



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M.I. MARCO ANTONIO MARTINEZ QUINTANA

Asignatura: FUNDAMENTOS DE PROGRAMACIÓN

Grupo: 3

No de Práctica(s): 11

Integrante(s): CARRILLO CERVANTES IVETTE ALEJANDRA

*No. de Equipo de
cómputo empleado:* NO APLICA

No. de Lista o Brigada: 7

Semestre: PRIMER SEMESTRE

Fecha de entrega: 04 ENERO 2021

Observaciones:

CALIFICACIÓN: _____

Arreglos unidimensionales y multidimensionales

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

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.

Arreglos unidimensionales

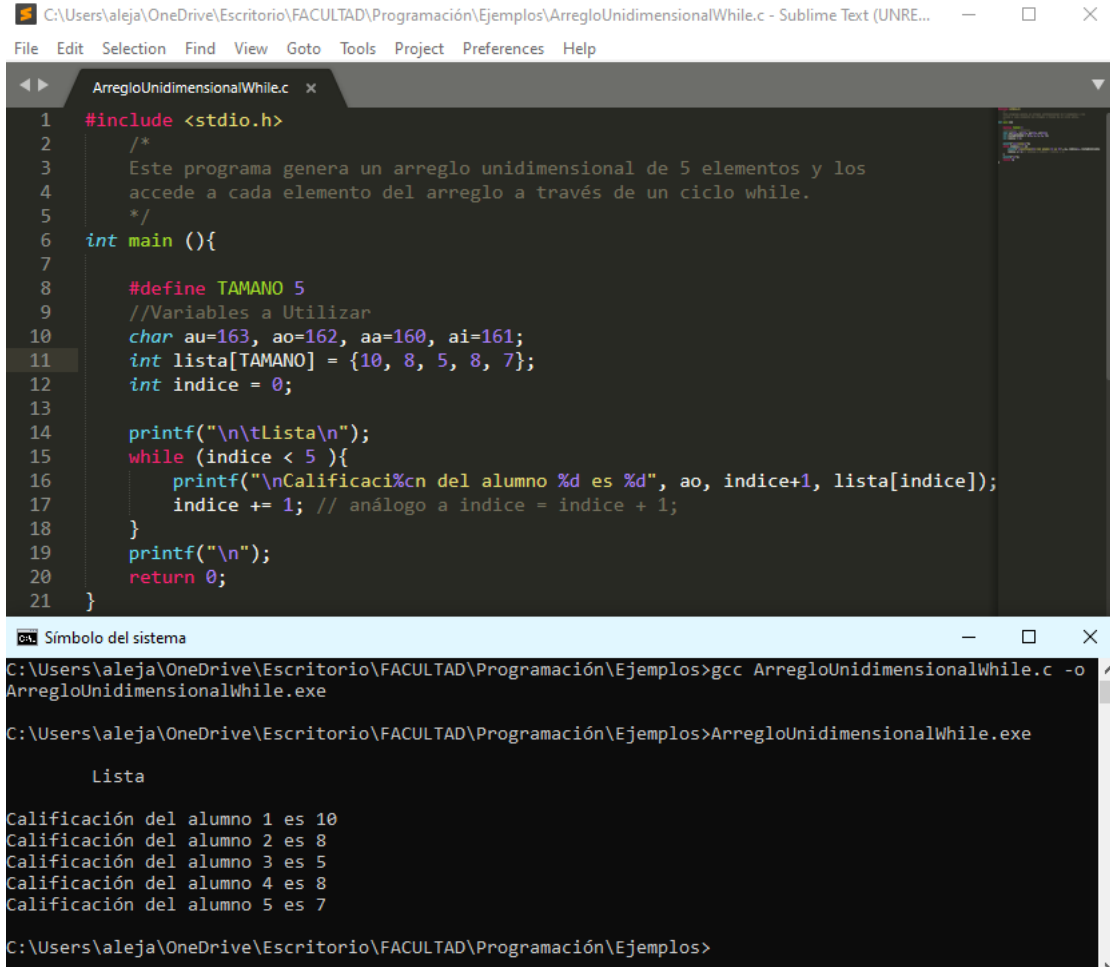
Un arreglo unidimensional de n elementos en la memoria se almacena de la siguiente manera:

La primera localidad del arreglo corresponde al índice 0 y la última corresponde al índice $n-1$, donde n es el tamaño del arreglo.

