 5) y M2= (1, 3, 5)

Para que el resultado sea M3= (2, 6, 10)

Conclusión: los arreglos unidimensionales y multidimensionales, nos ayudan a almacenar matrices y listas, de manera que pudimos en el primer ejercicio guardamos una lista y verificamos cada número para ver cuál era el mayor y cuál el menor. Y en el segundo ejercicio que consistió en una suma de matrices. Estos arreglos hacen que al escribir nuestro programa sea más corto y más preciso que si lo hacemos paso a paso.