



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

<i>Profesor:</i>	M.I. Marco Antonio Martínez Quintana
<i>Asignatura:</i>	Fundamentos de Programación
<i>Grupo:</i>	03
<i>No de Práctica(s):</i>	11
<i>Integrante(s):</i>	Teran García Rodolfo Mario
<i>No. de Equipo de cómputo empleado:</i>	No aplica
<i>No. de Lista o Brigada:</i>	
<i>Semestre:</i>	2021-1
<i>Fecha de entrega:</i>	04/01/2021
<i>Observaciones:</i>	M.I. Marco Antonio Martínez Quintana

Calificación:

Practica #11: Arreglos unidimensionales y multidimensionales

Objetivos:

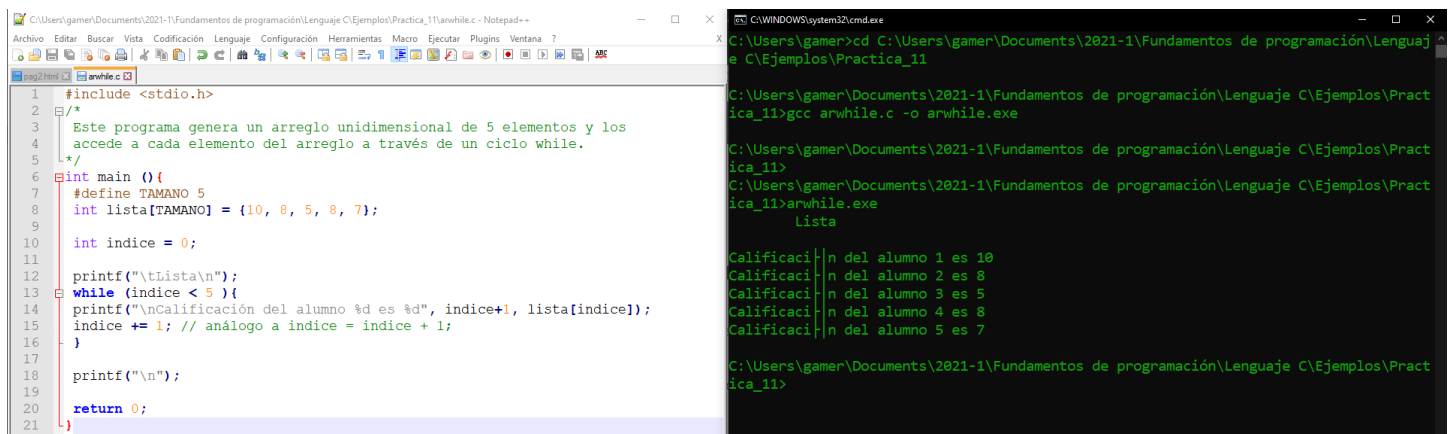
Reconocer la importancia y utilidad de los arreglos, en la elaboración de programas que resuelvan problemas que requieran agrupar datos del mismo tipo, así como trabajar con arreglos tanto unidimensionales como multidimensionales.

Introducción:

Un arreglo es un conjunto de datos contiguos del mismo tipo con un tamaño fijo definido al momento de crearse. A cada elemento (dato) del arreglo se le asocia una posición particular, el cual se requiere indicar para acceder a un elemento en específico. Esto se logra a través del uso de índices. Los arreglos pueden ser unidimensionales o multidimensionales. Los arreglos se utilizan para hacer más eficiente el código de un programa.

Desarrollo:

Arreglo unidimensional while:



The screenshot displays two windows. The left window is a Notepad++ editor showing a C program that uses a while loop to iterate through a 1D array. The right window is a Windows Command Prompt showing the compilation and execution of the program.

```
#include <stdio.h>

/*
Este programa genera un arreglo unidimensional de 5 elementos y los
accede a cada elemento del arreglo a través de un ciclo while.
*/

int main (){
    #define TAMANO 5
    int lista[TAMANO] = {10, 8, 5, 8, 7};

    int indice = 0;

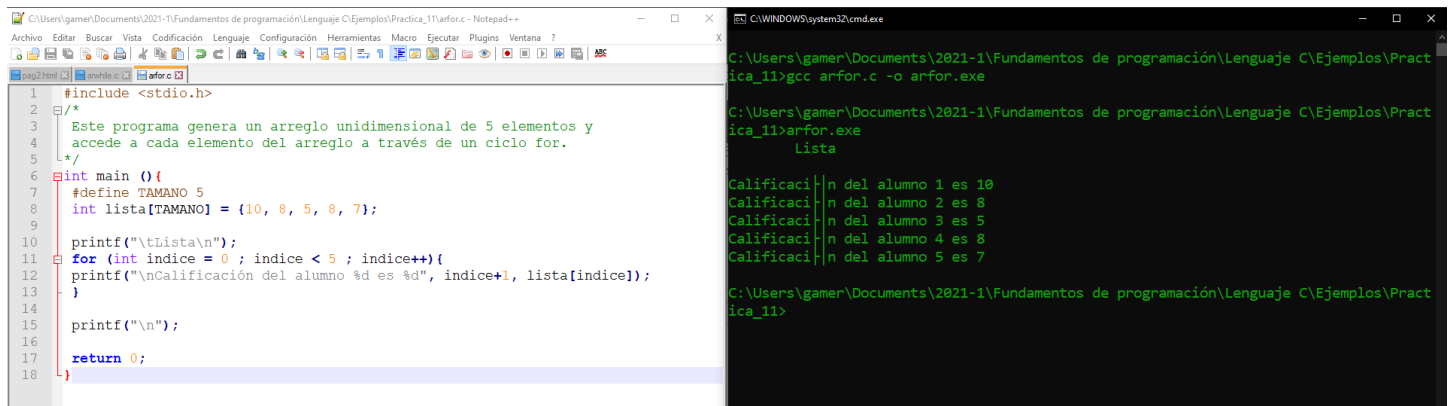
    printf("\tLista\n");
    while (indice < 5 ){
        printf("\nCalificación del alumno %d es %d", indice+1, lista[indice]);
        indice += 1; // análogo a indice = indice + 1;
    }

    printf("\n");

    return 0;
}
```

```
C:\Users\gamer>cd C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc arwhile.c -o arwhile.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>arwhile.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\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>
```

Arreglo unidimensional for:



The screenshot displays two windows. The left window is a Notepad++ editor showing a C program that uses a for loop to iterate through a 1D array. The right window is a Windows Command Prompt showing the compilation and execution of the program.

```
#include <stdio.h>

/*
Este programa genera un arreglo unidimensional de 5 elementos y
accede a cada elemento del arreglo a través de un ciclo for.
*/

int main (){
    #define TAMANO 5
    int lista[TAMANO] = {10, 8, 5, 8, 7};

    printf("\tLista\n");
    for (int indice = 0 ; indice < 5 ; indice++){
        printf("\nCalificación del alumno %d es %d", indice+1, lista[indice]);
    }

    printf("\n");

    return 0;
}
```

```
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc arfor.c -o arfor.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>arfor.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\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>
```

Apuntadores: Un apuntador es una variable que contiene la dirección de una variable, es decir, hace referencia a la localidad de memoria de otra variable. Debido a que los apuntadores trabajan directamente con la memoria, a través de ellos se accede con rapidez a un dato.

```
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11\apuntadores.c - Notepad++
1 #include <stdio.h>
2
3 /*
4  * Este programa crea un apuntador de tipo carácter.
5  */
6 int main () {
7     char *ap, c = 'a';
8     ap = &c;
9
10    printf("Carácter: %c\n", *ap);
11    printf("Código ASCII: %d\n", *ap);
12    printf("Dirección de memoria: %d\n", ap);
13
14    return 0;
15 }
```

```
C:\WINDOWS\system32\cmd.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc apuntadores.c -o apuntadores.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>apuntadores.exe
Carácter: a
Código ASCII: 97
Dirección de memoria: 6422299
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>
```