La sintaxis para definir un arreglo en lenguaje C es la siguiente: `tipoDeDato nombre[tamaño]`

Donde `nombre` se refiere al identificador del arreglo, `tamaño` es un número entero y define el número máximo de elementos que puede contener el arreglo. Un arreglo puede ser de los tipos de dato entero, real, carácter o estructura.

Código (arreglo unidimensional while)



```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\ArregloUnidimensionalWhile.c - Sublime Text (UNRE...
File Edit Selection Find View Goto Tools Project Preferences Help

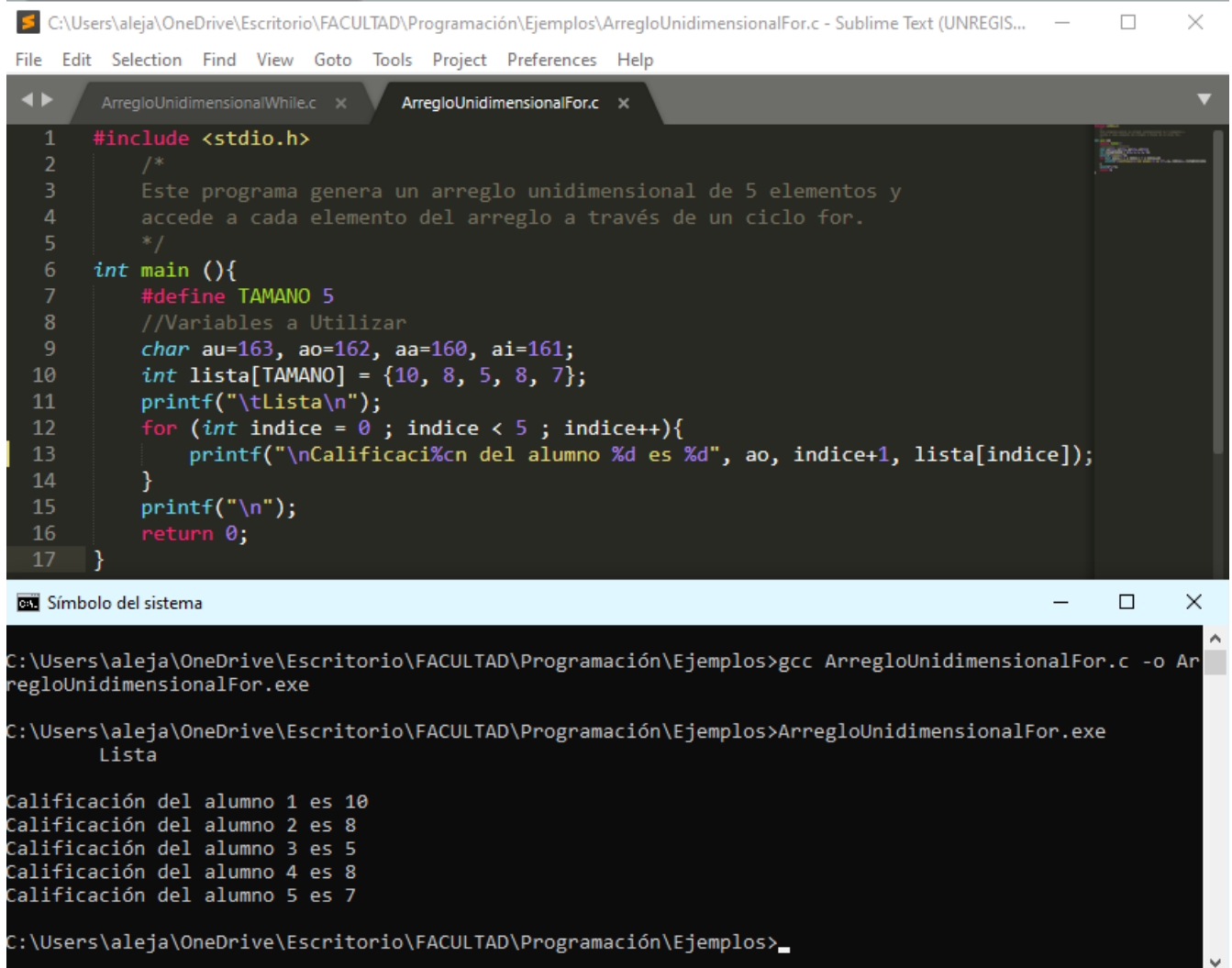
ArregloUnidimensionalWhile.c x
1  #include <stdio.h>
2  /*
3   Este programa genera un arreglo unidimensional de 5 elementos y los
4   accede a cada elemento del arreglo a través de un ciclo while.
5   */
6  int main (){
7
8   #define TAMANO 5
9   //Variables a Utilizar
10  char au=163, ao=162, aa=160, ai=161;
11  int lista[TAMANO] = {10, 8, 5, 8, 7};
12  int indice = 0;
13
14  printf("\n\tLista\n");
15  while (indice < 5 ){
16      printf("\nCalificaci%cn del alumno %d es %d", ao, indice+1, lista[indice]);
17      indice += 1; // análogo a indice = indice + 1;
18  }
19  printf("\n");
20  return 0;
21  }
```

```
Símbolo del sistema
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc ArregloUnidimensionalWhile.c -o ArregloUnidimensionalWhile.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>ArregloUnidimensionalWhile.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\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

Código (arreglo unidimensional For)



