



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M. I. Marco Antonio Martínez Quintana

Asignatura: Fundamentos de programación

Grupo: 3

No de Práctica(s): #11

Integrante(s): Fausto Ángel Reséndiz Álvarez

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

No. de Lista o Brigada: 38

Semestre: 2021-1

Fecha de entrega: 08/01/2021

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.

Introducción

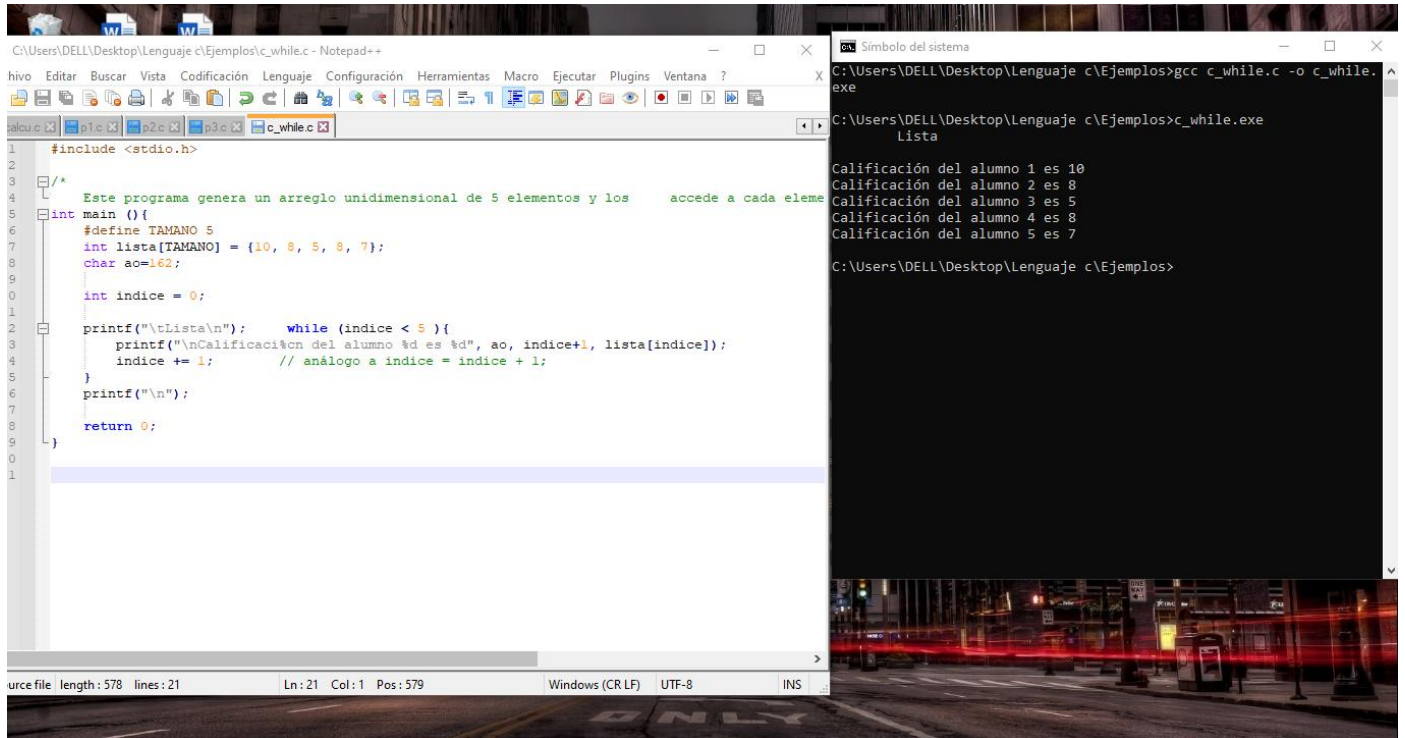
A un conjunto de datos constantes similares entre si con un tamaño delimitado al momento de crearse se le denomina arreglo y estos sirven para darle eficiencia mayor eficiencia al código de un programa.

Los arreglos pueden ser unidimensionales del tipo for, apuntadores; apuntadores en ciclo for y apuntadores en cadenas o multidimensionales de tipo for, apuntadores, apuntadores en ciclo for, apuntadores en cadenas. Y en el desarrollo de esta práctica se ejecutarán algunos ejemplos de los ya mencionados para observar su comportamiento.