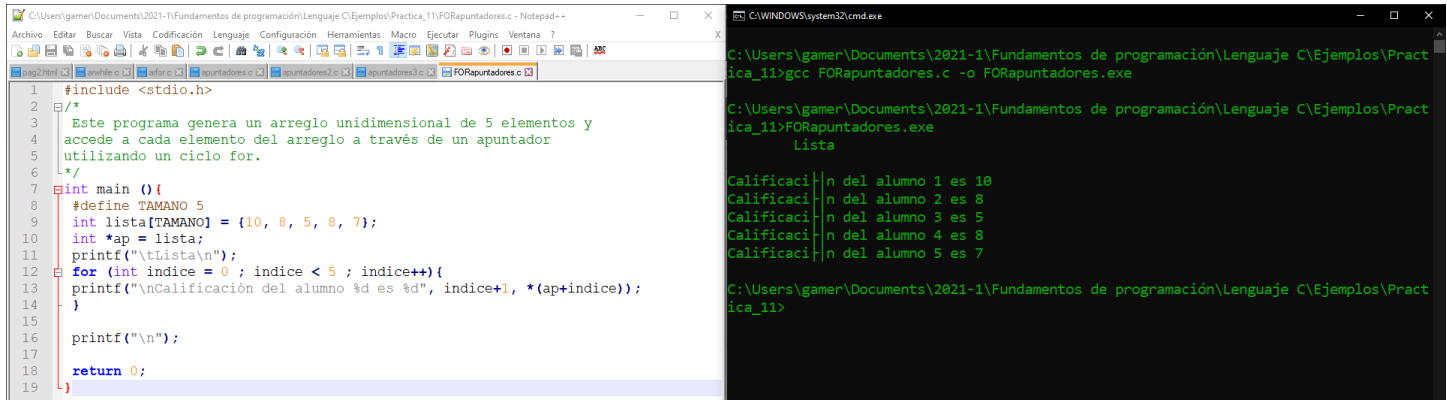
```
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11\apuntadores2.c - Notepad++
1 #include <stdio.h>
2
3 /*
4  * Este programa accede a las localidades de memoria de distintas variables a través de un apuntador.
5  */
6 int main () {
7     int a = 5, b = 10, c[10] = {5, 4, 3, 2, 1, 9, 8, 7, 6, 0};
8     int *apEnt;
9     apEnt = &a;
10
11    printf("a = 5, b = 10, c[10] = {5, 4, 3, 2, 1, 9, 8, 7, 6, 0}\n");
12    printf("apEnt = %a\n");
13
14    b = *apEnt;
15    printf("b = *apEnt \t-> b = %i\n", b);
16
17    b = *apEnt + 1;
18    printf("b = *apEnt + 1 \t-> b = %i\n", b);
19
20    *apEnt = 0;
21    printf("*apEnt = 0 \t-> a = %i\n", a);
22
23    apEnt = &c[0];
24    printf("apEnt = &c[0] \t-> apEnt = %i\n", *apEnt);
25
26    return 0;
27 }
```

```
C:\WINDOWS\system32\cmd.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc apuntadores2.c -o apuntadores2.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>apuntadores2.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\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>
```

```
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11\apuntadores3.c - Notepad++
1 #include <stdio.h>
2
3 /*
4  * Este programa trabaja con aritmética de apuntadores para acceder a los valores de un arreglo.
5  */
6 int main () {
7     int arr[] = {5, 4, 3, 2, 1};
8     int *apArr;
9     apArr = arr;
10
11    printf("int arr[] = {5, 4, 3, 2, 1};\n");
12    printf("apArr = &arr[0]\n");
13
14    int x = *apArr;
15    printf("x = *apArr \t-> x = %d\n", x);
16
17    x = *(apArr+1);
18    printf("x = *(apArr+1) \t-> x = %d\n", x);
19
20    x = *(apArr+2);
21    printf("x = *(apArr+1) \t-> x = %d\n", x);
22
23    return 0;
24 }
```

```
C:\WINDOWS\system32\cmd.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc apuntadores3.c -o apuntadores3.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>apuntadores3.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\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>
```