```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\ArregloUnidimensionalFor.c - Sublime Text (UNREGIS...
File Edit Selection Find View Goto Tools Project Preferences Help

ArregloUnidimensionalWhile.c x ArregloUnidimensionalFor.c x
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 ciclo for.
5  */
6 int main (){
7     #define TAMANO 5
8     //Variables a Utilizar
9     char au=163, ao=162, aa=160, ai=161;
10    int lista[TAMANO] = {10, 8, 5, 8, 7};
11    printf("\tLista\n");
12    for (int indice = 0 ; indice < 5 ; indice++){
13        printf("\nCalificaci%cn del alumno %d es %d", ao, indice+1, lista[indice]);
14    }
15    printf("\n");
16    return 0;
17 }

Simbolo del sistema
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc ArregloUnidimensionalFor.c -o ArregloUnidimensionalFor.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>ArregloUnidimensionalFor.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\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

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.

La sintaxis para declarar un apuntador y para asignarle la dirección de memoria de otra variable es, respectivamente:

TipoDeDato *apuntador, variable;

apuntador = &variable;

La declaración de una variable apuntador inicia con el carácter *. Cuando a una variable le antecede un ampersand, lo que se hace es acceder a la dirección de memoria de la misma (es lo que pasa cuando se lee un dato con scanf).

Los apuntadores solo pueden apuntar a direcciones de memoria del mismo tipo de dato con el que fueron declarados; para acceder al contenido de dicha dirección, a la variable apuntador se le antepone *.

Código (apuntadores)

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\Apuntadores.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

ArregloUnidimensionalWhile.c x ArregloUnidimensionalFor.c x Apuntadores.c x

1 #include <stdio.h>
2 /*
3  Este programa crea un apuntador de tipo carácter.
4  */
5 int main () {
6     char *ap, c = 'a', au=163, ao=162, aa=160, ai=161;
7     ap = &c;
8     printf("Car%cter: %c\n", aa,*ap);
9     printf("Código ASCII: %d\n", ao, *ap);
10    printf("Direcci%cn de memoria: %d\n", ao, ap);
11    return 0;
12 }

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc Apuntadores.c -o Apuntadores.exe

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>Apuntadores.exe
Carácter: a
Código ASCII: 97
Dirección de memoria: 6422295

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

Código (apuntadores)

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\Apuntadores2.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

ArregloUnidimensionalWhile.c x ArregloUnidimensionalFor.c x Apuntadores.c x Apuntadores2.c x

1 #include<stdio.h>
2 /*
3  Este programa accede a las localidades de memoria de distintas variables a
4  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    printf("a = 5, b = 10, c[10] = {5, 4, 3, 2, 1, 9, 8, 7, 6, 0}\n");
11    printf("apEnt = &a\n");
12    b = *apEnt;
13    printf("b = *apEnt \t-> b = %i\n", b);
14    b = *apEnt +1;
15    printf("b = *apEnt + 1 \t-> b = %i\n", b);
16    *apEnt = 0;
17    printf("*apEnt = 0 \t-> a = %i\n", a);
18    apEnt = &c[0];
19    printf("apEnt = &c[0] \t-> apEnt = %i\n", *apEnt);
20    return 0;
21 }

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>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\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

Cabe mencionar que el nombre de un arreglo es un apuntador fijo al primero de sus elementos; por lo que las siguientes instrucciones, para el código de arriba, son equivalentes:

apEnt = &c[0];

apEnt = c;

Código (apuntadores)

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\Apuntadores3.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

ArregloUnidimensionalWhile.c  ArregloUnidimensionalFor.c  Apuntadores.c  Apuntadores2.c  Apuntadores3.c

