ARREGLOS EN EL LENGUAJE C

ING. GIANKARIS G. MORENO R., M.SC.

ARREGLOS EN C

- DEFINICIÓN DE UN ARREGLO
- ARREGLOS UNIDIMENSIONALES
- ARREGLOS BIDIMENSIONALES
- PASO DE ARREGLOS A FUNCIONES
- ARREGLOS Y CADENAS DE CARACTERES

DEFINICIÓN DE UN ARREGLO

- Se define como una estructura de datos compuestas, formada por una colección finita de elementos homogéneos, ordenados, que se referencian con un nombre común.

CARACTERISTICAS MAS SOBRESALIENTES DE LOS ARREGLOS.

- Finitud
- Homogeneidad
- Un solo nombre de variable

ARREGLOS

- Es una estructura estática, ya que en tiempo de compilación debe ser conocido su tamaño.
- Los arreglo se accesan de manera directa a través de un índice.

ARREGLOS UNIDIMENSIONALES

- Formato de declaración:

```
tipo_dato       nombre_arreglo [tamaño] ;
```

Tipo de dato: puede ser cualquier tipo de dato estándar o definido por el usuario.

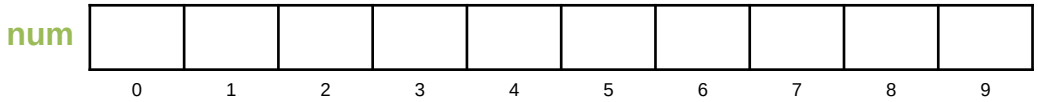
Tamaño: se refiere a la cantidad de elementos que puede almacenar

ARREGLOS UNIDIMENSIONALES

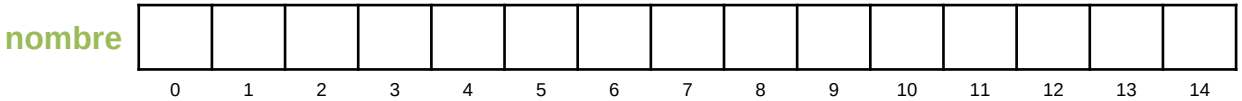
- Ejemplos:

```
int notas[8];
```


```
float num[10];
```




```
char nombre[20];
```



INICIALIZACIÓN DE UN ARREGLO

- Consiste en dar valores iniciales a un arreglo, y se puede realizar a través de una lista de valores, si la lista es más corta que el número de elementos en el arreglo, el resto de los elementos se inicializan a 0. Se puede obviar el tamaño de un arreglo al inicializarlo con una lista de valores, el tamaño será la cantidad de valores que se declaren para el arreglo.

Ejemplos:

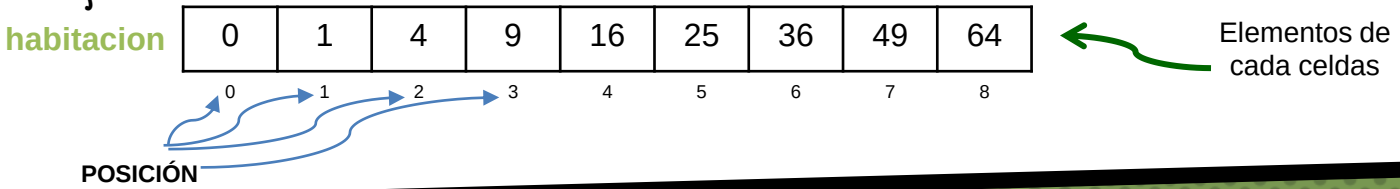
```
float arr[5] = {6.33, 4.56, 8.99, 1.33, 2,54};
double banco[100] = {29.89};
int num[ ] = {2, 3, -5, 7};
```

RECORRIDO DE UN ARREGLO

- Esta operación es propia de los arreglos y permite visitar cada celda del arreglo ya sea de forma parcial o total. Un arreglo se recorre con un índice que se mueve desde el primero al último elemento o desde el último al primero.

Ejemplo:

```
int habitacion[9];
for (i=0; i<9; i++) {
    habitacion [i] = pow (i, 2);
}
```



LECTURA Y ESCRITURA EN UN ARREGLO

- Un arreglo puede recorrerse para almacenar o para imprimir sus elementos.

