



## Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

# Laboratorios de computación salas A y B

*Profesor:* Marco Antonio Martínez Quintana.

*Asignatura:* Fundamentos de programación

*Grupo:* 3

*No de Práctica(s):* 11

*Integrante(s):* Moreno Razo Laura Mildred

*No. de Equipo de  
cómputo empleado:* No aplica

*No. de Lista o Brigada:* No aplica

*Semestre:* 2021-1

*Fecha de entrega:* Lunes 4 de Enero del 2021

*Observaciones:*

**CALIFICACIÓN:** \_\_\_\_\_

**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.

**Actividades:**

- Elaborar un programa en lenguaje C que emplee arreglos de una dimensión.
- Resolver un problema que requiera el uso de un arreglo de dos dimensiones, a través de un programa en lenguaje C.
- Manipular arreglos a través de índices y apuntadores.

**Introducción**

Se conoce como arreglo o array a un conjunto de datos o estructura de datos, los cuales se llegan a encontrar organizados de manera homogénea y que se encuentran ubicados en la memoria RAM (que es donde se almacenan los datos de forma temporal). Estos datos no deben de tener ningún tipo de diferencias u anomalía en sus formatos como en sus cualidades que pudieran dar lugar a problemas.

Se organizan de manera consecutiva para que estos tengan un orden predeterminado en su ejecución y posterior almacenamiento en la memoria RAM de la computadora, ya que sus operaciones son almacenadas de manera temporal. Los datos en un array son completamente flexibles y pueden ser combinados como anidados los datos que se puedan estar usando en la programación.

Los arreglos que tienen una sola dimensión se les llega a conocer como Vectores, mientras los que poseen dos dimensiones se les llama Matrices y por último aquellos arreglos que tengan sus dimensiones igual o superior a tres dimensiones se llegan a conocer como tablas multidimensionales.

## Actividad 1

Nos muestra las calificaciones de 5 alumnos, mediante un arreglo unidimensional de 5 elementos del tipo entero, además del uso de la estructura while.

```
C:\Users\mildr\Desktop\C>arreglos.exe
Lista

Calificación del alumno 1 es 10
Calificación del alumno 2 es 8
Calificación del alumno 3 es 5
Calificación del alumno 4 es 8
Calificación del alumno 5 es 7

C:\Users\mildr\Desktop\C>_
```

## Actividad 2

En el siguiente ejemplo la función es la misma, pero haciendo uso de la estructura de repetición for, la diferencia radica en que sabemos cuantas veces se va repetir el ciclo, además se minimiza una línea de código ya que desde la sintaxis del ciclo definimos el incremento, aunque no necesariamente siempre podemos usar una en vez de la otra estructura, depende de la utilidad que estemos buscando.

## Actividad 3

Se muestra el uso de los apuntadores, en este caso hacia una variable del tipo carácter.

En los primeros dos se accede al contenido de la variable mediante el apuntador, en el último no, por lo que solo nos indica hacia donde dirige ese apuntador, es decir el espacio en memoria donde se aloja la variable.

```
C:\Users\mildr\Desktop\C>apuntadores.exe
Carácter: a
Código ASCII: 97
Dirección de memoria: 6422299

C:\Users\mildr\Desktop\C>
```

## Actividad 4

Accede a las localidades de memoria de distintas variables a través de un apuntador.

Código comentado.

```
int *apEnt;
//Nos indica hacia que direccion de memoria apunta en este caso: a
apEnt = &a;

printf("a = 5, b = 10, c[10] = {5, 4, 3, 2, 1, 9, 8, 7, 6, 0}\n");
printf("apEnt = &a\n");

/*Le asigna a b el valor del contenido de la variable a la que apunta apEnt,
es decir el valor de a*/
b = *apEnt;
//Imprime el valor que fue asignado
printf("b = *apEnt \t-> b = %i\n", b);

/*Nuevamente se asigna el valor contenido en la variable hacia la que se apunta
y se incrementa en 1 */
b = *apEnt + 1;
printf("b = *apEnt + 1 \t-> b = %i\n", b);

//Se accede al valor de la variable apuntada (a) y se le asigna el valor 0
*apEnt = 0;
//Al imprimir el valor de a, aparecera 0 por lo realizado en la línea anterior
printf("*apEnt = 0 \t-> a = %i\n", a);

//Se indica que el apuntador apunta ahora hacia el primer elemento del arreglo c
apEnt = &c[0];
//Accede al contenido del primer elemento de c y lo imprime
printf("apEnt = &c[0] \t-> apEnt = %i\n", *apEnt);

return 0;
```

```
C:\Users\mildr\Desktop\C>apuntadores.exe
a = 5, b = 10, c[10] = {5, 4, 3, 2, 1, 9, 8, 7, 6, 0}
apEnt = &a
b = *apEnt      -> b = 5
b = *apEnt + 1  -> b = 6
*apEnt = 0      -> a = 0
apEnt = &c[0]   -> apEnt = 5

C:\Users\mildr\Desktop\C>
```