Desarrollo

Arreglos unidimensionales

Código (arreglo unidimensional while)



The screenshot shows a C program in Notepad++ and its execution in a Windows Command Prompt. The program defines an array of 5 integers: {10, 8, 5, 8, 7}. It uses a while loop to iterate through the array, printing the index and the corresponding value. The output shows the grades for 5 students.

```
#include <stdio.h>

/*
Este programa genera un arreglo unidimensional de 5 elementos y los accede a cada elemento
*/
int main (){
#define TAMANO 5
int lista[TAMANO] = {10, 8, 5, 8, 7};
char ao=162;

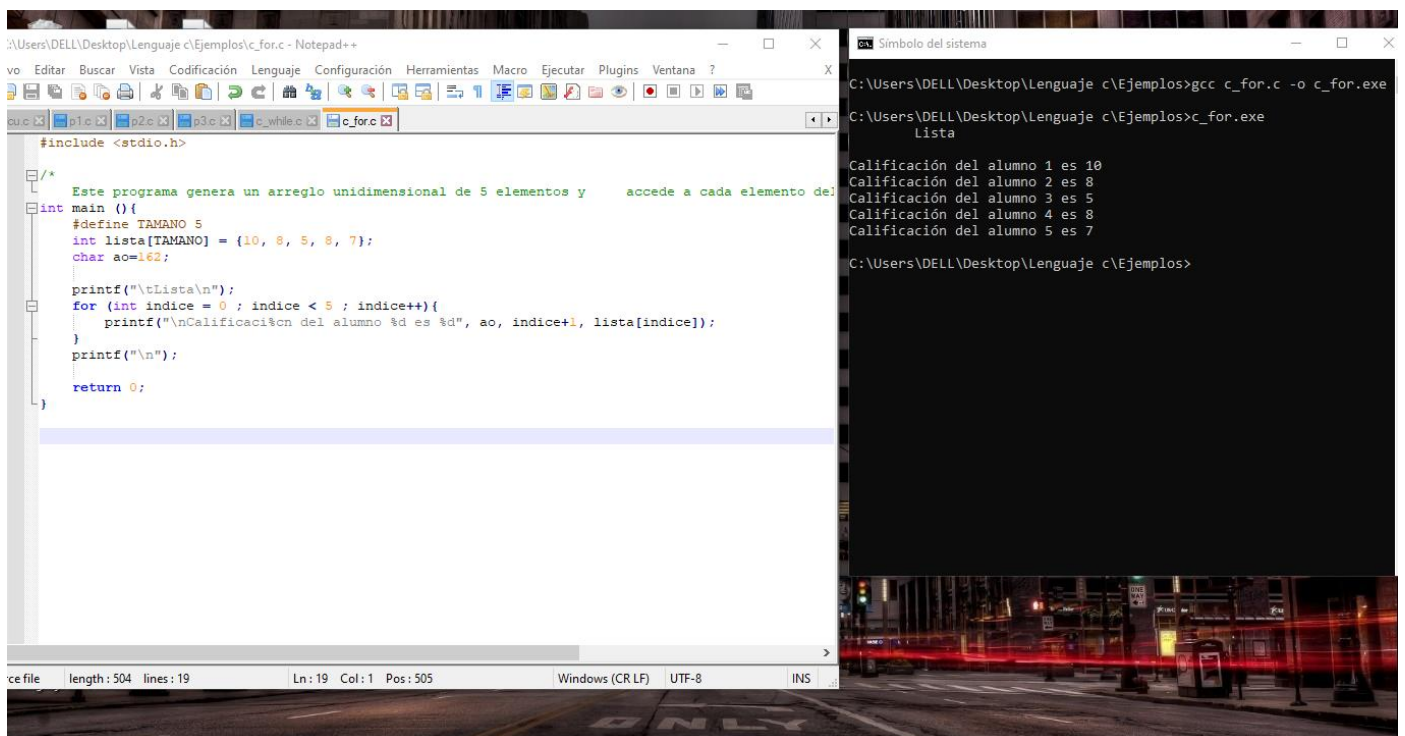
int indice = 0;

printf("\nLista\n"); while (indice < 5){
printf("\nCalificación del alumno %d es %d", ao, indice+1, lista[indice]);
indice += 1; // análogo a indice = indice + 1;
}
printf("\n");
return 0;
}
```

Simbolo del sistema

```
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>gcc c_while.c -o c_while.exe
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>c_while.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\DELL\Desktop\Lenguaje c\Ejemplos>
```

Código (arreglo unidimensional for)



