LICENCIATURA EN INFORMATICA

PROGRAMADOR UNIVERSITARIO

INGENIERÍA EN INFORMÁTICA

PROGRAMACIÓN

Unidad 5 - Parte 2: Tipos de datos derivados: arreglos de cadena. Uso de funciones de la biblioteca estándar. Arreglos y funciones.

2024

Lic. Mariela A. Velázquez

Repasemos lo visto

Inicialización de Arreglos

Hay 3 formas de inicializar un arreglo:

- Por omisión.
- Por inicialización explícita.
- En tiempo de ejecución.

Por omisión

```c
#include <stdio.h>
#define TAMA 5
int arre2[TAMA];

int main()
{
    int arre1[TAMA];

    printf("El arreglo declarado de manera global es: ");
    for(int j = 0; j<TAMA; j++)
    {
        printf("\n arre2[%d]=%d", j, arre2[j]);
    }

    printf("\nEl arreglo declarado local a main(): ");
    for(int j = 0; j<TAMA; j++)
    {
        printf("\n arre1[%d]=%d", j, arre1[j]);
    }
    return 0;
}
```

Un arreglo declarado en forma global, se inicializa por omisión (por default), en ceros binarios, a menos que se indique lo contrario.

```
El arreglo declarado de manera global es:
 arre2[0]=0
 arre2[1]=0
 arre2[2]=0
 arre2[3]=0
 arre2[4]=0
El arreglo declarado local a main():
 arre1[0]=8
 arre1[1]=0
 arre1[2]=22
 arre1[3]=0
 arre1[4]=9708464
```

Por inicialización explicita

```c
#include <stdio.h>
#define TAMA 3

int main()
{
    float arre1[]= {78.6,  98.9,  45.7, 34.5, 56.7 } ;
    int arre3[] = {2,4,6};

    printf("El arreglo 3 es: ");
    for(int j = 0; j<TAMA; j++)
    {
        printf("\n arre3[%d]=%d", j, arre3[j]);
    }

    printf("\nEl arreglo 1: ");
    for(int j = 0; j<TAMA; j++)
    {
        printf("\n arre1[%d]=%f", j, arre1[j]);
    }

    return 0;
}
```

En la declaración, se asignan valores, según la siguiente norma: los valores a ser asignados a los elementos del arreglo deben estar encerrados entre llaves y separados por comas.

```
El arreglo 3 es:
 arre3[0]=2
 arre3[1]=4
 arre3[2]=6
El arreglo 1:
 arre1[0]=78.599998
 arre1[1]=98.900002
 arre1[2]=45.700001
```

En tiempo de Ejecución

```c
#include <stdio.h>
#define TAMA 5

int main()
{
    float arre1[TAMA];

    printf("Ingresar 5 numeros reales: ");
    for(int j = 0; j<TAMA; j++)
    {
        printf("\n arre1[%d]", j);
        scanf("%f", &arre1[j]);
    }

    printf("\nEl arreglo 1: ");
    for(int j = 0; j<TAMA; j++)
    {
        printf("\n arre1[%d]=%.2f", j, arre1[j]);
    }

    return 0;
}
```

El caso mas común. Las componentes del arreglo tomarán valores que son leídos desde algún dispositivo de entrada ó sino valores generados por el mismo programa.

```c
    …
    printf("Ingresar 5 numeros reales: ");
    for(int j = 0; j<TAMA; j++)
    {
        arre1[j] = j * 2;
    }

    printf("\nEl arreglo 1: ");
    for(int j = 0; j<TAMA; j++)
    {
        printf("\n arre1[%d]=%d", j, arre1[j]);
    }

    return 0;
}
```

Cadenas –Arreglos de caracteres

- Este es el tipo de arreglo mas popular en código C.
- La ultima celda del vector de caracteres se reserva para el carácter que marca el fin de la cadena, que es el carácter nulo: “\0”.

