	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: 03

No de Práctica(s): 11

Integrante(s): Ingrid Ailin Girón Reyes

No. de Equipo de cómputo empleado: No aplica

No. de Lista o Brigada: 18

Semestre: 2021-1

Fecha de entrega: enero2021

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.

Intruducción:

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.

Actividades:

Código (arreglo unidimensional while)

```
#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;
}
```

Código (arreglo unidimensional for)

```
#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;
}
```

Código (apuntadores)

```
#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;
}
```

Código (apuntadores)

```
#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");
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;
}
```

Código (apuntadores)

```
#include <stdio.h>
/*
Este programa trabaja con aritmética de apuntadores para acceder a los
valores de un arreglo.
*/
```

```c
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+1) \t -> x = %d\n", x);
return 0;
}
```

Código (apuntadores en ciclo for)

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

Código (apuntadores en cadenas)

```c
#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;
}
```

Código (arreglos multidimensionales)

```c
#include<stdio.h>
/* Este programa genera un arreglo de dos dimensiones (arreglo
multidimensional) y accede a sus elementos a través de dos ciclos
for, uno anidado dentro de otro.
```

```
*/
int main(){
int matriz[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
int i, j;
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;
}
```

Código (arreglos multidimensionales con apuntadores)

```
#include<stdio.h>
/* Este programa genera un arreglo de dos dimensiones (arreglo
multidimensional) y accede a sus elementos a través de un apuntador
utilizando
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;
}
```

Conclución:

Gracias a la práctica entendí la utilidad e importancia de arreglos cuando elaboramos programas que resuelven problemas que necesitan agrupar datos del mismo tipo, y a trabajar con los arreglos unidimensionales y multidimensionales, que por cierto me parecieron super interesantes e increíbles.