The screenshot shows a C program in Notepad++ and its execution in a Windows Command Prompt. The program defines an array of 5 integers: {10, 8, 5, 8, 7}. It uses a for loop to iterate through the array, printing the index and the corresponding value. The output shows the grades for 5 students.

```
#include <stdio.h>

/*
Este programa genera un arreglo unidimensional de 5 elementos y accede a cada elemento del
*/
int main (){
#define TAMANO 5
int lista[TAMANO] = {10, 8, 5, 8, 7};
char ao=162;

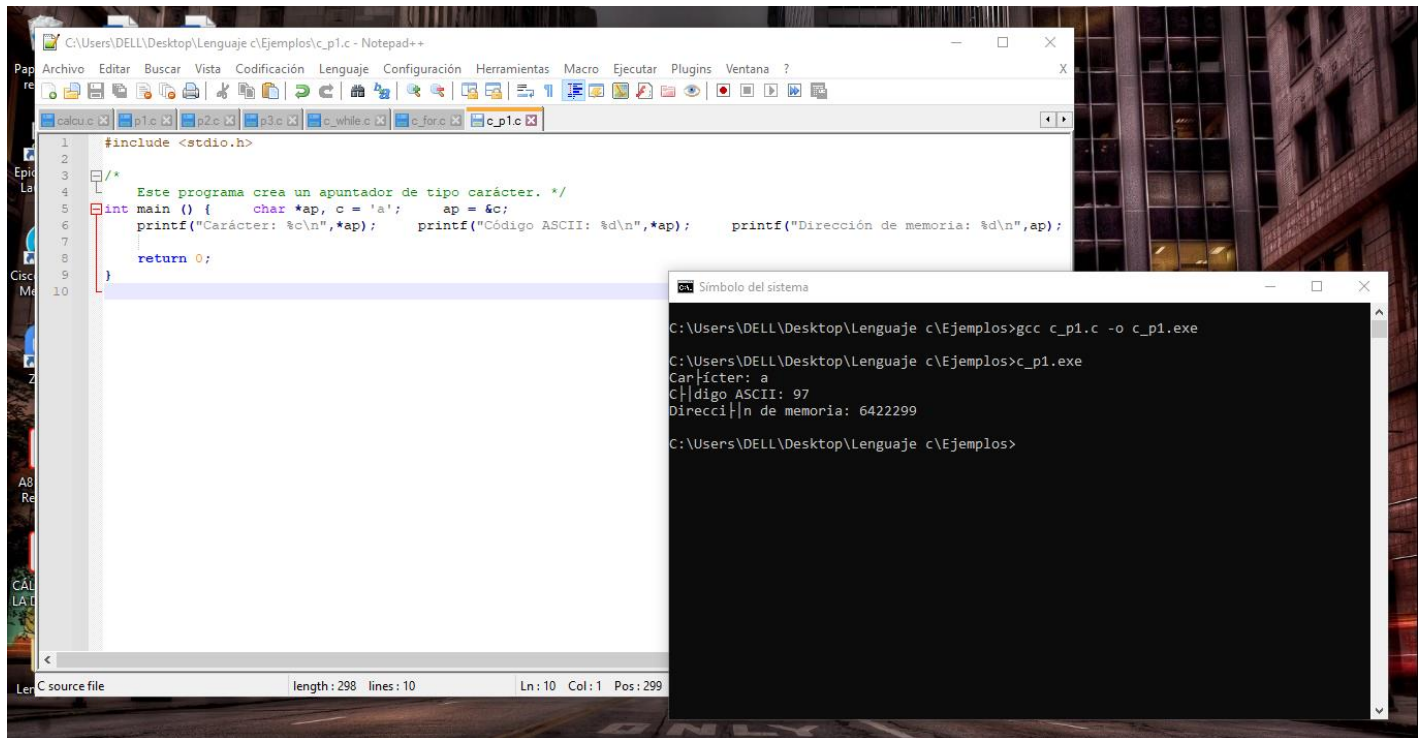
printf("\nLista\n");
for (int indice = 0 ; indice < 5 ; indice++){
printf("\nCalificación del alumno %d es %d", ao, indice+1, lista[indice]);
}
printf("\n");
return 0;
}
```

Simbolo del sistema

```
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>gcc c_for.c -o c_for.exe
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>c_for.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\DELL\Desktop\Lenguaje c\Ejemplos>
```

Apuntadores

Código (apuntadores)