Inicialización explícita de una cadena

Los valores a asignar deben estar encerrados entre llaves y separados por comas (lista).

```c
#include <stdio.h>
#define MAX 45

int main(void)
{
    char apellido[MAX] = {'P','e','r','e', 'z','\0'};
    // agrega al final el correspondiente cero de terminación al final de la cadena
    char nombre[MAX] = "Marcelo";
    char unt[12]= {'U','n','i','v','e','r','s','i','d','a','d'};

    ...

```

Funciones de biblioteca y arreglos

Función de entrada para cadena de caracteres:

gets(arre): almacena datos ingresados desde stdin a la cadena denominada arre. Un carácter ingresado \n de nueva línea se convierte en un cero de terminación (\0)

Función de salida para cadena de caracteres:

puts(arre): encamina la cadena arre hacia stdout. Un cero de terminación (\0) al final de la cadena se convierte en un carácter de nueva línea (\n).

Ejemplos

```c
#include <stdio.h>
#define TAMA 50
int main()
{
    char nombre[TAMA];

    printf("Ingrese su nombre: ");
    gets(nombre);

    printf("El nombre ingresado es:");
    puts(nombre);

    return 0;
}
```

```
Ingrese su nombre: Ana Sofia
El nombre ingresado es:Ana Sofia
```

Funciones de biblioteca <string.h>

Mencionaremos solo algunas funciones.

Prototipo	Descripción
int strlen(cad1)	Retorna la longitud de cad1
int strcmp(cad1, cad2)	Compara cad1 con cad2, carácter a carácter
char *strcat(cad1, cad2)	Concatena la cad2 al final de la cad1. Retorna cad1.

¿Cómo se trabaja con el tipo arreglo en diseño?

Ejemplo: Realizar un algoritmo que cuenta la cantidad de cada digito que hay en una frase.

```
Algoritmo contadorDigito
ENTRADA: frase: arreglo de caracteres.  MF= '\0'
SALIDA: cont_digito: entero >0
Vble aux: i: entero >=0

A0. Inicializar
A1. LEER(frase)
A2. Mientras (frase_i <> MF)
        Si (esdigito(frase_i)) //función definida por el usuario
            cont_digito ← cont_digito +1
        fin_si
        i ←i+1
    fin_mientras
A3. ESCRIBIR(cont_digito)
A4. PARAR
```

Funciones y Arreglos

```c
#include <stdio.h>
#define TAMA 10

void Inicializar(int arre[TAMA] , int cant);
void CargarArreglo(int arre[TAMA], int cant);

int main()
{
    int arre[TAMA];
    int cant;

    printf("Ingrese la cantidad de Elementos (<10):");
    scanf("%d", &cant);

    Inicializar(arre, cant);
    CargarArreglo(arre, cant);

    for(int i=0; i< cant; i++)
    {
        printf("\n arre[%d] = %d", i, arre[i]);
    }

    return 0;
}
```

```c
void CargarArreglo(int arre[TAMA], int cant)
{
    int i;
    for(i=0; i<cant; i++)
    {  arre[i]= i * 3; }
}

void Inicializar(int arre[TAMA] , int cant)
{
    int i;
    for(i=0; i<cant; i++)
    {   arre[i]=0;     }
}
```

A modo de resumen

Los arreglos, como una unidad, no admiten:

- Asignación directa entre ellos
- Operaciones aritméticas directas
- Comparaciones directas
- Devolución como valor de devolución de una función

Ejercicio

Escribe un algoritmo que realice las siguientes operaciones:

1. Realizar un modulo que cuente y muestre el número de vocales que aparecen en la cadena.
2. Realizar una función que invierta la cadena y muestre.
3. Realizar una función que calcule la longitud de la cadena (Sin hacer uso de la función de biblioteca).

Implemente en Lenguaje C.

Algoritmo: Invertir cadena

Procedimiento invertirCadena (cadena, cadenaInv): arreglo de caracteres, arreglo de caracteres

Vble auxiliar: longitud: entero, i: entero

longitud ← longitudCadena(cadena)

HACER longitud VECES (i=0, …, longitud)

$cadenaInv_i \leftarrow cadena_{(longitud-i-1)}$

FIN HACER

```c
#include <stdio.h>

int main()
{
    printf("Hasta la próxima clase!!\n");

    return 0;
}
```