PROGRAMACIÓN BÁSICA

PROF. ALMA GONZÁLEZ

APUNTADORES (POINTER)

- Cada variable de un programa tiene un espacio asignado en la memoria, y una dirección para ubicarlo en la memoria, a la cual podemos accesar, usando el operador &. Ej. scanf("%f",&a).
- Los apuntadores en C (C++, y otros lenguajes de programación) son usados para acceder a la memoria y manipular la dirección asociada a las variables. A veces es más sencillo realizar tareas usando los apuntadores que usando las variables.
- Un apuntador es una variable cuyo valor asignado es la dirección en la memoria de otra variable. Debemos declarar las variables de tipo apuntadores de la siguiente forma:

```
int    *ip;    /* apuntador a un entero*/
double *dp;    /* apuntador a un double  */
float  *fp;    /* apuntador a un float */
char   *ch     /* apuntador a un char */
```

ip, dp,fp,ch, son solo nombres de las variables, que pueden ser diferentes.

USO DE APUNTADORES

- Las operaciones mas importantes que realizaremos con apuntadores son:
    - 1) Definir las variables de tipo apuntador
    - 2) Asignar la dirección de otra variable a un apuntador
    - 3) Acceder al valor guardado en la dirección dada por la variable apuntador.

```
#include <stdio.h>
int main () {
   int  var = 20;   /* declaración de la variable */
   int  *ip;        /* declaración de la variable apuntador  */
   ip = &var;  /* asigna la dirección de la variable var al apuntador ip*/

   printf("La dirección de la variable var es: %x\n", (int) &var  );
   printf("Direccion guardada en el apuntador ip: %x\n", (int) ip );
  printf("El valor de *ip: %d\n", *ip );
   return 0;
}
```

USO DE APUNTADORES

- Es buena practica asignar el valor NULL al apuntador, cuando aún no se tiene la dirección a asignar. De esta forma no tendremos apuntadores sin asignar. Ej.

```
#include <stdio.h>
int main () {
   int  *ptr = NULL;
   printf("The value of ptr is : %x\n", ptr  );
   return 0;
}
```

- Es posible checar si un pointer tiene dirección nula, o no:

```
if(ptr)     /* verdadero si p es no NULL */
if(!ptr)    /* verdadero si p es NULL */
```

ARITMETICA CON APUNTADORES

- Un apuntador contiene una dirección en la computadora, la cual tiene un valor numérico. Podemos realizar 4 operaciones aritméticas: ++,- -, +, -

- Ejemplo: Supongamos que la variable **ptr** es un apuntador a una variable entera, que tiene la dirección 1000.

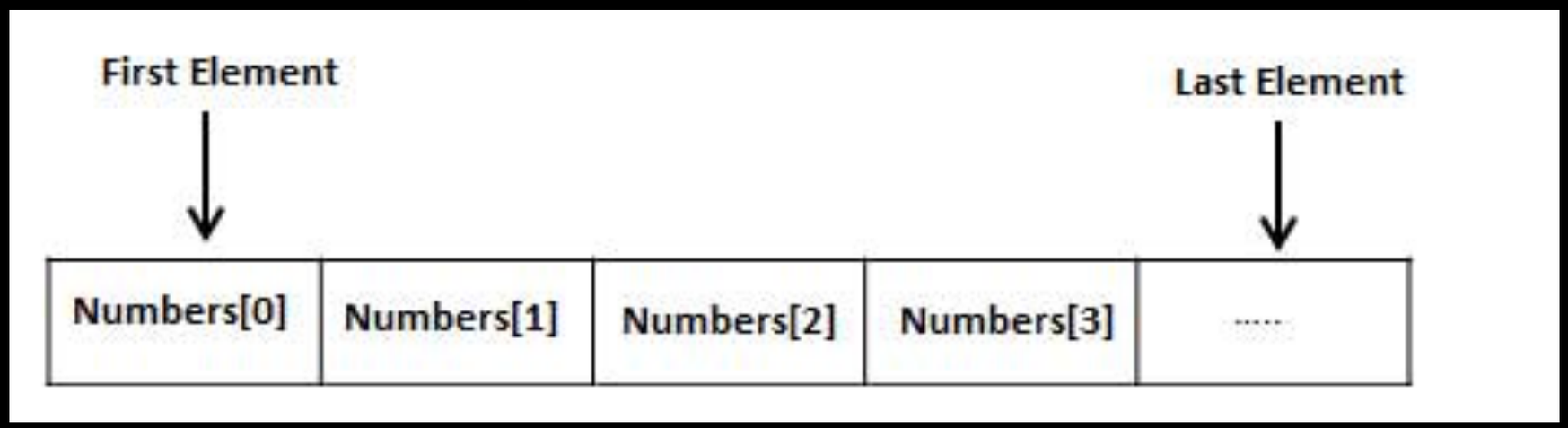
- La operación **ptr++** apuntara entonces a la posición 1004, porque cada vez que le sumamos 1 nos movemos una posición en la dirección de la memoria.