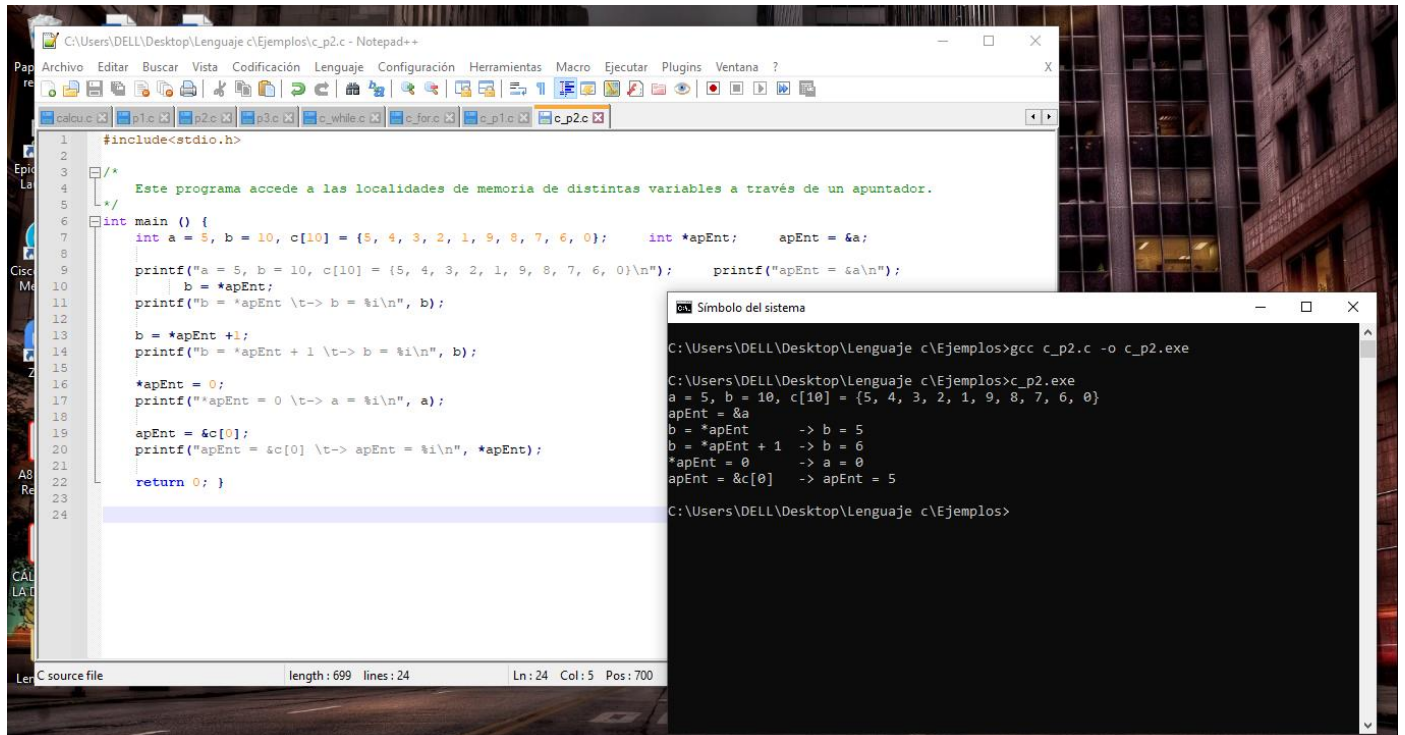
The screenshot shows a C program in Notepad++ and its execution in a command prompt. The program, named `c_p1.c`, includes `<stdio.h>` and has a `main` function that declares a character pointer `ap`, assigns it the address of a character `c` (which is 'a'), and prints the character, its ASCII value, and its memory address. The command prompt shows the compilation and execution of the program, resulting in the output: "Carácter: a", "Código ASCII: 97", and "Dirección de memoria: 6422299".

```
#include <stdio.h>

/*
 * Este programa crea un apuntador de tipo carácter.
 */
int main () {
    char *ap, c = 'a';
    ap = &c;
    printf("Carácter: %c\n", *ap);
    printf("Código ASCII: %d\n", *ap);
    printf("Dirección de memoria: %d\n", ap);
    return 0;
}
```

```
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>gcc c_p1.c -o c_p1.exe
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>c_p1.exe
Carácter: a
Código ASCII: 97
Dirección de memoria: 6422299
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>
```

Código (apuntadores)



The screenshot shows a C program in Notepad++ and its execution in a command prompt. The program, named `c_p2.c`, includes `<stdio.h>` and has a `main` function that declares an integer array `c` and a pointer `apEnt` pointing to its first element. It then prints the array, increments the pointer, and prints the array again. The command prompt shows the compilation and execution of the program, resulting in the output: "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", and "apEnt = &c[0] -> apEnt = 5".

```
#include <stdio.h>

/*
 * Este programa accede a las localidades de memoria de distintas variables a través de un apuntador.
 */
int main () {
    int a = 5, b = 10, c[10] = {5, 4, 3, 2, 1, 9, 8, 7, 6, 0};
    int *apEnt;
    apEnt = &a;

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

    b = *apEnt + 1;
    printf("b = *apEnt + 1 \t-> b = %i\n", b);

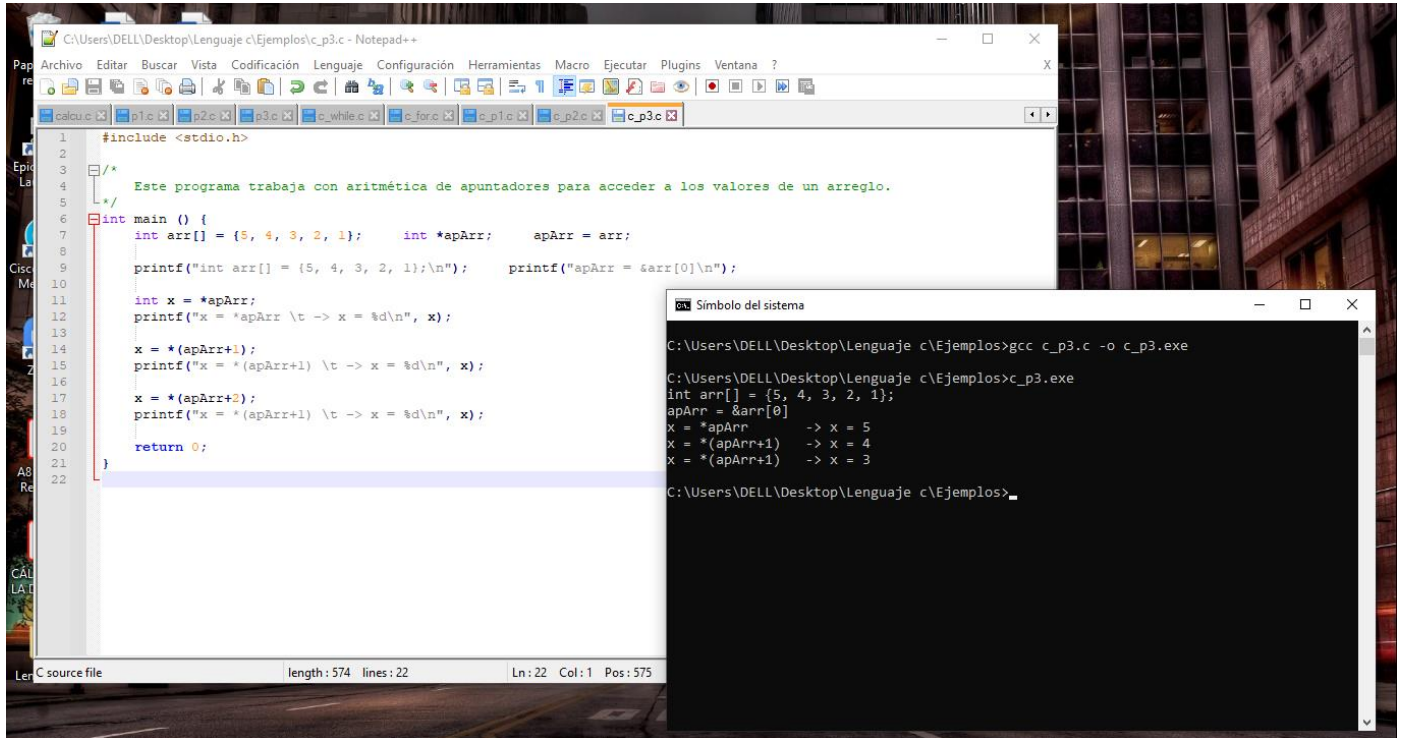
    *apEnt = 0;
    printf("**apEnt = 0 \t-> a = %i\n", a);

    apEnt = &c[0];
    printf("apEnt = &c[0] \t-> apEnt = %i\n", *apEnt);

    return 0;
}
```

```
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>gcc c_p2.c -o c_p2.exe
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>c_p2.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\DELL\Desktop\Lenguaje c\Ejemplos>
```


