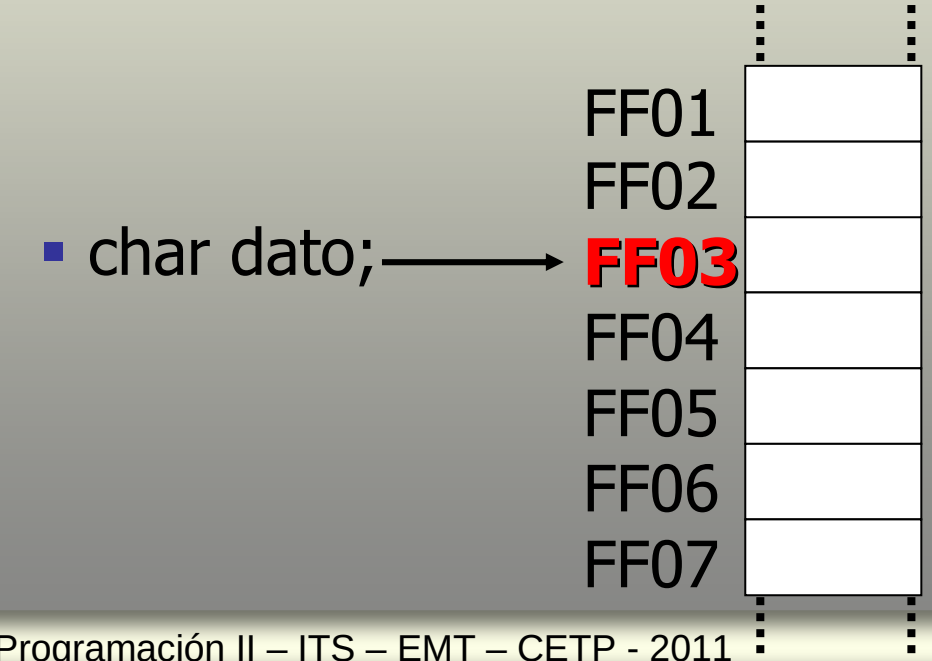
Variables, Vectores o Arrays y Matrices

Variables

Variables

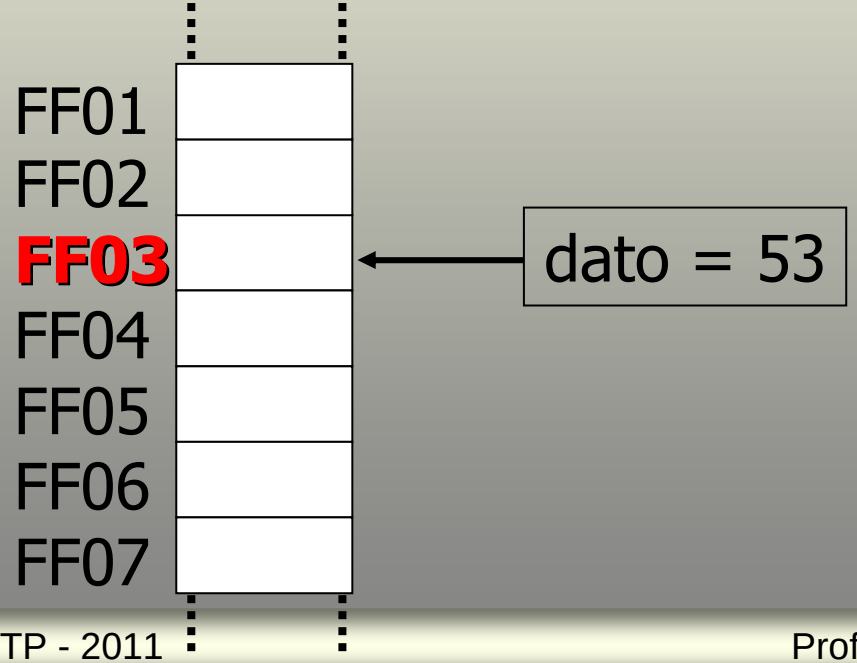
- Variables, **espacio en memoria que almacena un dato, haciendo referencia al mismo por un nombre.**



