



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): Cuevas Antunez Samantha

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

No. de Lista o Brigada: 12

Semestre: Primer semestre

Fecha de entrega: 04/01/2021

Observaciones:

CALIFICACIÓN: _____



Arreglos unidimensionales y multidimensionales

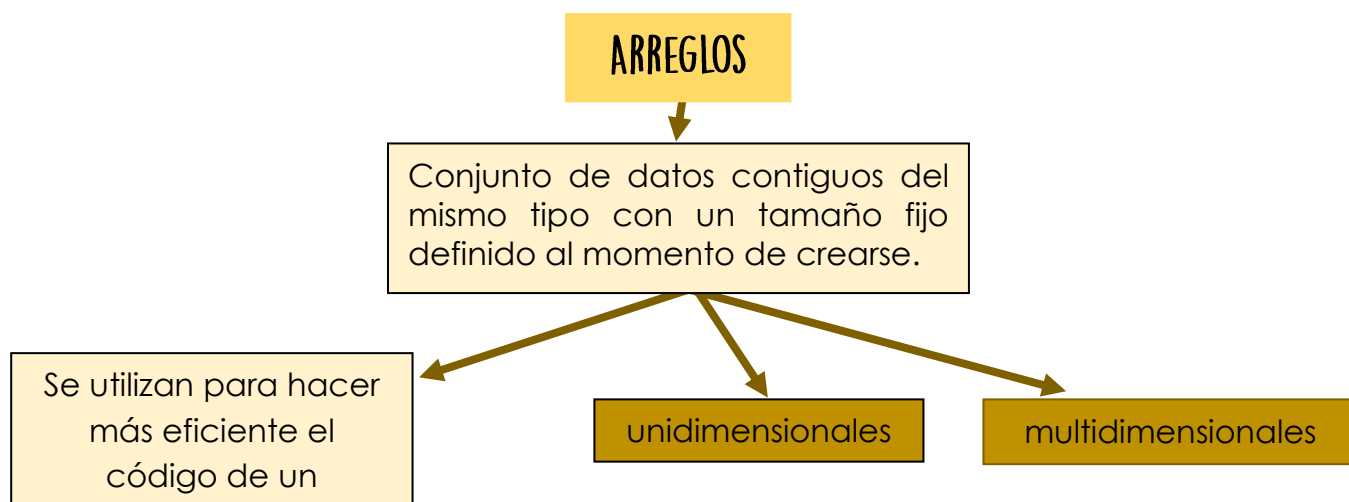
OBJETIVOS

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

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

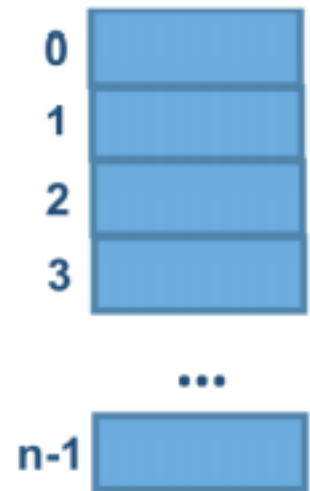


A cada elemento (dato) del arreglo se le asocia una posición particular, el cual se requiere indicar para acceder a un elemento en específico. Esto se logra a través del uso de índices.

ARREGLOS UNIDIMENSIONALES

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

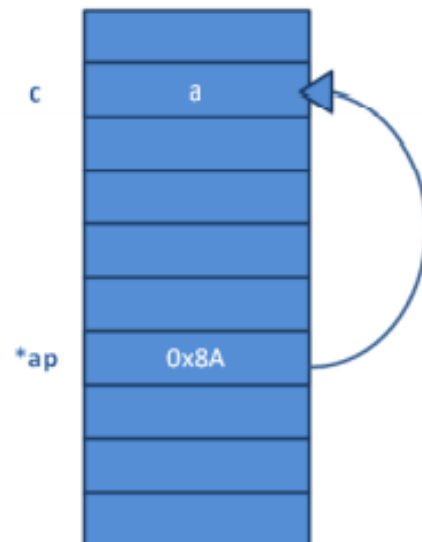
La primera localidad del arreglo corresponde al índice 0 y la última corresponde al índice $n-1$, donde n es el tamaño del arreglo.