Código (apuntadores)



The screenshot shows a C program in Notepad++ and its execution in the Windows Command Prompt. The program defines an array `arr` with values {5, 4, 3, 2, 1} and a pointer `apArr` that points to the first element of the array. It then prints the value at `apArr` and increments the pointer to access subsequent elements.

```
#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;    apArr = arr;

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

    int x = *apArr;
    printf("x = *apArr \t -> x = %d\n", x);

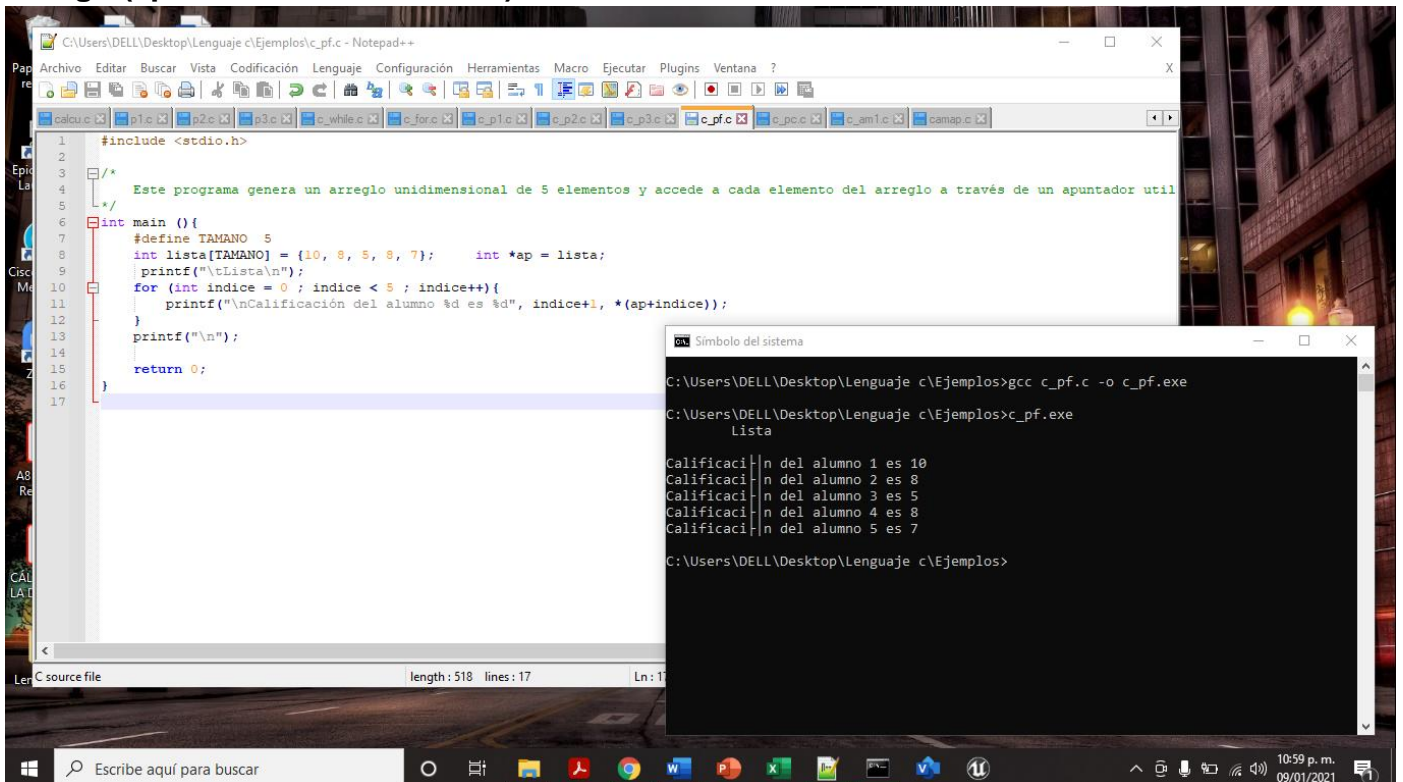
    x = *(apArr+1);
    printf("x = *(apArr+1) \t -> x = %d\n", x);

    x = *(apArr+2);
    printf("x = *(apArr+2) \t -> x = %d\n", x);

    return 0;
}
```

```
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>gcc c_p3.c -o c_p3.exe
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>c_p3.exe
int arr[] = {5, 4, 3, 2, 1};
apArr = &arr[0]
x = *apArr    -> x = 5
x = *(apArr+1) -> x = 4
x = *(apArr+2) -> x = 3
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>
```