Variables

- Variables, **espacio en memoria que almacena un dato, haciendo referencia al mismo por un nombre.**

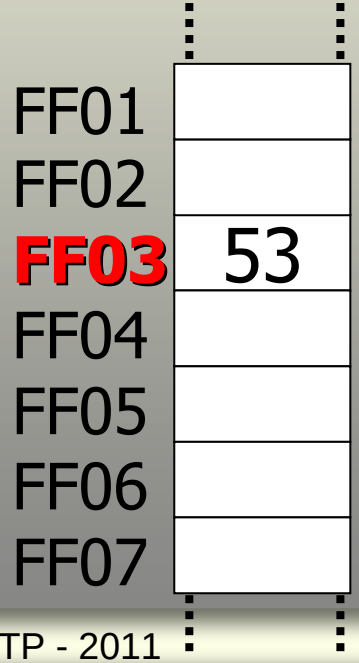
  - char dato;
  - dato = 53;



Variables

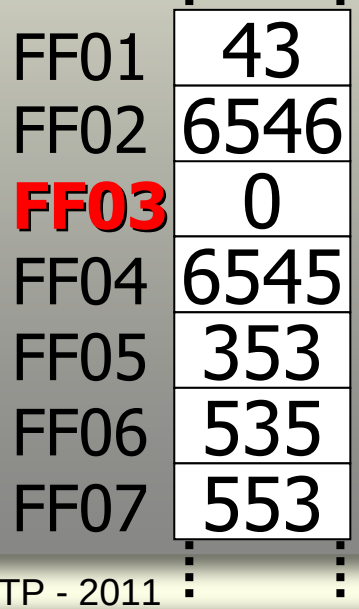
- Variables, **espacio en memoria que almacena un dato, haciendo referencia al mismo por un nombre.**

  - char dato;
  - dato = 53;



Variables

- **La dirección de memoria asignada para almacenar el dato puede tener algún dato anterior (“basura”).**
- **Hay que inicializar las variables.**
  - char dato = 0;



Variables

- **Las variables se definen de un tipo.**
- **Esto determina:**
  - **la cantidad de valores posibles.**
  - **la cantidad de bytes de memoria a utilizar.**
- **char: 1 byte - 2^8 = 256.**
- **int: 2 bytes - 2^{16} = 65,536.**
- **long 4 bytes - 2^{32} = 4,294,967,296.**

Vectores
o
Arrays

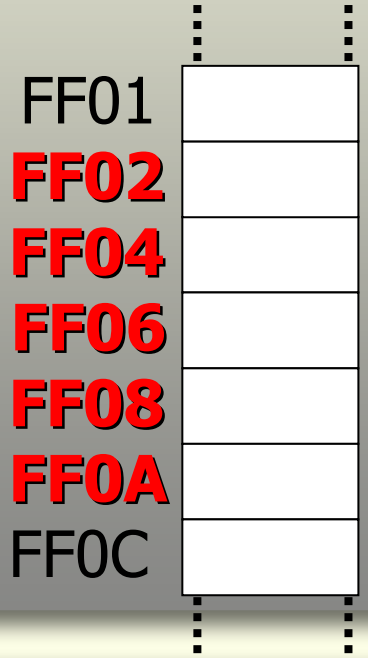
Vectores o Arrays

- Vectores:
  - **conjunto de variables del mismo tipo.**
  - **permiten almacenar varios valores del mismo tipo de dato.**