Ejemplo:

```
float salario[10];
for ( i = 0; i < 10; i++)  {
        scanf ("%f", &salario [i]);
        printf ("%f salario[%i]", salario[i], i);
}
```

```
#include<stdio.h>
#include<conio.h>
void main () {
    int tempe [24], i, menor, mayor, prom;
    prom= 0;

    for (i=0;i<24;i++) {
        printf (“Introduzca la temperatura: ");
        scanf ("%i",&tempe[i]);
    }
    for (i=0;i<24;i++) {
        printf ("Temperatura: %i",tempe[i]);
        prom =  prom +  tempe[i] ;
        if (i== 0)  {
            menor=  tempe[0] ;
            mayor = tempe[0] ;
        }
        if (tempe[i] < menor) {
            menor=  tempe[i] ;
        }
        if (tempe[i] > mayor) {
            mayor=  tempe[i] ;
        }
    }
        prom= prom / 5;
        printf ("\n\n  Mayor temperatura %i", mayor);
        printf ("\n\n  Menor temperatura %i", menor);
        printf ("\n\n  promedio  %i",prom);
}
```

Declaración del arreglo

Lectura del arreglo

Recorrido de un arreglo

Impresión del arreglo

ARREGLOS BIDIMENSIONALES

- Formato de declaración:

```
tipo_dato       nombre_arreglo [tamaño][tamaño] ;
```

Tipo de dato: puede ser cualquier tipo de dato estándar o definido por el usuario.

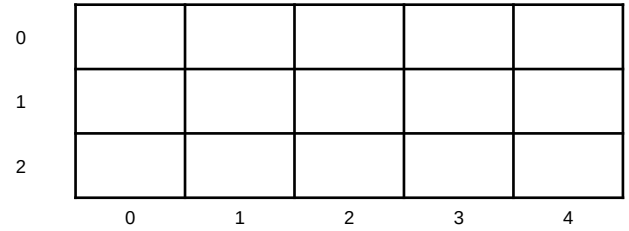
Tamaño: se refiere a la cantidad de elementos que puede almacenar, el primer corchete corresponde a la cantidad de filas, y el segundo corchete a la cantidad de columnas.

ARREGLOS BIDIMENSIONALES

- Ejemplos:

```
int notas[3][5];
```

notas




```
float num[2][10];
```

num

INICIALIZACIÓN DE UN ARREGLO BIDIMENSIONAL

- Al igual que en los arreglos unidimensionales, podemos darle valores iniciales a un arreglo bidimensional mediante una lista como se muestra en el ejemplo.

Ejemplo:

```
int arreglo[2][4]={
        {1, 3, -5, 9},
        {0, 4, 6, 7}
};
```

ó

```
int arreglo[2][4]={ {1, 3, -5, 9},  {0, 4, 6, 7} };
```

RECORRIDO DE UN ARREGLO BIDIMENSIONAL

- Para poder realizar esta operación es necesario utilizar dos estructuras repetitivas anidadas, una para visitar los índices de las filas y otro para visitar los índices de las columnas.

Ejemplo:

```
int pesos[3][5];
for (f= 0; f < 3;  f++) {
        for (c= 0; c < 5; c++) {
                scanf("%d", &pesos [f] [c]);
        }
}
```

LECTURA Y ESCRITURA EN UN ARREGLO BIDIMENSIONAL

- Consiste en recorrer el arreglo ya sea por filas o por columnas para almacenar o para imprimir los valores almacenados en él.

Ejemplo:

```
float pesos[3][5];
    for (f= 0; f < 3;  f++) {
        for (c= 0; c < 5; c++) {
            scanf("%f", &pesos [f] [c]);
            printf("pesos[%i][%i]= %f \n", f, c, pesos[f][c]);
        }
    }
```