Código (apuntadores en ciclo for)



The screenshot shows a C program in Notepad++ and its execution in the Windows Command Prompt. The program defines an array `lista` with values {10, 8, 5, 8, 7} and a pointer `ap` that points to the first element of the array. It then uses a `for` loop to iterate through the array, printing the value at `ap+indice` for each index.

```
#include <stdio.h>

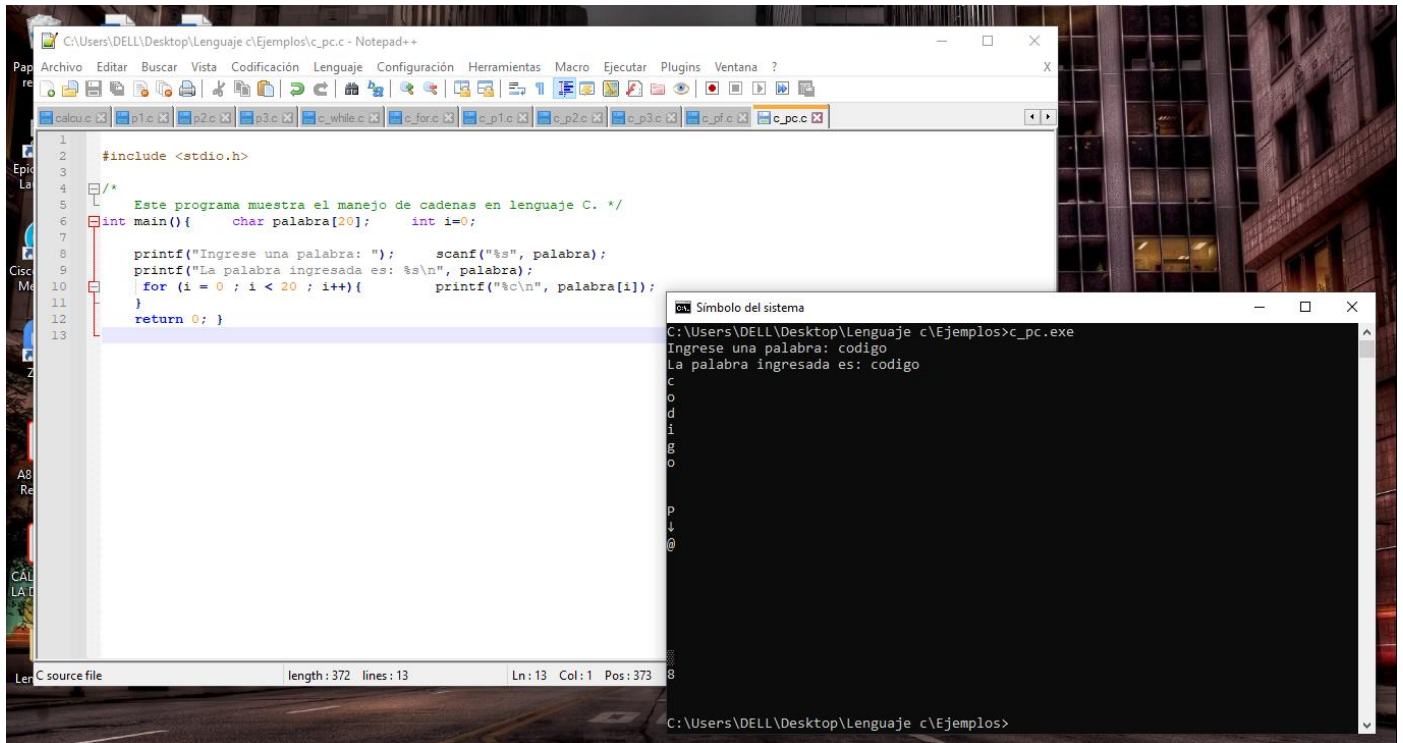
/*
 * Este programa genera un arreglo unidimensional de 5 elementos y accede a cada elemento del arreglo a través de un apuntador util
 */

int main () {
    #define TAMANO 5
    int lista[TAMANO] = {10, 8, 5, 8, 7};    int *ap = lista;
    printf("\nLista\n");
    for (int indice = 0 ; indice < 5 ; indice++) {
        printf("\nCalificación del alumno %d es %d", indice+1, *(ap+indice));
    }
    printf("\n");

    return 0;
}
```

```
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>gcc c_pf.c -o c_pf.exe
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>c_pf.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\DELL\Desktop\Lenguaje c\Ejemplos>
```