Vectores o Arrays

- Vectores:
  - **Declaración:**

```
int numeros[5];
```

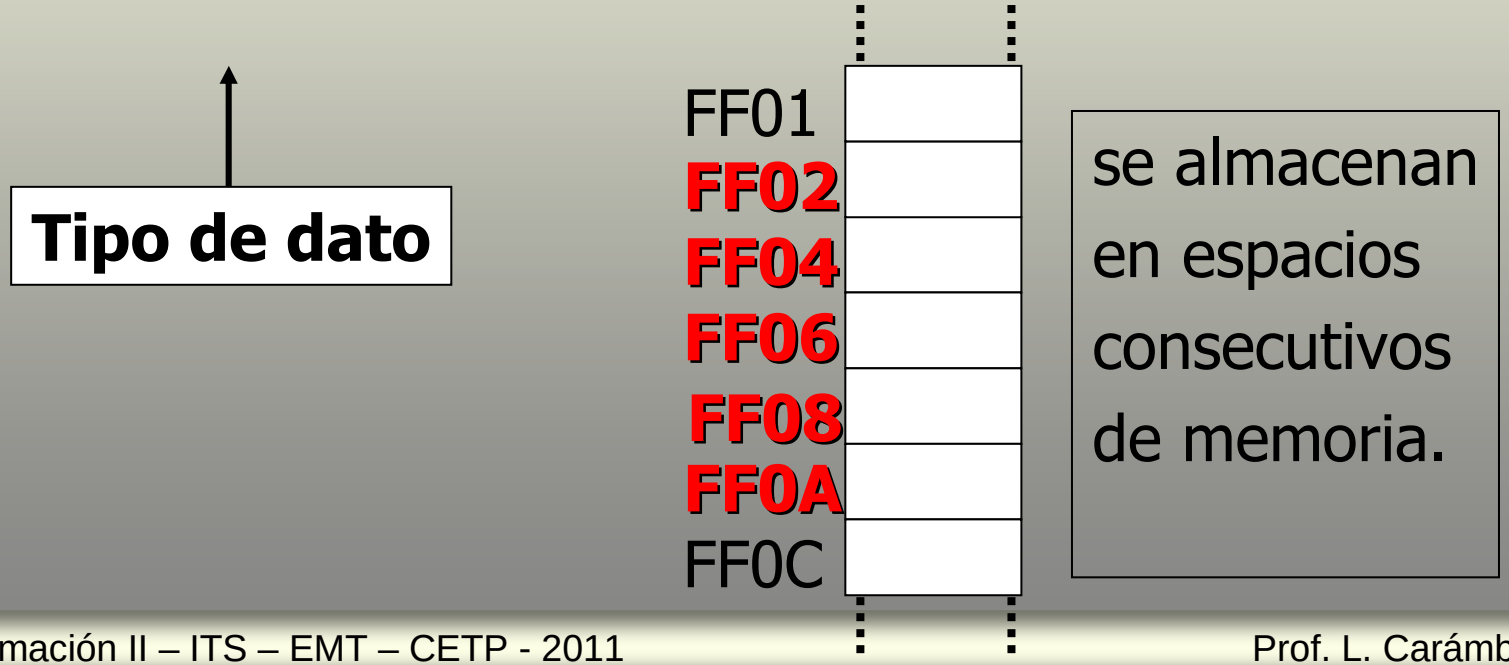


se almacenan en espacios consecutivos de memoria.