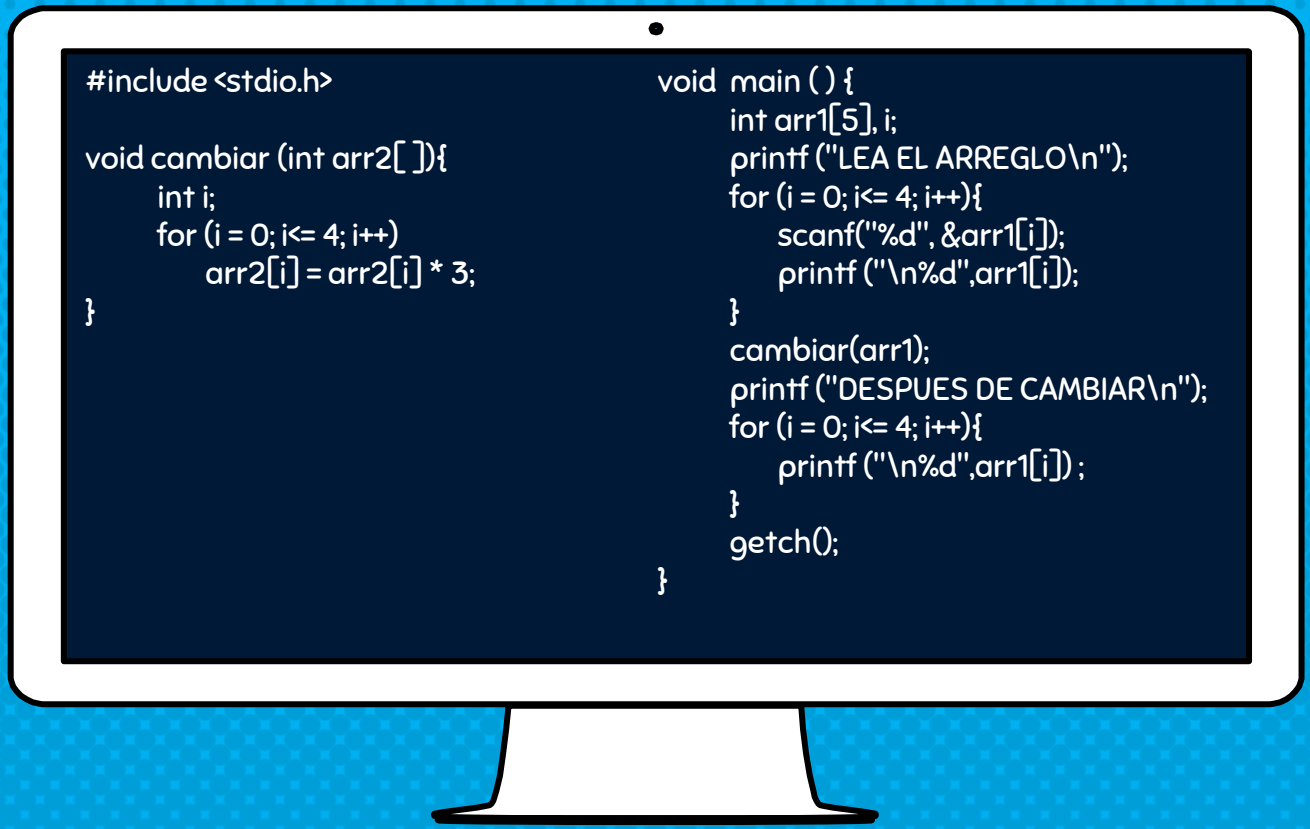
PASO DE ARREGLOS A FUNCIONES

- Los arreglos pasan por referencia a una función, por lo tanto se esta pasando la dirección del primer elemento del mismo, cuando estos sucede los cambios que se realicen sobre los mismos serán retornados desde la función sin necesidad que la función tenga un tipo de retorno
- Los elementos de un arreglo se pueden pasar todos a una función o simplemente ser pasados uno a uno.

Formato para pasar todo el arreglo a la función

```
nombreFunción (nombre_arreglo) ;  //parámetros actuales
void nombreFunción (tipodato varArreglo[ ]) //parámetros formales
```

PASO DE ARREGLO A LA FUNCIÓN



```c
#include <stdio.h>

void cambiar (int arr2[ ]){
        int i;
        for (i = 0; i<= 4; i++)
            arr2[i] = arr2[i] * 3;
}
```

```c
void  main ( ) {
        int arr1[5], i;
        printf ("LEA EL ARREGLO\n");
        for (i = 0; i<= 4; i++){
            scanf("%d", &arr1[i]);
            printf ("\n%d",arr1[i]);
        }
        cambiar(arr1);
        printf ("DESPUES DE CAMBIAR\n");
        for (i = 0; i<= 4; i++){
            printf ("\n%d",arr1[i]) ;
        }
        getch();
}
```