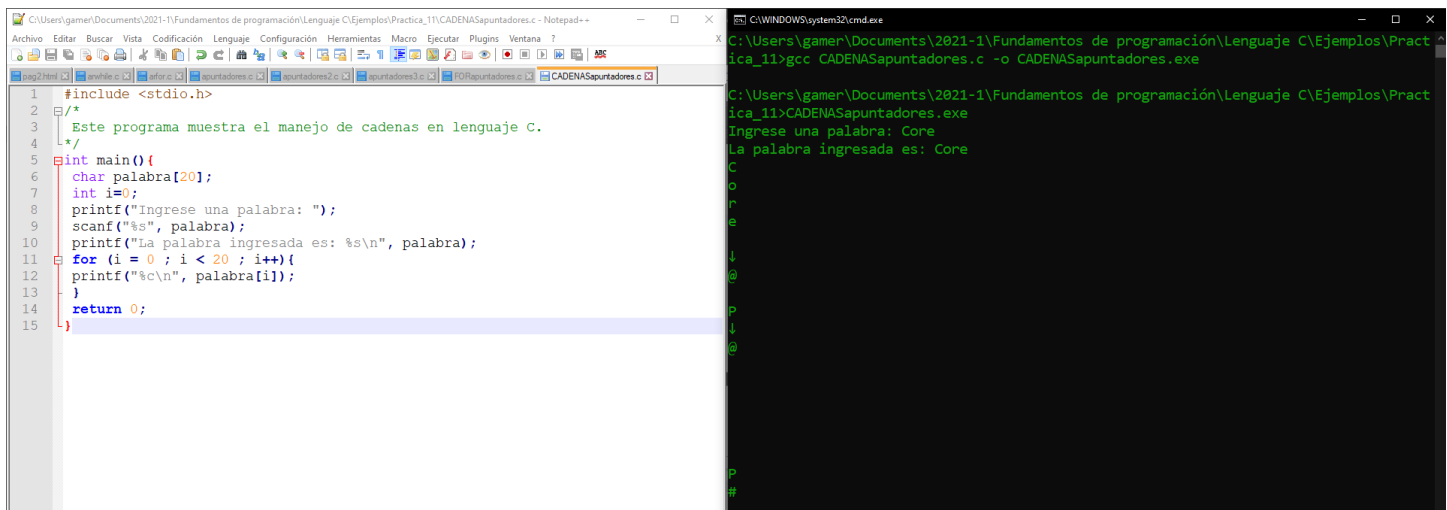
Apuntadores ciclo for:



```
1 #include <stdio.h>
2 /*
3  * Este programa genera un arreglo unidimensional de 5 elementos y
4  * accede a cada elemento del arreglo a través de un apuntador
5  * utilizando un ciclo for.
6  */
7 int main () {
8     #define TAMANO 5
9     int lista[TAMANO] = {10, 8, 5, 8, 7};
10    int *ap = lista;
11    printf("\tLista\n");
12    for (int indice = 0 ; indice < 5 ; indice++){
13        printf("\nCalificación del alumno %d es %d", indice+1, *(ap+indice));
14    }
15
16    printf("\n");
17
18    return 0;
19 }
```

```
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc FORapuntadores.c -o FORapuntadores.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>FORapuntadores.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\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>
```