Vectores o Arrays

- Vectores:
  - **Declaración:**

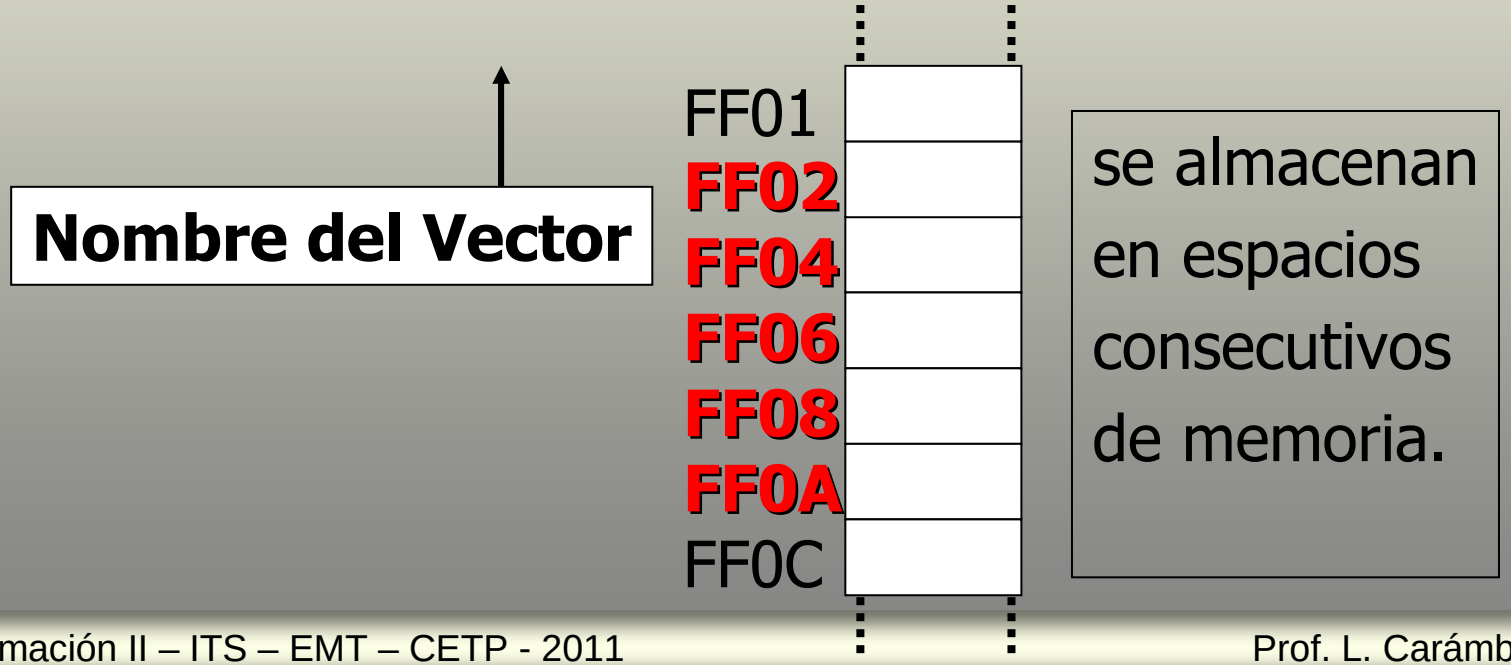
```
int numeros[5];
```



Vectores o Arrays

- Vectores:
  - **Declaración:**

int numeros[5];

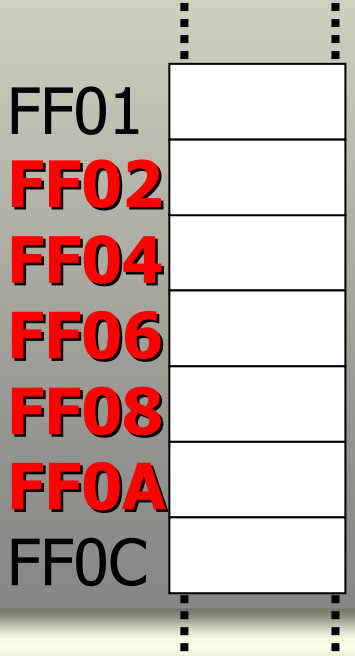


Vectores o Arrays

- Vectores:
  - **Declaración:**

Cantidad de elementos (tamaño del vector)

```
int numeros[5];
```

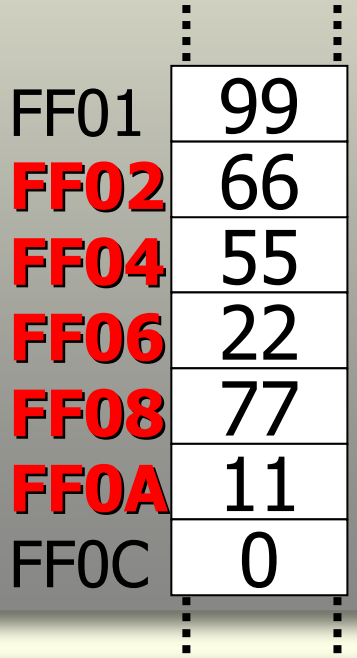


se almacenan en espacios consecutivos de memoria.