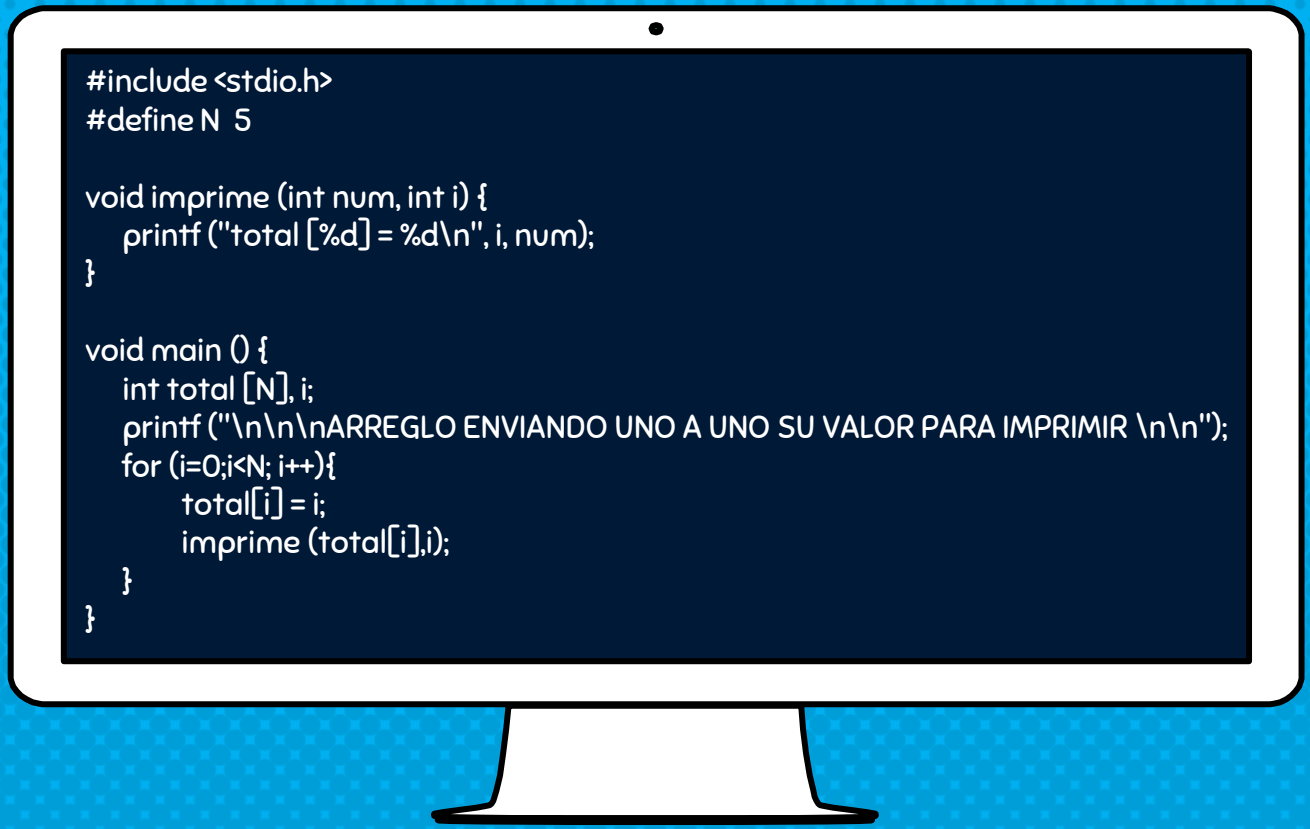
PASO DE ARREGLOS A FUNCIONES

- Cuando se pasa un solo elemento del arreglo, en la lista de parámetros actuales se coloca el arreglo y el índice del valor que se va a enviar, y en la lista de parámetros formales de la función se debe colocar una variable simple y del tipo de dato del arreglo.

Formato para pasar un elemento del arreglo a la función

```
nombreFunción (nombre_arreglo[índice]) ;  //parámetros actuales
void nombreFunción (tipodato var) //parámetros formales
```

PASO DE ARREGLO A LA FUNCIÓN



```
#include <stdio.h>
#define N  5

void imprime (int num, int i) {
    printf ("total [%d] = %d\n", i, num);
}

void main () {
    int total [N], i;
    printf ("\n\n\nARREGLO ENVIANDO UNO A UNO SU VALOR PARA IMPRIMIR \n\n");
    for (i=0;i<N; i++){
            total[i] = i;
            imprime (total[i],i);
    }
}
```

ARREGLOS Y CADENAS DE CARACTERES

- Una cadena de caracteres puede ser representada por un arreglo unidimensional de caracteres.

Formato:

```
char nombre_cadena[longitud];
Ó char nombre_cadena[ ]= "cadena";
```

NOTA: entre los corchetes se debe colocar el largo de la cadena deseada más uno, la posición adicional representa final de la cadena y se representa con "\0".

ARREGLOS Y CADENAS DE CARACTERES

Ejemplo:

char telefono[9]= “233-4400”;

telefono

2	3	3	-	4	4	0	0	\0
0	1	2	3	4	5	6	7	8

- Las cadenas al ser simuladas en arreglos, requieren el uso de funciones para su manipulación (revisar librería string.h)

ARREGLOS Y CADENAS DE CARACTERES

- Para leer una cadena se pueden utilizar dos funciones:

```
gets(varCadena);
Ó scanf("%s", &varCadena);
```