- Ejemplo:

```
#include <stdio.h>
int main () {
   int MAX = 3
   int  var[] = {10, 100, 200};
   int  i, *ptr;
   /* Asignamos la dirección del arreglo al apuntador*/
   ptr = &var;

   for ( i = 0; i < MAX; i++) {
      printf("la dirección de la variable var[%d] = %x\n", i, ptr);
      printf("Valor de la variable var[%d] = %d\n", i, *ptr );
      /* nos movemos a la siguiente posición en la memoria */
      ptr++;
   }
   return 0;
}
```

ARREGLOS (ARRAYS)

- Manejo dinamico de la memoria asignada a un arreglo.
- Usaremos las funciones
    - malloc() #Asigna el numero de bytes indicados y devuelve un apuntador al primer byte del espacio asignado. Es un solo bloque
    - calloc() #Reserva bloques de memoria todos del mismo tamaño y los inicializa a cero.
    - free() #Libera la memoria reservada
    - realloc() #Si la memoria previamente reservada es insuficiente o es demasiada, es posible ajustarla.



EJEMPLO: MALLOC

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int num, i, *ptr, sum = 0;
    ptr = (int*) malloc(num * sizeof(int));  //memoria reservada usando malloc

    printf(“Introduce el numero de elementos: ");
    scanf("%d", &num);

    if(ptr == NULL)
    {
        printf("Error! memoria no reservada.");
        exit(0);
    }

    printf(“Introduce los elementos del arreglo: ");
    for(i = 0; i < num; ++i)
    {
        scanf("%d", ptr + i);
        sum += *(ptr + i);
    }

    printf("Sum = %d", sum);
    free(ptr);
    return 0;
}
```

EJEMPLO: CALLOC

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

int main()
{
    int num, i, *ptr, sum = 0;


    printf(“Introduce el numero de elementos: ");
    scanf("%d", &num);

    ptr = (int*) calloc(num, sizeof(int));

    if(ptr == NULL)
    {
        printf("Error! memoria no reservada");
        exit(0);
    }

    printf(“Introduce los elementos de el arreglo: ");
    for(i = 0; i < num; ++i)
    {
        scanf("%d", ptr + i);
        sum += *(ptr + i);
    }

    printf("Sum = %d", sum);
    free(ptr);
    return 0;
}
```