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.

Un apuntador almacena la dirección de la memoria de la variable a la que apunta



ESCTRUC- TURA	SINTAXIS	
ARREGLO	tipoDeDato nombre[tamaño]	<p>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.</p>
APUN- TADOR	TipoDeDato *apuntador, variable; apuntador = &variable;	<p>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).</p> <p>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</p>
ARREGLOS MULTIDI- MENSIO- NALES	tipoDato nombre[tamaño][tama ño]...[tamaño];	<p>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.</p> <p>De manera práctica se puede considerar que la primera dimensión</p>

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.

RESULTADOS

Ejemplos.

Arreglo unidimensional while

```
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
7 int main () {
8     #define TAMANO 5
9     int lista[TAMANO] = {10, 8, 5, 8, 7};
10    char aa=160, ae=130, ai=161, ao=162, au=163, sp=168;
11
12    int indice = 0;
13
14    printf("\tLista\n");
15    while (indice < 5)
16    {
17        printf("\nCalificaci%cn del alumno %d es %d",ao, indice+1, lista[indice]);
18        indice += 1; // análogo a indice = indice + 1;
19    }
20
21    printf("\n");
22
23    return 0;
24 }
```

```
C:\Users\La familia\Desktop\Lenguaje C\practicas>gcc
P11auniw.c -o P11auniw.exe

C:\Users\La familia\Desktop\Lenguaje C\practicas>P11a
uniw.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\La familia\Desktop\Lenguaje C\practicas>
```

Arreglo unidimensional 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 ciclo for.
5 */
6
7 int main () {
8     #define TAMANO 5
9     int lista[TAMANO] = {10, 8, 5, 8, 7};
10    char aa=160, ae=130, ai=161, ao=162, au=163, sp=168;
11
12    printf("\tLista\n");
13    for (int indice = 0 ; indice < 5 ; indice++)
14    {
15        printf("\nCalificaci%cn del alumno %d es %d", ao, indice+1, lista[indice]);
16    }
17
18    printf("\n");
19
20    return 0;
21 }
```

```
2 dirs  736,323,444,736 bytes libre
s
C:\Users\La familia\Desktop\Lenguaje C\practicas>g
cc P11aunifor.c -o P11aunifor.exe

C:\Users\La familia\Desktop\Lenguaje C\practicas>P
11aunifor.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\La familia\Desktop\Lenguaje C\practicas>
```

Apuntadores

```
1 #include <stdio.h>
2 /*
3  Este programa crea un apuntador de tipo carácter.
4 */
5
6 int main () {
7     char *ap, c = 'a';
8     char aa=160, ae=130, ai=161, ao=162, au=163, sp=168;
9     ap = &c;
10
11     printf("Carácter: %c\n",aa,*ap);
12     printf("Código ASCII: %d\n",ao,*ap);
13     printf("Dirección de memoria: %d\n",ao,ap);
14
15     return 0;
16 }
17
```

```
Símbolo del sistema
2 dirs 736,325,591,040 bytes libres

C:\Users\La familia\Desktop\Lenguaje C\practicass>gcc P11apunta.c -o P11apunta.exe

C:\Users\La familia\Desktop\Lenguaje C\practicass>P11apunta.exe
Carácter: a
Código ASCII: 97
Dirección de memoria: 6422291

C:\Users\La familia\Desktop\Lenguaje C\practicass>
```

Apuntadores 2.