1  #include <stdio.h>
2  /*
3   Este programa trabaja con aritmética de apuntadores para acceder a los
4   valores de un arreglo.
5   */
6  int main () {
7      int arr[] = {5, 4, 3, 2, 1};
8      int *apArr;
9      apArr = arr;
10     printf("int arr[] = {5, 4, 3, 2, 1};\n");
11     printf("apArr = &arr[0]\n");
12     int x = *apArr;
13     printf("x = *apArr \t -> x = %d\n", x);
14     x = *(apArr+1);
15     printf("x = *(apArr+1) \t -> x = %d\n", x);
16     x = *(apArr+2);
17     printf("x = *(apArr+1) \t -> x = %d\n", x);
18     return 0;
19 }
```

```
Simbolo del sistema
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc Apuntadores3.c -o Apuntadores3.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>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\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

Código (apuntadores en ciclo for)

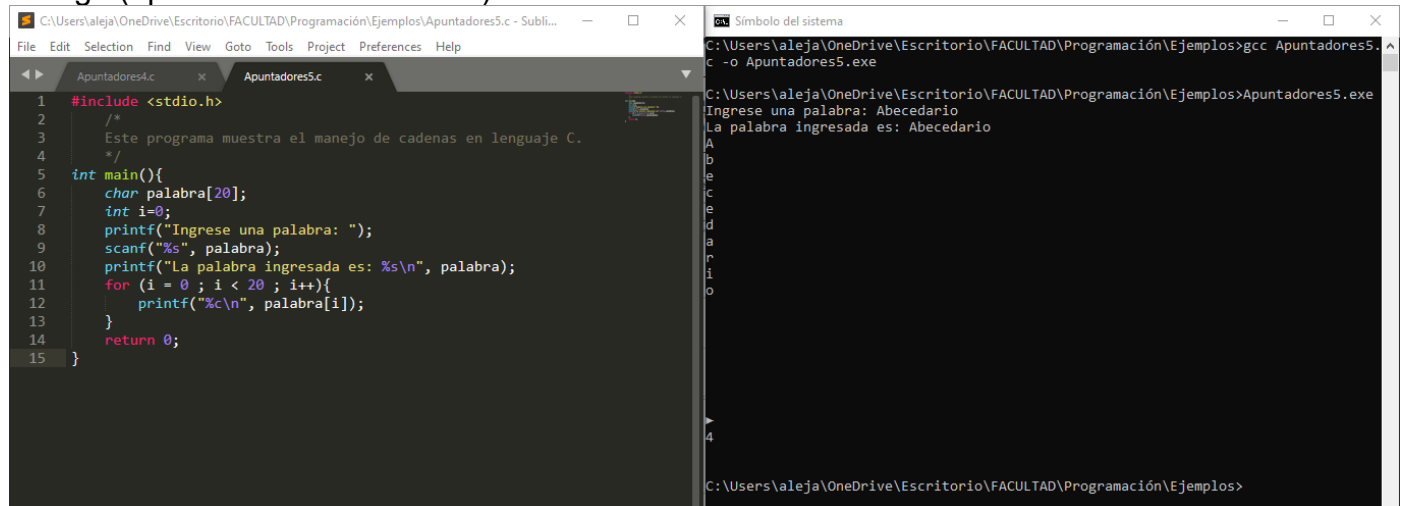
```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\Apuntadores4.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

Apuntadores4.c

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     char au=163, ao=162, aa=160, ai=161;
12     printf("\tLista\n");
13     for (int indice = 0 ; indice < 5 ; indice++){
14         printf("\nCalificaci%Cn del alumno %d es %d", ao, indice+1, *(ap+indice));
15     }
16     printf("\n");
17     return 0;
18 }
```

```
Simbolo del sistema
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc Apuntadores4.c -o Apuntadores4.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>Apuntadores4.exe
Lista
Calificacin del alumno 1 es 10
Calificacin del alumno 2 es 8
Calificacin del alumno 3 es 5
Calificacin del alumno 4 es 8
Calificacin del alumno 5 es 7
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

Código (apuntadores en cadenas)



```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\Apuntadores5.c - Subli...
File Edit Selection Find View Goto Tools Project Preferences Help