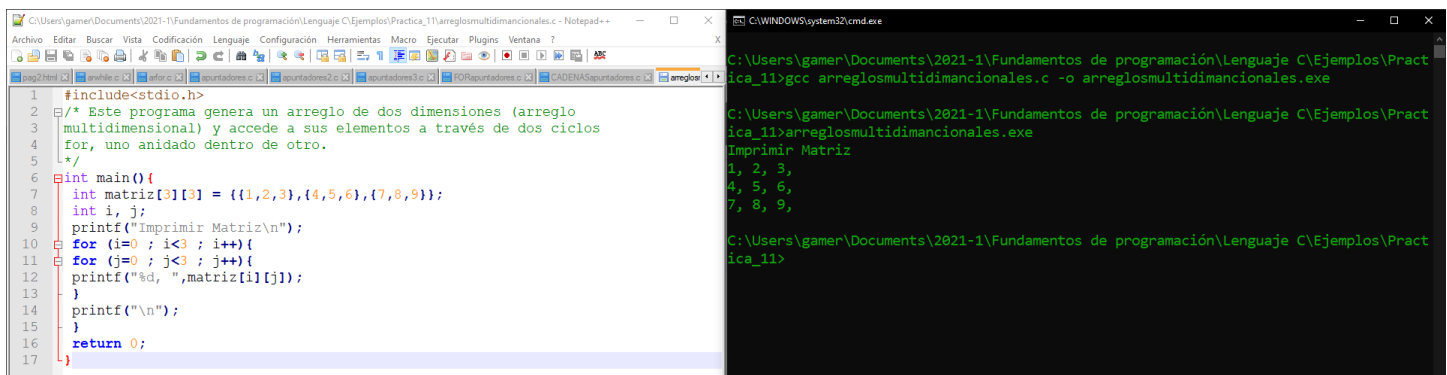
Apuntadores en cadenas:



```
1 #include <stdio.h>
2 /*
3  * Este programa muestra el manejo de cadenas en lenguaje C.
4  */
5 int main() {
6     char palabra[20];
7     int i=0;
8     printf("Ingrese una palabra: ");
9     scanf("%s", palabra);
10    printf("La palabra ingresada es: %s\n", palabra);
11    for (i = 0 ; i < 20 ; i++){
12        printf("%c\n", palabra[i]);
13    }
14    return 0;
15 }
```

```
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc CADENASapuntadores.c -o CADENASapuntadores.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>CADENASapuntadores.exe
Ingrese una palabra: Core
La palabra ingresada es: Core
C
o
r
e
↓
@
↓
@
↓
@
P
↓
@
P
#
```

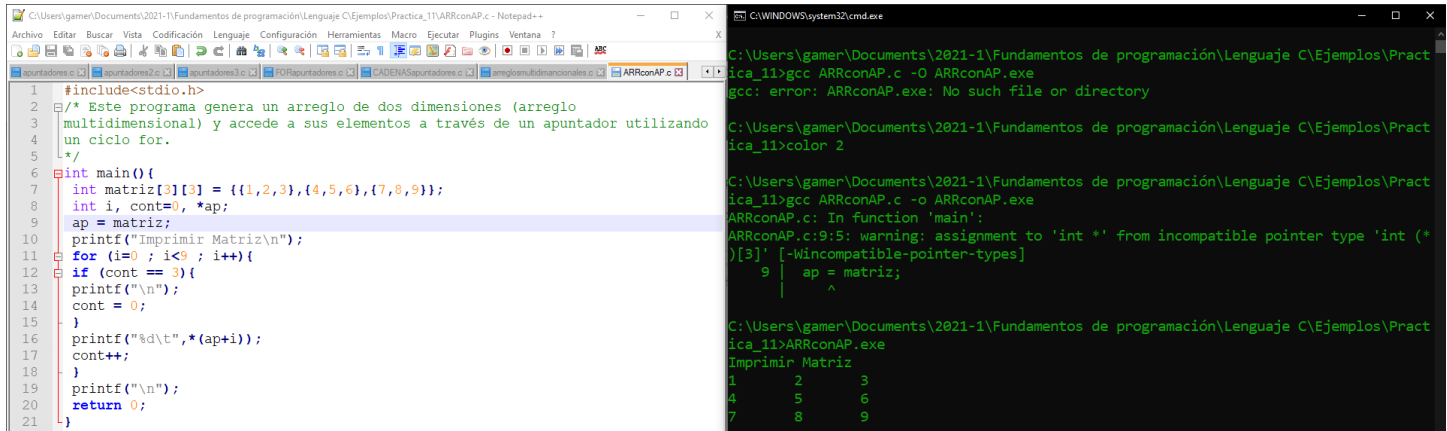
Arreglos multidimensionales: De manera práctica se puede considerar que la primera dimensión corresponde a los renglones, la segunda a las columnas, la tercera al plano, y así sucesivamente. Sin embargo, en la memoria cada elemento del arreglo se guarda de forma contigua, por lo tanto, se puede recorrer un arreglo multidimensional con apuntadores.



```
1 #include <stdio.h>
2 /* Este programa genera un arreglo de dos dimensiones (arreglo
3  * multidimensional) y accede a sus elementos a través de dos ciclos
4  * for, uno anidado dentro de otro.
5  */
6 int main() {
7     int matriz[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
8     int i, j;
9     printf("Imprimir Matriz\n");
10    for (i=0 ; i<3 ; i++){
11        for (j=0 ; j<3 ; j++){
12            printf("%d, ",matriz[i][j]);
13        }
14        printf("\n");
15    }
16    return 0;
17 }
```

```
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc arreglosmultidimensionales.c -o arreglosmultidimensionales.exe
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>arreglosmultidimensionales.exe
Imprimir Matriz
1, 2, 3,
4, 5, 6,
7, 8, 9,
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>
```