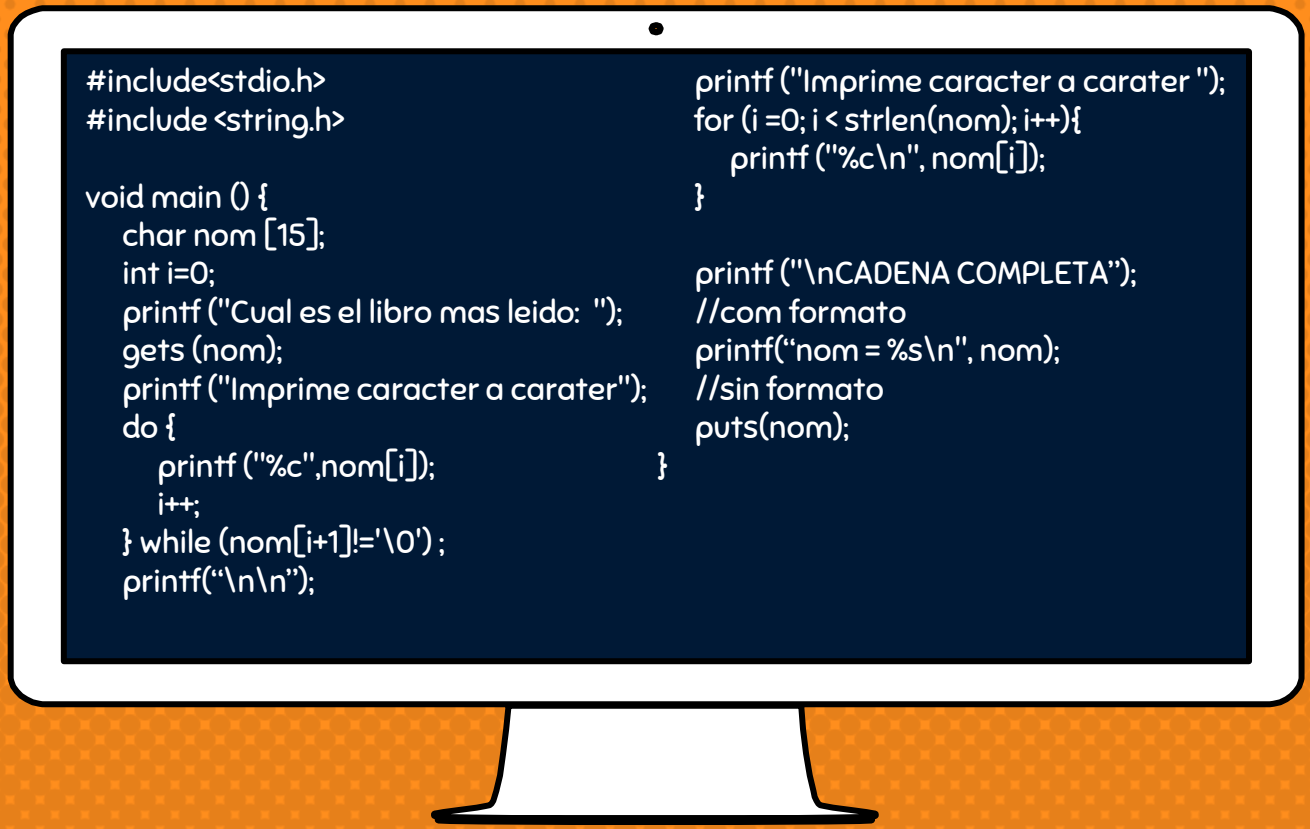
- La función gets( ) permite ingresar espacios en blanco dentro de la cadena.
- La función scanf( ) no permite ingresar espacio en blanco, si se ingresa un espacio en blanco la cadena finaliza.

ARREGLOS Y CADENAS DE CARACTERES

Para imprimir una cadena se pueden realizar de varias formas:

- Carácter por carácter como se hace con un arreglo convencional
- Imprimir completa la cadena mediante la instrucción de salida con formato printf( ) con la cadena de control %s
- Utilizando la instrucción de salida sin formato puts( ) se imprime completa la cadena.

ARREGLO Y CADENA DE CARACTERES



```
#include<stdio.h>
#include <string.h>

void main () {
    char nom [15];
    int i=0;
    printf ("Cual es el libro mas leido:  ");
    gets (nom);
    printf ("Imprime caracter a carater");
    do {
        printf ("%c",nom[i]);
        i++;
    } while (nom[i+1]!='\0') ;
    printf("\n\n");
```

```
printf ("Imprime caracter a carater ");
for (i =0; i < strlen(nom); i++){
    printf ("%c\n", nom[i]);
}

printf ("\nCADENA COMPLETA");
//com formato
printf("nom = %s\n", nom);
//sin formato
puts(nom);
}
```

Preguntas?