Vectores o Arrays

- Vectores:
  - **Inicialización:**

int numeros[5]={53,77,11,99,31};

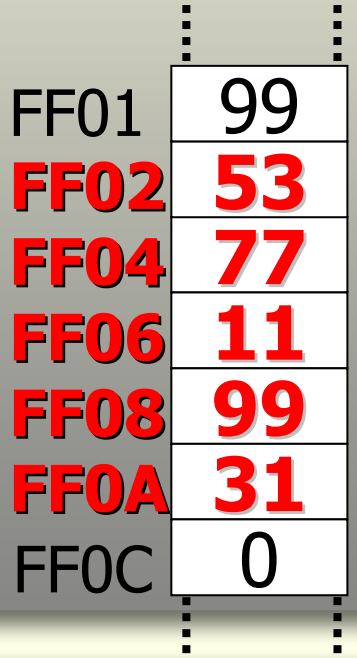


se almacenan en espacios consecutivos de memoria.

Vectores o Arrays

- Vectores:
  - **Inicialización:**

int numeros[5]={53,77,11,99,31};

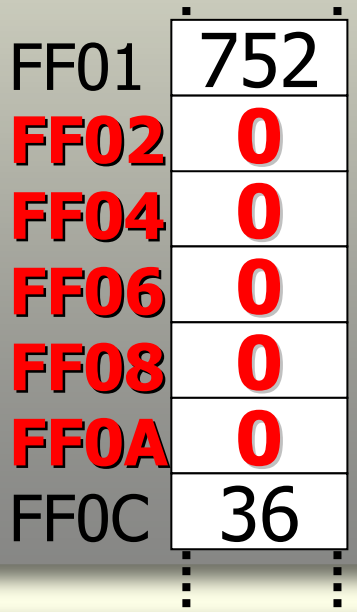


se almacenan en espacios consecutivos de memoria.

Vectores o Arrays

- Vectores:
  - **Inicialización**

```
int i, numeros[5];
for (i = 0; i < 5; i++)
    numeros [ i ]=0;
```

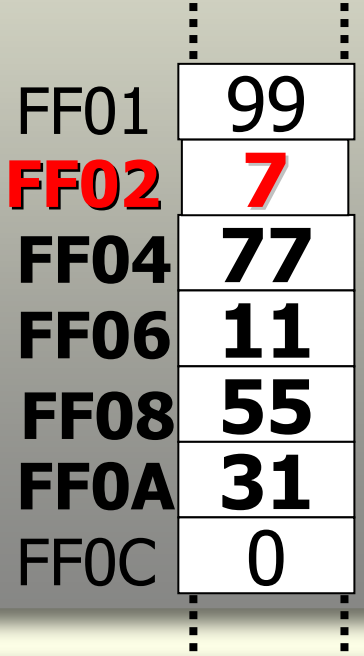


se almacenan en espacios consecutivos de memoria.

Vectores o Arrays

- Vectores:
  - **Utilización: Hay que indicar cual de los 5 valores quiero utilizar a través del índice.**

numeros[0]=7;



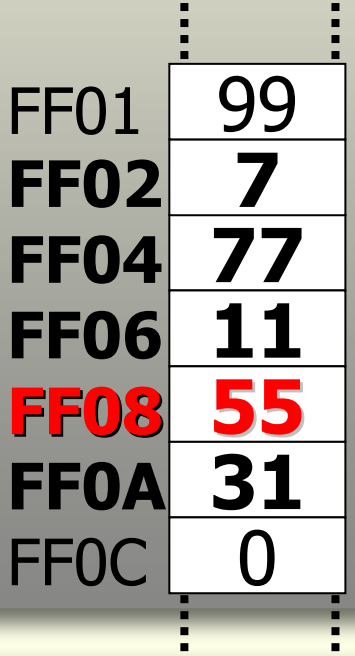
La primera posición es 0 (cero).