```
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
7 int main () {
8     int a = 5, b = 10, c[10] = {5, 4, 3, 2, 1, 9, 8, 7, 6, 0};
9     int *apEnt;
10    apEnt = &a;
11
12    printf("a = 5, b = 10, c[10] = {5, 4, 3, 2, 1, 9, 8, 7, 6, 0}\n");
13    printf("apEnt = %a\n");
14
15    b = *apEnt;
16    printf("b = *apEnt \t-> b = %i\n", b);
17
18    b = *apEnt + 1;
19    printf("b = *apEnt + 1 \t-> b = %i\n", b);
20
21    *apEnt = 0;
22    printf("*apEnt = 0 \t-> a = %i\n", a);
23
24    apEnt = &c[0];
25    printf("apEnt = &c[0] \t-> apEnt = %i\n", *apEnt);
26
27    return 0;
28 }
```

```
Símbolo del sistema

C:\Users\La familia\Desktop\Lenguaje C\practicass>gcc P11apunta2.c -o P11apunta2.exe

C:\Users\La familia\Desktop\Lenguaje C\practicass>P11apunta2.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\La familia\Desktop\Lenguaje C\practicass>
```

Apuntadores 3.

```
1 #include <stdio.h>
2 /*
3  Este programa trabaja con aritmética de apuntadores para acceder a los
4  valores de un arreglo.
5 */
6
7 int main () {
8     int arr[] = {5, 4, 3, 2, 1};
9     int *apArr;
10    apArr = arr;
11
12    printf("int arr[] = {5, 4, 3, 2, 1};\n");
13    printf("apArr = &arr[0]\n");
14
15    int x = *apArr;
16    printf("x = *apArr \t -> x = %d\n", x);
17
18    x = *(apArr+1);
19    printf("x = *(apArr+1) \t -> x = %d\n", x);
20
21    x = *(apArr+2);
22    printf("x = *(apArr+1) \t -> x = %d\n", x);
23
24    return 0;
25 }
```

```
Símbolo del sistema

C:\Users\La familia\Desktop\Lenguaje C\practicass>gcc P11apunta3.c -o P11apunta3.exe

C:\Users\La familia\Desktop\Lenguaje C\practicass>P11apunta3.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\La familia\Desktop\Lenguaje C\practicass>
```

Apuntadores en 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
8 int main ()
9 {
10     #define TAMANO 5
11     int lista[TAMANO] = {10, 8, 5, 8, 7};
12     int *ap = lista;
13     char aa=160, ae=130, ai=161, ao=162, au=163, sp=168;
14     printf("\tLista\n");
15     for (int indice = 0 ; indice < 5 ; indice++)
16     {
17         printf("\nCalificaci%cn del alumno %d es %d",ao, indice+1, *(ap+indice));
18     }
19     printf("\n\n");
20     return 0;
21 }
22
23
```

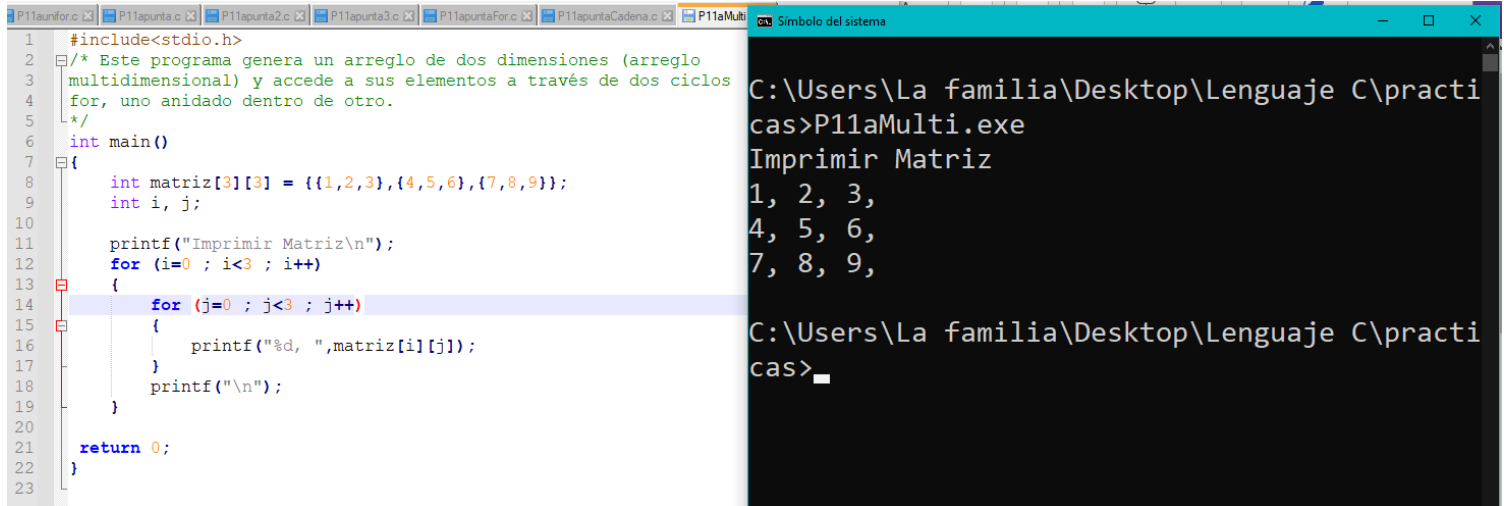
```
Símbolo del sistema
C:\Users\La familia\Desktop\Lenguaje C\pract
icas>P11apuntaFor.exe
Lista
Calificación del alumno 1 es 10
Calificación del alumno 2 es 8
Calificación del alumno 3 es 5
Calificación del alumno 4 es 8
Calificación del alumno 5 es 7
C:\Users\La familia\Desktop\Lenguaje C\pract
icas>
```