Apuntadores4.c x Apuntadores5.c x
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\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc Apuntadores5.c -o Apuntadores5.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>Apuntadores5.exe
Ingrese una palabra: Abecedario
La palabra ingresada es: Abecedario
A
b
c
d
e
f
g
h
i
o
```

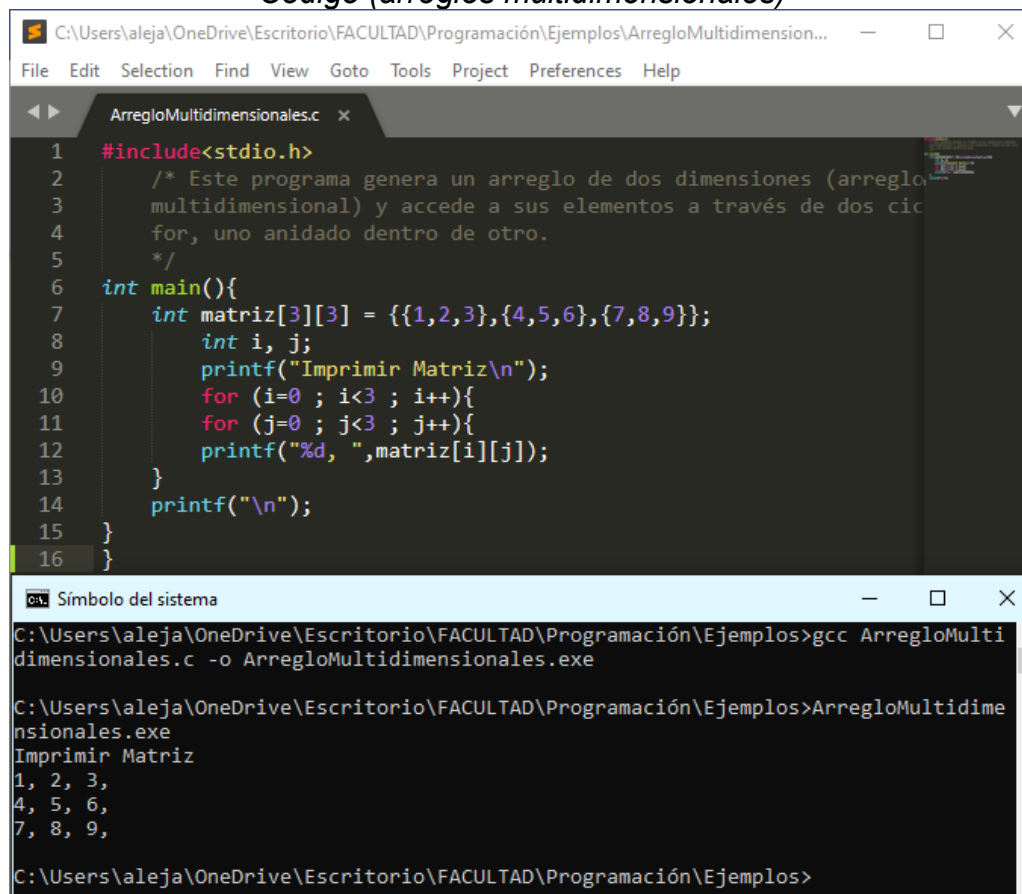
Arreglos multidimensionales

Lenguaje C permite crear arreglos de varias dimensiones con la siguiente sintaxis: tipoDato nombre[tamaño][tamaño]...[tamaño];

Donde nombre se refiere al identificador del arreglo, tamaño es un número entero y define el número máximo de elementos que puede contener el arreglo por dimensión (el número de dimensiones está determinado por el número de corchetes). Los tipos de dato que puede tolerar un arreglo multidimensional son: entero, real, carácter o estructura.

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.

Código (arreglos multidimensionales)

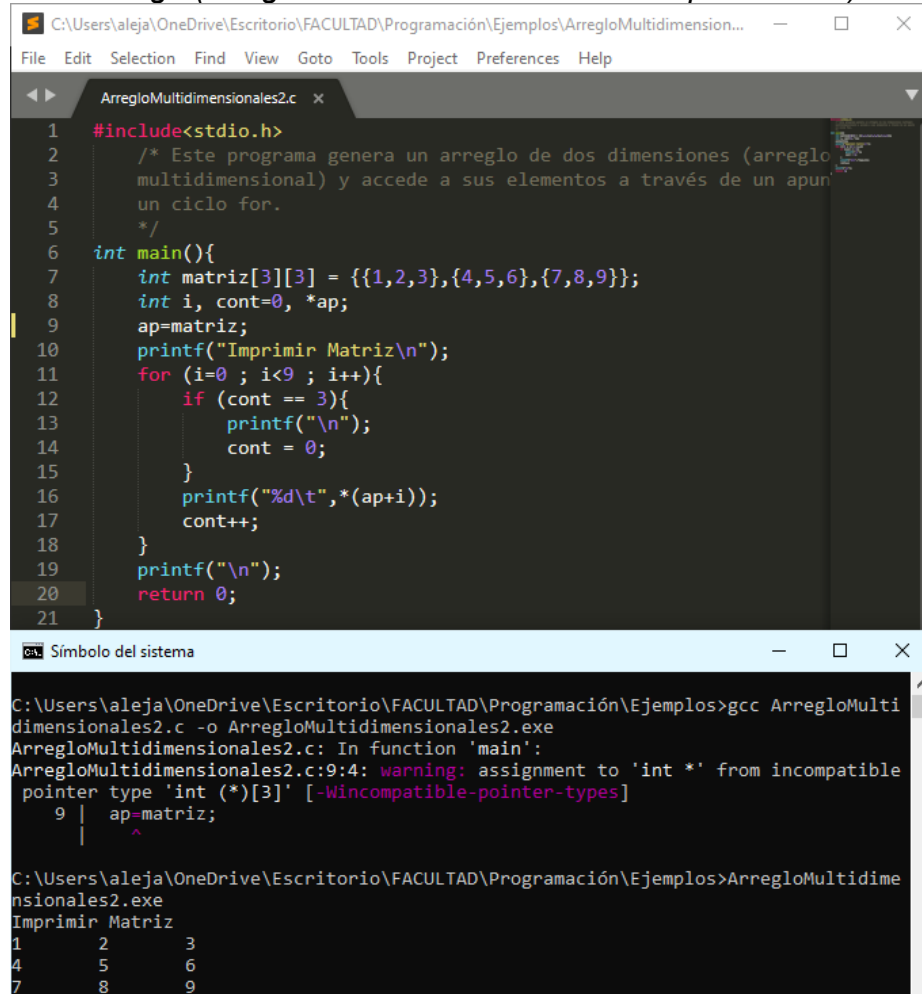