Vectores o Arrays

- Vectores:
  - **Utilización: Hay que indicar cual de los 5 valores quiero utilizar a través del índice.**

numeros[0]=7;

numeros[3]=55;



El programador debe controlar que el índice no supere el máximo valor (tamaño – 1)

Vectores o Arrays

- Ejercicio:
  - –Realizar las funciones para el siguiente menú:
    - 1)Inicializar valores (limpiar el vector)
    - 2)Ingresar 5 números enteros.
    - 3)Sumatoria (suma de todos los valores)
    - 4)El promedio
    - 5)El máximo valor
    - 6)El valor mínimo
    - 0)Salir

```
#define LARGO 5
void menu();
void menu_opciones();
void limpiar_vector();
void cargar_vector();
int sumatoria();
int promedio();
void maximo();
void minimo();

int numeros[LARGO];

void main (){
   menu();
   getch();
}
```

```
void menu(){
   char opcion=0;    int suma = 0;
   while (opcion != '0'){
            menu_opciones();
            opcion = getch();
            switch (opcion){
                  case '1':
                        limpiar_vector();
                        break;
                  case '2':
                        cargar_vector();
                        break;
                        ...
```

```
            ...
            case ‘3’:
                  suma = sumatoria();
                  printf(“La Σ= %d”, suma);
                  getch();  break;
            case ‘4’:
                  suma = sumatoria();
                  promedio(suma);      break;
            case ‘5’:
                  maximo(); break;
            case ‘6’:
                  minimo();  break;
            case ‘0’:
                  printf(“\n Terminando...”);
        }
    }
}
```

- Función menu_opciones:

```
void menu_opciones(){
    clrscr();
    printf("1)Inicializar valores\n");
    printf("2)Ingresar 5 números\n");
    printf("3)Sumatoria\n");
    printf("4)El promedio\n");
    printf("5)El máximo valor\n");
    printf("6)El valor mínimo\n\n");
    printf("0)Salir\n\n");
}
```

- Función Limpiar Vector:

```
void limpiar_vector(){
    int i;
    for (i = 0; i<LARGO ; i++)
        vector [ i ]=0;
}
```

- Función Cargar Vector:

```
void cargar_vector(){
    int i;
    for (i = 0; i<LARGO ; i++){
        printf(“Nota: “);
        scanf(“%d”,vector [ i ]);
    }
}
```

- Función Sumatoria:

```
int sumatoria(){
    int i, suma = 0;
    for (i = 0; i<LARGO ; i++)
        suma += vector [ i ];
    return suma;
}
```

- Función Promedio:

```
void promedio(int sum){
    float prom = 0;
    prom =(float) sum/LARGO;
    printf("\n promedio= %.2f ", prom) ;
    getch();
}
```

- Función Máximo:

```
void maximo(){
    int i, max = vector[0];
    for (i = 1; i < LARGO; i++)
        if (vector[i] > max)
            max = vector[i];
    printf( “\n Máximo= %d“ , max);
    getch();
}
```

- Función Mínimo:

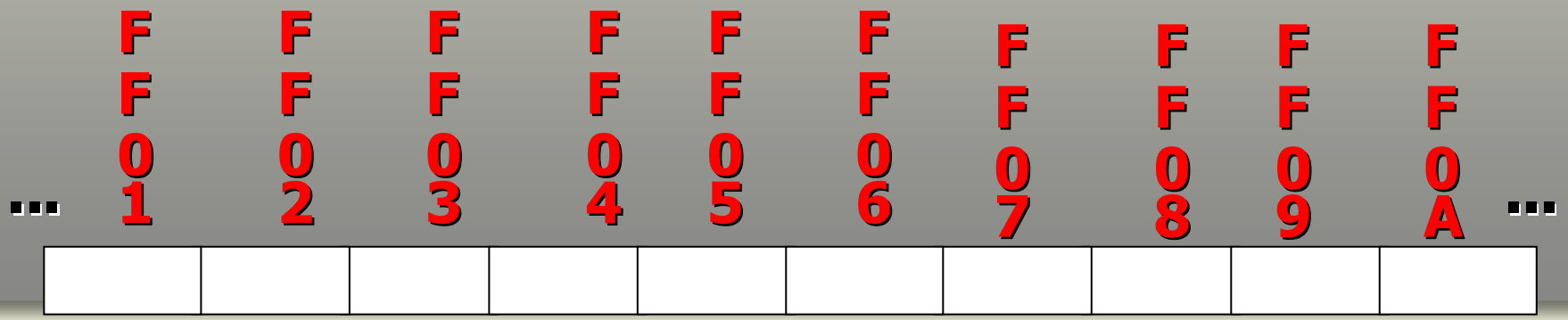
```
void minimo(){
    int i, min = vector[0];
    for (i = 0; i < LARGO; i++)
        if (vector[i] < min)
            min = vector[i];
    printf( “\n Mínimo= %d“ , min);
    getch();
}
```