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    {
13        printf("%c\n", palabra[i]);
14    }
15    return 0;
16 }
17
```

```
C:\Users\La familia\Desktop\Lenguaje C\practicas>gcc P11apuntaCadena.c -o P11apuntaCadena.exe
C:\Users\La familia\Desktop\Lenguaje C\practicas>P11apuntaCadena.exe
Ingrese una palabra: holi
La palabra ingresada es: holi
h
o
l
i
@
É
@
C:\Users\La familia\Desktop\Lenguaje C\practicas>P11apuntaCadena.exe
Ingrese una palabra: Sofi
La palabra ingresada es: Sofi
S
o
f
i
@
É
@
C:\Users\La familia\Desktop\Lenguaje C\practicas>
```

Arreglos multidimensionales



The image shows a screenshot of a C program and its execution. On the left, a code editor displays the source code for a program that generates a 3x3 matrix and prints it. The code includes a header file, a comment, and a main function with nested loops. On the right, a command prompt window shows the execution of the program, displaying the output of the matrix.

```
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 {
8     int matriz[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
9     int i, j;
10
11     printf("Imprimir Matriz\n");
12     for (i=0 ; i<3 ; i++)
13     {
14         for (j=0 ; j<3 ; j++)
15         {
16             printf("%d, ",matriz[i][j]);
17         }
18         printf("\n");
19     }
20
21     return 0;
22 }
23
```

```
C:\Users\La familia\Desktop\Lenguaje C\practi
cas>P11aMulti.exe
Imprimir Matriz
1, 2, 3,
4, 5, 6,
7, 8, 9,
C:\Users\La familia\Desktop\Lenguaje C\practi
cas>
```

CONCLUSIONES

Como vimos en esta práctica, los arreglos son conjunto de datos del mismo tipo asociados por el nombre de una variable se utilizan principalmente para almacenar datos numéricos. Unas características destacables son que, los elementos que contienen tienen un orden, se pueden acceder a estos mediante su posición y se pueden recorrer usando un ciclo for.

De igual forma se explicaron ciertas características de dichos arreglos. Existen los arreglos unidimensionales y multidimensionales, la primera como nos lo indica, tiene una dimensión mientras la segunda tiene más.

Otro tema relevante que se trató en la práctica fueron los apuntadores, el cual a grandes rasgos señala o apunta a otra variable.

Personalmente los arreglos los relacione con las matrices para lograr entenderlos mejor y como se exploró en la práctica, los arreglos resultan ser muy útiles en inventarios, calificaciones, etc. ya que almacenan grandes conjuntos de datos

REFERENCIAS

Facultad de Ingeniería. (2018, 6 abril). Manual de prácticas de fundamentos de programación MADO-17_FP. Laboratorio de Computación Salas A y B. <http://lcp02.fi-b.unam.mx/>