Código (apuntadores en cadenas)



```
#include <stdio.h>

/*
Este programa muestra el manejo de cadenas en lenguaje C.
*/
int main() {
    char palabra[20];
    int i=0;

    printf("Ingrese una palabra: ");
    scanf("%s", palabra);
    printf("La palabra ingresada es: %s\n", palabra);
    for (i = 0; i < 20; i++) {
        printf("%c\n", palabra[i]);
    }
    return 0;
}
```

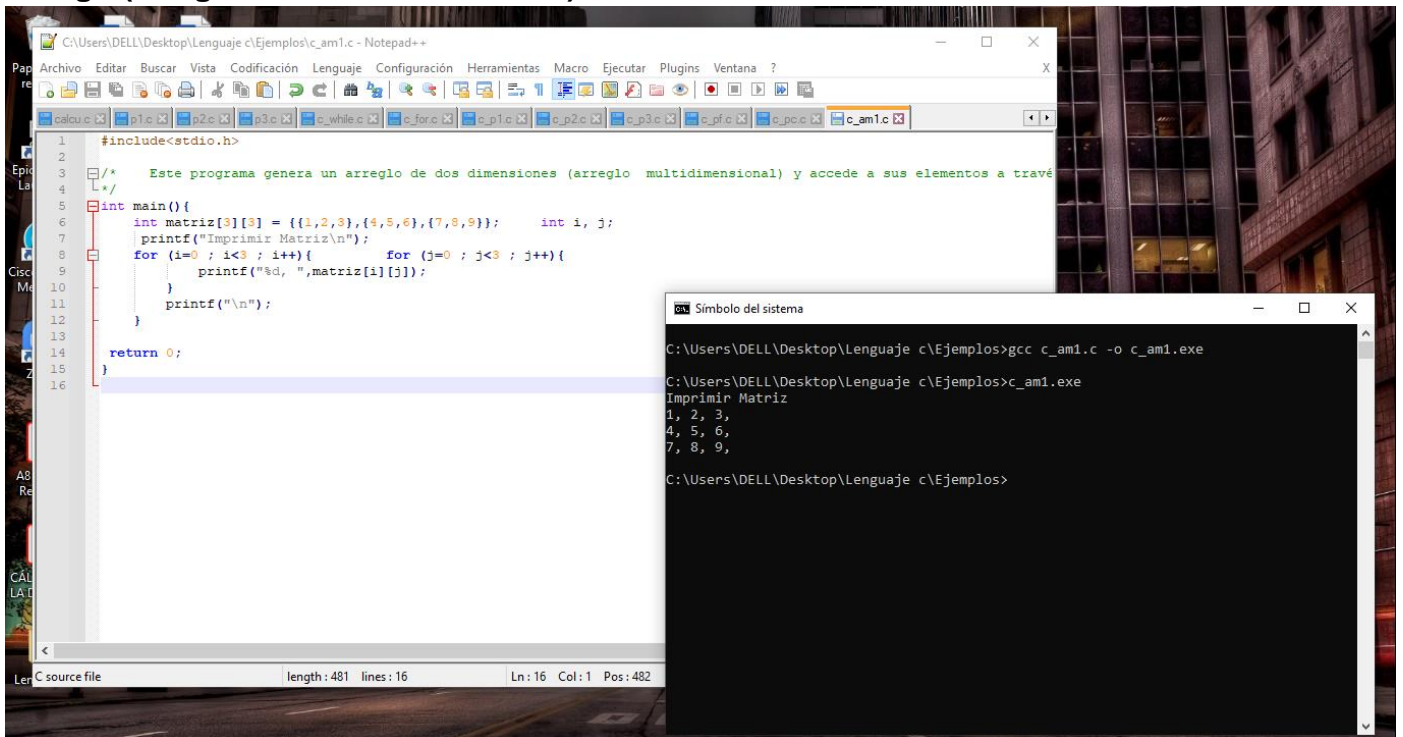
```
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>c_pc.exe
Ingrese una palabra: codigo
La palabra ingresada es: codigo
c
o
d
i
g
o

p
↓
@

C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>
```

Arreglos multidimensionales

Código (arreglos multidimensionales)