```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\ArregloMultidimension...
File Edit Selection Find View Goto Tools Project Preferences Help

ArregloMultidimensionales.c x
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 cic
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 }

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc ArregloMulti
dimensionales.c -o ArregloMultidimensionales.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>ArregloMultidime
nsionales.exe
Imprimir Matriz
1, 2, 3,
4, 5, 6,
7, 8, 9,
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

Código (arreglos multidimensionales con apuntadores)



The screenshot shows a C program named `ArregloMultidimensionales2.c` and its execution output in a Windows command prompt.

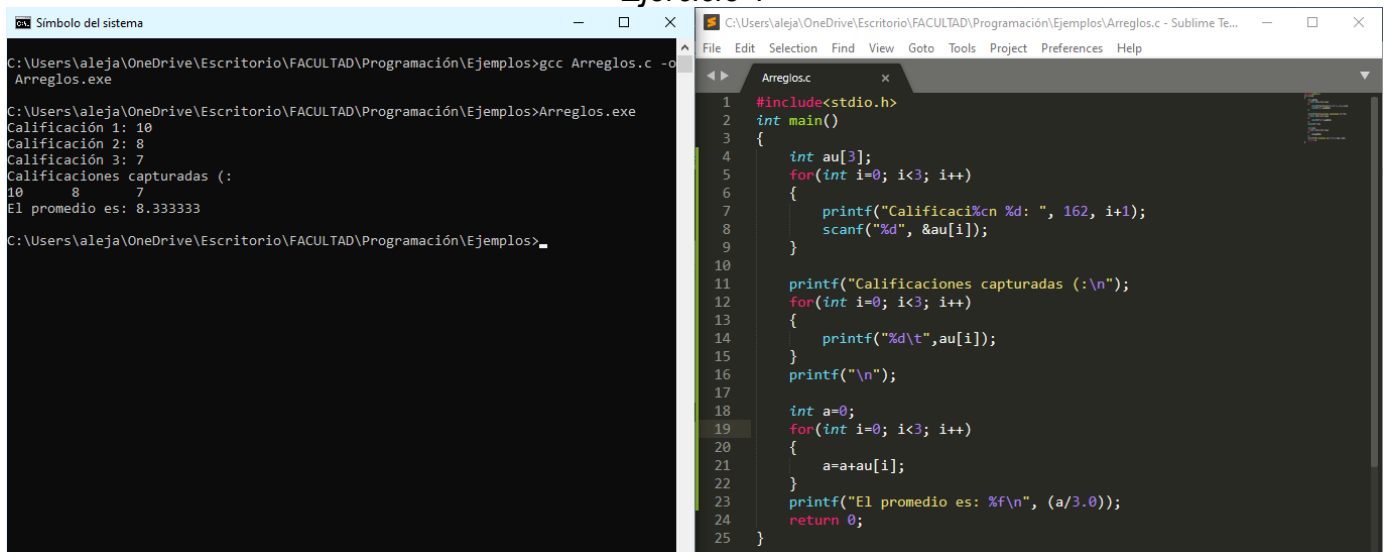
```
#include<stdio.h>
/* Este programa genera un arreglo de dos dimensiones (arreglo
multidimensional) y accede a sus elementos a través de un apun
un ciclo for.
*/
int main(){
    int matriz[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
    int i, cont=0, *ap;
    ap=matriz;
    printf("Imprimir Matriz\n");
 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

The command prompt shows the compilation and execution of the program:

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc ArregloMulti
dimensionales2.c -o ArregloMultidimensionales2.exe
ArregloMultidimensionales2.c: In function 'main':
ArregloMultidimensionales2.c:9:4: warning: assignment to 'int *' from incompatible
pointer type 'int (*)[3]' [-Wincompatible-pointer-types]
    9 | ap=matriz;
      | ^
      |
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>ArregloMultidime
nsionales2.exe
Imprimir Matriz
1      2      3
4      5      6
7      8      9
```