Arreglos multidimensionales con apuntadores:



The image shows a C program in a Notepad++ editor and its execution in a Windows Command Prompt. The C program, named `ARRconAP.c`, is located at `C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11\ARRconAP.c`. It includes `<stdio.h>` and contains a `main` function that declares a 3x3 integer array `matriz` with values `{{1,2,3},{4,5,6},{7,8,9}}`. A pointer `ap` is assigned to `matriz`. A `for` loop prints each row of the matrix, with a `printf` statement `printf("%d\t", *(ap+i));` and a `cont++` increment. The output in the command prompt shows the matrix printed as three rows of three numbers each, with a color change command `color 2` and a warning about incompatible pointer types.

```
1 #include<stdio.h>
2 /* Este programa genera un arreglo de dos dimensiones (arreglo
3 multidimensional) y accede a sus elementos a través de un apuntador utilizando
4 un ciclo for.
5 */
6 int main() {
7     int matriz[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
8     int i, cont=0, *ap;
9     ap = matriz;
10    printf("Imprimir Matriz\n");
11    for (i=0; i<9; i++){
12        if (cont == 3){
13            printf("\n");
14            cont = 0;
15        }
16        printf("%d\t", *(ap+i));
17        cont++;
18    }
19    printf("\n");
20    return 0;
21 }
```

```
C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc ARRconAP.c -o ARRconAP.exe
gcc: error: ARRconAP.exe: No such file or directory

C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>color 2

C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>gcc ARRconAP.c -o ARRconAP.exe
ARRconAP.c: In function 'main':
ARRconAP.c:9:5: warning: assignment to 'int *' from incompatible pointer type 'int (*)[3]' [-Wincompatible-pointer-types]
     9 |     ap = matriz;
       |         ^

C:\Users\gamer\Documents\2021-1\Fundamentos de programación\Lenguaje C\Ejemplos\Practica_11>ARRconAP.exe
Imprimir Matriz
1  2  3
4  5  6
7  8  9
```

Conclusiones:

Hasta ahora siento que esta ha sido la practica más compleja que hemos realizado en esta asignatura, debido a que no solo las estructuras de programación lo son, sino que los diferentes tópicos que puede aplicar lo son también, de manera que suma complejidad a su elaboración, sin mencionar que el tema de matrices es algo reciente para mí, sin embargo creo que se pudo realizar y solventar de una forma bastante cómoda, pues si pude comprender a que hace referencia cada uno de los diferentes arreglos y si utilidad en la industria, así como la funcionalidad que puede tener en un programa cualquiera solicitado por una empresa y de manera emprendedora, los arreglos desde mi punto de vista son el paso previo que existe incluso de manera anatómica en una computado hacia lo que trata en la computación gráfica, arreglos multidimensionales que tras una serie de procesos acaban por proyectar una imagen a través de un dispositivo de salida conocido como monitor o proyector, sin duda alguna es un tema imprescindible para el avance del curso y del conocimiento, pues me parece absurdo tratar de avanzar por el lenguaje de programación en general sin tener al menos una idea de lo que representan los arreglos, ya ni hablar del tremendo diferenciador que resulta entre alguien que conoce de manera fluida el manejo de los arreglos ya sean unidimensionales o multidimensionales en la industria, pues lo son prácticamente todo, o al menos la mayoría de aplicaciones que tienen son aplicables directamente en todos los ámbitos humanos existentes, tanto de educación como modelos de negocio centralista, ha ahí la importancia de haber superado este tema y añadirlo a la lista de herramientas del ingeniero.