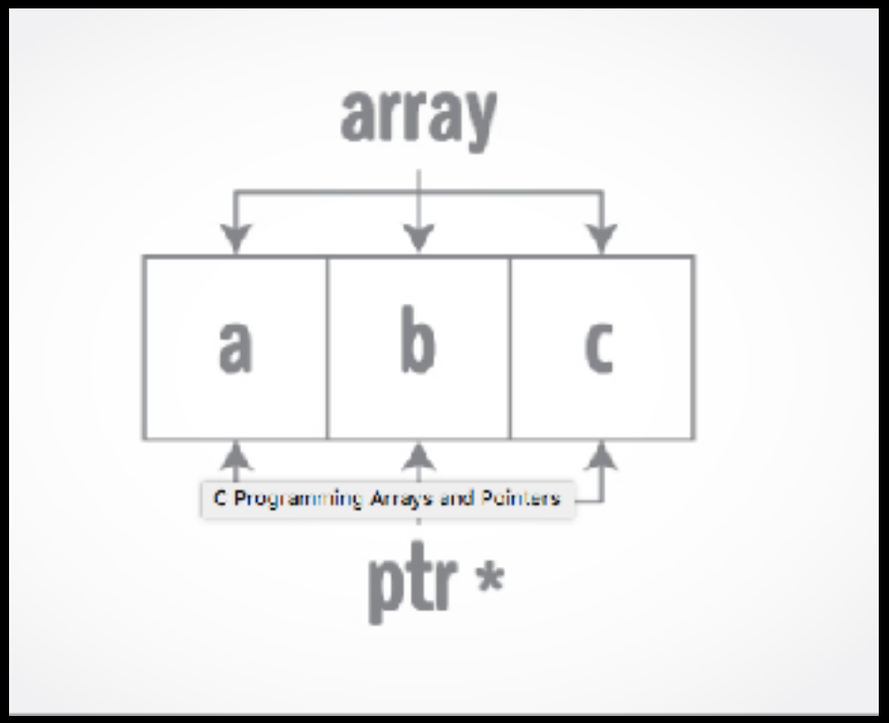
EJEMPLO: REALLOC

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int *ptr, i , n1, n2;
    printf(“Introduce el numero de elementos n1: ");
    scanf("%d", &n1);
     printf("\nIntroduce el nuevo numero de elementos n2: ");
    scanf("%d", &n2);
    ptr = (int*) malloc(n1 * sizeof(int));

    printf(“Dirección de la memoria reservada: ");
     for(i = 0; i < n1; ++i){
      printf("%x\t",ptr + i);
     }
    ptr = realloc(ptr, n2);

    for(i = 0; i < n2; ++i){
         printf("%u\t", ptr + i);
         }

         free(ptr);
    return 0;
}
```

ARREGLO Y APUNTADORES

- Podemos accesar a los elementos de un arreglo a través del arreglo mismo, o bien a través de la dirección en la memoria, usando apuntadores.

Supongamos que se define un arreglo de la forma:
int arr[5];



&arr[0] es equivalente a escribir: arr
y se refiere a la dirección del primer elemento de arr.

arr[0] es equivalente a escribir: *arr (Que es el valor guardado en la dirección a la que apunta arr)

&arr[1] es equivalente a : (arr+1) (La dirección del segundo elemento de arr)

arr[1] es equivalente a *(arr+1) (El valor del elemento 1 del arreglo)

EJEMPLO: ENCONTRAR LA SUMA DE 6 NÚMEROS GUARDADOS EN UN ARREGLO. (DECLARANDO UN ARREGLO Y MANIPULÁNDOLO COMO APUNTADOR)

```
#include <stdio.h>
int main()
{
  int i, num[6],sum = 0;
  printf(“Introduce 6 números enteros:\n");
  for(i = 0; i < 6; ++i)
  {
      // (num + i) es equivalente a  &num[i]
      scanf("%d",(num + i));

      // *(num + i) es equivalente a num[i]
      sum += *(num + i);
  }
  printf("Suma = %d", sum);
  return 0;
}
```

EJEMPLO: ENCONTRAR LA SUMA DE 6 NÚMEROS GUARDADOS EN UN ARREGLO (DECLARANDO UN APUNTADOR Y MANIPULANDO COMO ARREGLO)

```
#include <stdio.h>
int main()
{
  int i, sum = 0;
  int *num;
  printf(“Introduce 6 números enteros:\n”);
  num=(int*) malloc(num * sizeof(int))

  for(i = 0; i < 6; ++i)
  {
      // (num + i) es equivalente a  &num[i]
      scanf(“%d”,&num[i]);

      // *(num + i) es equivalente a num[i]
      sum +=num[i];
  }

  printf("Suma = %d", sum);
  return 0;
}
```