## Actividad 5

Mediante el uso de un apuntador y otra variable se accede al contenido de un arreglo

```
#include <stdio.h>
/*
Este programa trabaja con aritmética de apuntadores para acceder a los
valores de un arreglo.
*/
int main () {
    int arr[] = {5, 4, 3, 2, 1};
    int *apArr;
    //Se indica que el apuntador apunta al primer elemento del arreglo
    apArr = arr;

    printf("int arr[] = {5, 4, 3, 2, 1};\n");
    printf("apArr = &arr[0]\n");

    //Declara una variable (x) y le asigna el valor del contenido a donde dirige el apuntador (arr[0]=5)
    int x = *apArr;
    //Imprime el valor asignado de x
    printf("x = *apArr \t -> x = %d\n", x);

    //asigna el valor a x del contenido a donde dirige el apuntador,pero del siguiente elemento (arr[1]=4)
    x = *(apArr+1);
    printf("x = *(apArr+1) \t -> x = %d\n", x);

    //asigna el valor a x del contenido a donde dirige el apuntador,pero del elemento 2 (arr[2]=3)
    x = *(apArr+2);
    printf("x = *(apArr+1) \t -> x = %d\n", x);

    return 0;
}
```

```
C:\Users\mildr\Desktop\C>gcc apuntadores.c -o apuntadores.exe
C:\Users\mildr\Desktop\C>apuntadores.exe
int arr[] = {5, 4, 3, 2, 1};
apArr = &arr[0]
x = *apArr      -> x = 5
x = *(apArr+1)  -> x = 4
x = *(apArr+1)  -> x = 3
C:\Users\mildr\Desktop\C>
```

## Actividad 6

Al igual que en la actividad 2 , nos muestra el contenido (calificaciones) de un arreglo unidimensional mediante el ciclo for, pero en este caso accedemos a los valores del arreglo mediante el uso de un apuntador, que va siendo incrementado para mostrar cada elemento.

```
Lista
Calificación del alumno 1 es 10
Calificación del alumno 2 es 8
Calificación del alumno 3 es 5
Calificación del alumno 4 es 8
Calificación del alumno 5 es 7

C:\Users\mildr\Desktop\C>_
```

## Actividad 7

El programa nos muestra como el manejo de las cadenas de caracteres, primeramente, creamos un arreglo de tamaño 20 para almacenar los caracteres que forman la cadena, por lo que un carácter corresponde a un elemento, para después acceder a la información por medio de un ciclo for, que recorre del elemento 0 al 19.

Este programa imprimía caracteres no deseados , ya que el arreglo continuaba poniendo datos aun cuando la cadena ya había terminado, por lo que pensé que si la inicializábamos en cero, los valores sin caracteres asignados mantendrían ese valor de 0 y efectivamente se solucionó.

```
#include <stdio.h>
/*
    manejo de cadenas en lenguaje C.
*/
int main() {
    //Crea un arreglo unidimensional de tamaño 20 para almacenar caracteres
    char palabra[20]={0};
    int i;
    printf("Ingrese una palabra: ");
    scanf("%s",palabra);
    printf("La palabra ingresada es: %s\n", palabra);
    /*En el ciclo se busca acceder al contenido del arreglo, valor por valor que hayamos ingresado
    para formar la palabra, empezando desde el elemento 0 hasta el 19
    */
    for (i = 0 ; i < 20 ; i++){
        printf("%c\n", palabra[i]);
    }
    return 0;
}
```

```
C:\Users\mildr\Desktop\C>apuntadores.exe
Ingrese una palabra: mildred
La palabra ingresada es: mildred
m
i
l
d
r
e
d

C:\Users\mildr\Desktop\C>
```

### Actividad 8

Imprime mediante dos ciclos for una matriz 3x3 cuyos valores estan almacenados en un arreglo multidimensional (2 dimensiones) .

Podemos decir que en el ciclo for, al estar uno anidado en otro se realizan las combinaciones de los valores tomados por j con los de i para imprimir el valor guardado de cada posicion.

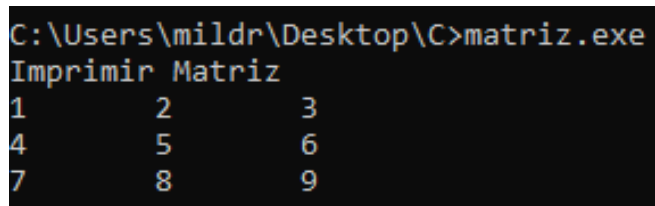
```
Imprimir Matriz
1, 2, 3,
4, 5, 6,
7, 8, 9,

C:\Users\mildr\Desktop\C>
```

## Actividad 9

Al igual que el programa anterior imprime una matriz 3x3 , pero en este caso accede a la posición por medio de un apuntador usado en un ciclo for el cual aumenta en 1 para dar la siguiente posición ,ademas hay un if anidado que nos permite saber cuándo ya se terminó de imprimir una fila para dar un salto de línea y quede de la forma que deseamos.

El programa presentaba un error al compilar, el cual solucioné indicando al apuntador apuntar al arreglo matriz en la posición 0



```
C:\Users\mildr\Desktop\C>matriz.exe
Imprimir Matriz
1      2      3
4      5      6
7      8      9
```

## Conclusión

El uso de arreglos en C es importante para el uso y acceso a conjuntos de datos de manera eficiente, ya que nos permite tener posiciones bien definidas y acceder rápidamente.

Además, los apuntadores están muy relacionados, ya que el nombre del arreglo también lo podemos definir como un apuntador constate hacia el primero elemento, por lo que es necesario tener el conocimiento de la función de estas herramientas para poder utilizarlas.

La actividades y contenido de la practica fue útil, pues me permitió entender el funcionamiento de estos conceptos ya aplicado en los programas presentados, aunque algunos me parecieron complicados, como es el caso de la actividad 7, pues ocurría una excepción que imprimía símbolos no deseados, pero gracias a los conocimientos anteriormente obtenidos lo pude solucionar.

## Bibliografía

- Escalante, A. (2020). Los tipos de arreglos en la programación. Consultado el 6 Enero 2021, de <https://tecnoinformatic.com/c-programacion/tipos-de-arreglos/>