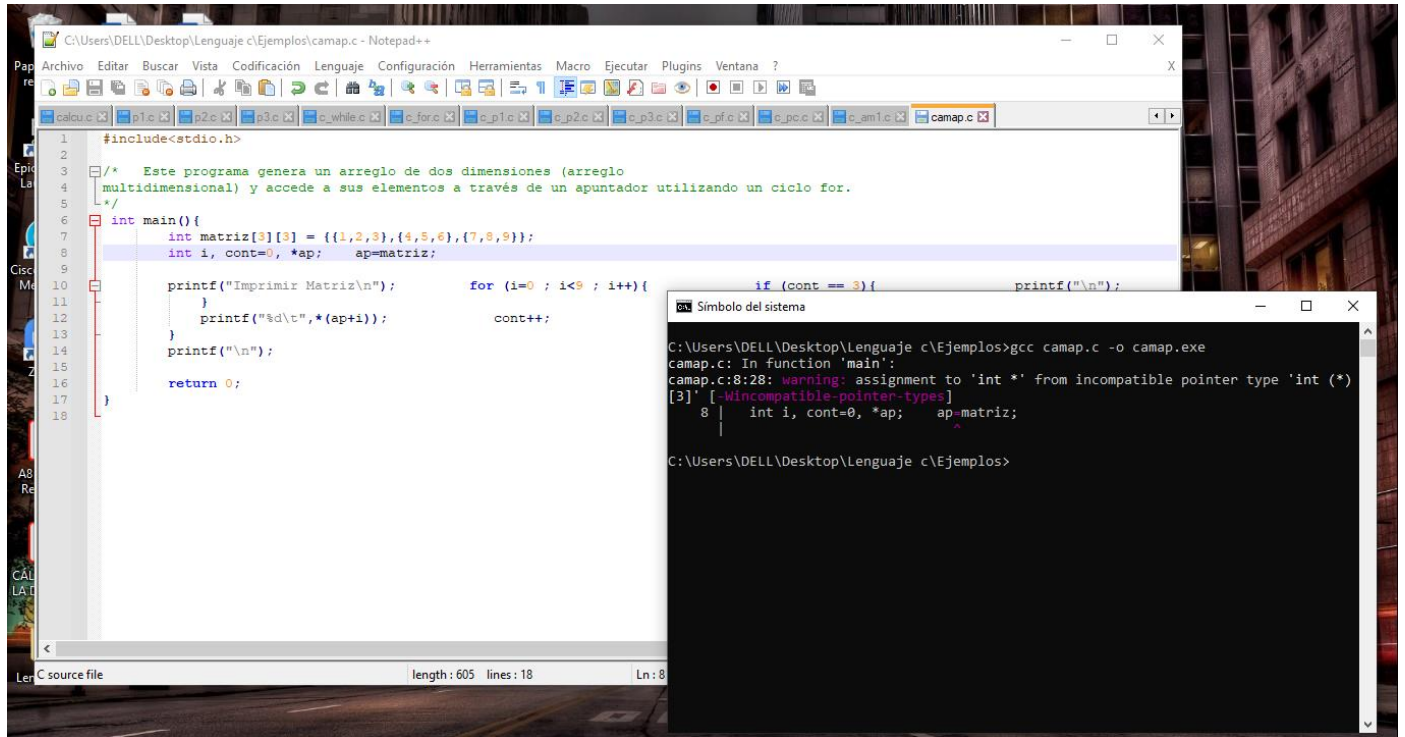
```
#include <stdio.h>

/*
Este programa genera un arreglo de dos dimensiones (arreglo multidimensional) y accede a sus elementos a través de sus índices.
*/
int main() {
    int matriz[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
    printf("Imprimir Matriz\n");
    for (i=0; i<3; i++){
        for (j=0; j<3; j++){
            printf("%d, ",matriz[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

```
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>gcc c_am1.c -o c_am1.exe
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>c_am1.exe
Imprimir Matriz
1, 2, 3,
4, 5, 6,
7, 8, 9,

C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>
```

Código (arreglos multidimensionales con apuntadores)



The image shows a Windows desktop environment. In the background, there is a window titled "C:\Users\DELL\Desktop\Lenguaje c\Ejemplos\camap.c - Notepad++". The code in this window is as follows:

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

In the foreground, there is a "Símbolo del sistema" (Command Prompt) window. It shows the compilation of the program:

```
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>gcc camap.c -o camap.exe
camap.c: In function 'main':
camap.c:8:28: warning: assignment to 'int *' from incompatible pointer type 'int (*)
[3]' [-Wincompatible-pointer-types]
8 |     int i, cont=0, *ap;    ap=matriz;
  |                               ^
C:\Users\DELL\Desktop\Lenguaje c\Ejemplos>
```


Conclusión

En base a lo estudiado anteriormente podríamos concluir definiendo a los arreglos como estructuras de datos que nos permiten almacenar otro tipo de dato dentro del mismo, es decir, es un contenedor que nos permite tener varios datos almacenados. Los datos que almacena un arreglo pueden ser de tipo booleano, cadenas de texto, números entre otros. Los arreglos son una herramienta de gran importancia que proporciona una gran ventaja a los programadores en el desarrollo de un código ya que solo se necesita tener una variable declarada para guardar los demás datos, permitiendo optimizar el proceso de desarrollo y hace más fácil y más clara la comprensión de los programas.