Ejercicios Propuestos

Ejercicio 1



The screenshot shows a C program named `Arreglos.c` and its execution output in a Windows command prompt.

```
#include<stdio.h>
int main()
{
    int au[3];
    for(int i=0; i<3; i++)
    {
        printf("Calificación %cn %d: ", 162, i+1);
        scanf("%d", &au[i]);
    }

    printf("Calificaciones capturadas (: \n");
    for(int i=0; i<3; i++)
    {
        printf("%d\t", au[i]);
    }
    printf("\n");

    int a=0;
    for(int i=0; i<3; i++)
    {
        a+=au[i];
    }
    printf("El promedio es: %f\n", (a/3.0));
    return 0;
}
```

The command prompt shows the compilation and execution of the program:

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc Arreglos.c -o
Arreglos.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>Arreglos.exe
Calificación 1: 10
Calificación 2: 8
Calificación 3: 7
Calificaciones capturadas (:
10      8      7
El promedio es: 8.333333
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

Ejercicio 2

The screenshot shows two windows. On the left, a Windows command prompt window titled 'Símbolo del sistema' displays the execution of a C program. The program prompts for three students' scores and calculates their averages. The output shows scores for three students and their respective averages, followed by the overall group average.

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc Arreglos2.c -o Arreglos2.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>Arreglos2.exe
Calificación 1 del alumno 1: 10
Calificación 2 del alumno 1: 9
Calificación 3 del alumno 1: 10
Calificación 1 del alumno 2: 6
Calificación 2 del alumno 2: 8
Calificación 3 del alumno 2: 5
Calificación 1 del alumno 3: 9
Calificación 2 del alumno 3: 7
Calificación 3 del alumno 3: 6
Calificaciones capturadas (:
10 9 10
6 8 5
9 7 6

El promedio del alumno 1 es: 9.666667
El promedio del alumno 2 es: 6.333333
El promedio del alumno 3 es: 7.333333
El promedio general de grupo es: 7.333333
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

On the right, a Sublime Text editor window titled 'C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\Arreglos2.c - Sublime T...' shows the source code of the program. The code uses nested loops to capture scores for three students and calculate their averages and the overall group average.

```
#include<stdio.h>
int main()
{
    int ab[3][3];
    for(int i=0; i<3; i++)
    {
        for(int j=0; j<3; j++)
        {
            printf("Calificación %cn %d del alumno %d: ", 162, j+1, i+1);
            scanf("%d", &ab[i][j]);
        }
    }
    printf("Calificaciones capturadas (: \n");
    for (int i=0; i<3; i++)
    {
        for(int j=0; j<3; j++)
        {
            printf("%d\t", ab[i][j]);
        }
        printf("\n");
    }
    printf("\n");

    float p[3];
    int a=0;
    for(int i=0; i<3; i++)
    {
        a=0;
        for (int j = 0; j < 3; j++)
        {
            a=a+ab[i][j];
        }
        p[i]=a/3.0;
        printf("\nEl promedio del alumno %d es: %f", i+1, p[i]);
    }
    a=0;
    for (int i = 0; i < 3; i++)
    {
        a=a+p[i];
    }
    printf("\nEl promedio general de grupo es: %f\n", (a/3.0));
    return 0;
}
```

Actividades:

Crear un sistema que almacene el inventario de una tienda en un arreglo y al final nos dé la cantidad total de artículos que tenemos en existencia.

The screenshot shows a Sublime Text editor window titled 'C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\Dulceria.c - Sublime Text (UNREGISTERED)'. The code defines a 3x3 array 'ab' to store the inventory of a candy store. It uses nested loops to display the inventory and calculate the total number of items.

```
#include<stdio.h>
int main()
{
    int ab[3][3];

    printf("\n\t\tDULCERIA (: \nSecciones: \n1) Chocolates \t \n2) Paletas \t \n3) Bolsa de Chicless \n");
    printf("\n\t\t1) Chocolates tipo \n1. Conejos Turin \n2. Bombones de Chocolate \n3. Carlos V \n");
    printf("\n\t\t2) Paletas tipo \n1. Tutsi \n2. Cajeta \n3. Coraz \n");
    printf("\n\t\t3) Bolsa de Chicless tipo \n1. De bola \n2. Bubli \n3. Canels \n");

    for(int i=0; i<3; i++)
    {
        for(int j=0; j<3; j++)
        {
            printf("%cCu%cntos dulces de tipo %d hay en la secci%cn %d) ? ", 168, 160, j+1, 162, i+1);
            scanf("%d", &ab[i][j]);
        }
    }
    printf("\n%cmerno de dulces capturado (: \n", 163);
    for (int i=0; i<3; i++)
    {
        for(int j=0; j<3; j++)
        {
            printf("%d\t", ab[i][j]);
        }
        printf("\n");
    }
    printf("\n");

    int p[3];
    int a=0;
    for(int i=0; i<3; i++)
    {

```



```

34     a=0;
35     for (int j = 0; j < 3; j++)
36     {
37         a=a+ab[i][j];
38     }
39     p[i]=a;
40     printf("\nEl n%cmero total de dulces en la secci%cno.%d es: %d", 163, 162, i+1, p[i]);
41 }
42
43 a=0;
44 for (int i = 0; i < 3; i++)
45 {
46     a=a+p[i];
47 }
48 printf("\n\nEl n%cmero total de dulces en la tienda es: %d\n", 163, a);
49
50 return 0;
51 }

```

```

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programaci3n\Ejemplos>gcc Dulceria.c -o Dulceria.exe

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programaci3n\Ejemplos>Dulceria.exe

DULCERIA (:
Secciones:
1) Chocolates
2) Paletas
3) Bolsa de Chicless

    1) Chocolates tipo
1. Conejos Turin
2. Bombones de Chocolate
3. Carlos V

    2) Paletas tipo
1. Tutsi
2. Cajeta
3. Coraz3n

    3) Bolsa de Chicless tipo
1. De bola
2. Bubli
3. Canelas

¿Cuántos dulces de tipo 1 hay en la secci3n 1) ? 10
¿Cuántos dulces de tipo 2 hay en la secci3n 1) ? 25
¿Cuántos dulces de tipo 3 hay en la secci3n 1) ? 24
¿Cuántos dulces de tipo 1 hay en la secci3n 2) ? 30
¿Cuántos dulces de tipo 2 hay en la secci3n 2) ? 43
¿Cuántos dulces de tipo 3 hay en la secci3n 2) ? 32
¿Cuántos dulces de tipo 1 hay en la secci3n 3) ? 40
¿Cuántos dulces de tipo 2 hay en la secci3n 3) ? 9
¿Cuántos dulces de tipo 3 hay en la secci3n 3) ? 8

Número de dulces capturado (:
10      25      24
30      43      32
40      9       8

El número total de dulces en la secci3n no.1 es: 59
El número total de dulces en la secci3n no.2 es: 105
El número total de dulces en la secci3n no.3 es: 57

El número total de dulces en la tienda es: 221

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programaci3n\Ejemplos>

```

Conclusiones

Esta práctica se me hizo bastante interesante, había visto anteriormente arreglos, pero no les había entendido del todo bien; sin embargo, con ayuda de esta práctica, los ejercicios y la tarea que se realizó, me quedo más clara la idea de como funcionan, en la tarea me costo un poco de trabajo al principio, pero checando mis apuntes acerca de este tema logré un buen trabajo. Pude entender muy bien los arreglos multidimensionales y también los arreglos unidimensionales. Lo que no me queda aún muy en claro, son los Apuntadores, pero ese tema lo retomaremos después.

Referencias

- Manual de prácticas del Laboratorio de Fundamentos de programación. 06 Enero 2021, de Facultad de Ingeniería Sitio web: <http://lcp02.fi-b.unam.mx/>