Cadenas de Caracteres Strings

Cadenas de Caracteres

- Son Vectores de tipo char (caracteres)
  - **Declaración:**

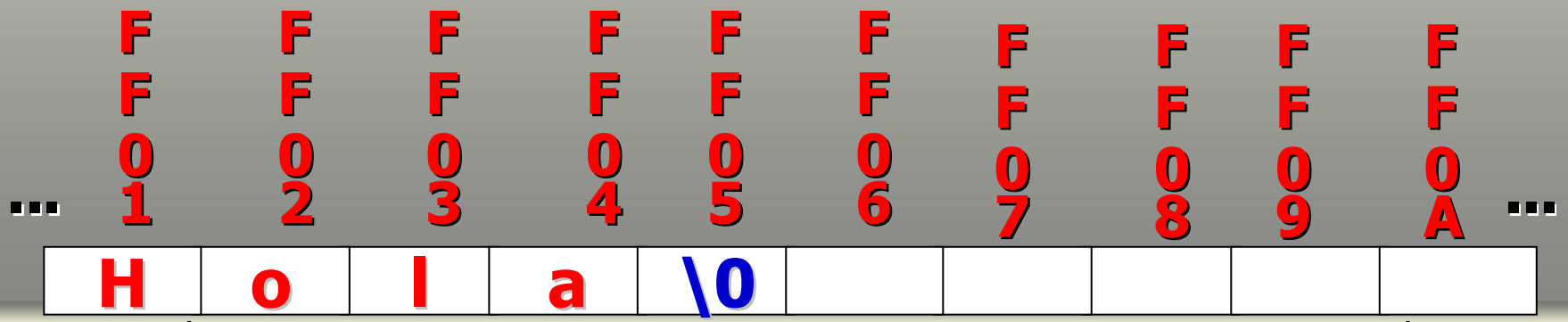
```
char palabra[10];
```



Cadenas de Caracteres

- Son Vectores de tipo char (caracteres)
  - **Inicialización:**

```
char palabra[10] = “Hola”;
```

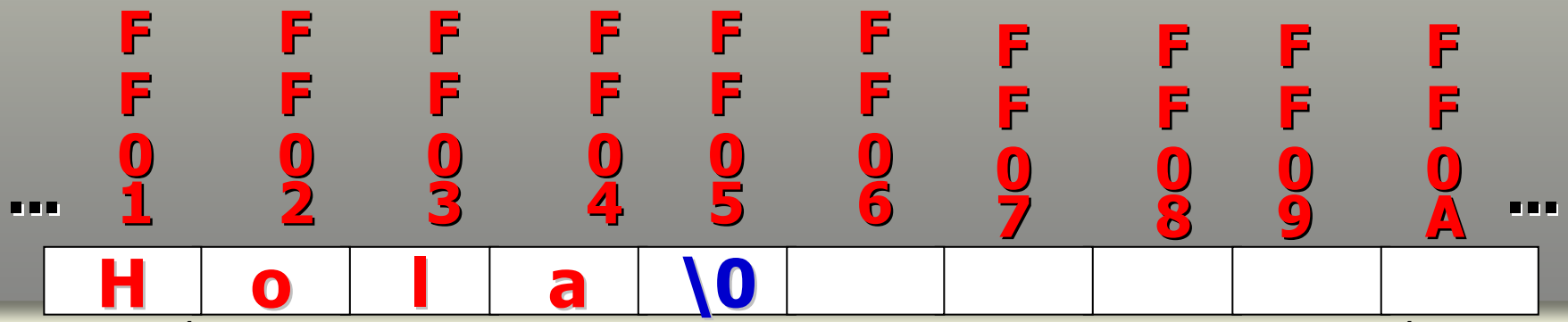


Cadenas de Caracteres

- Son Vectores de tipo char (caracteres)
  - **Inicialización:**

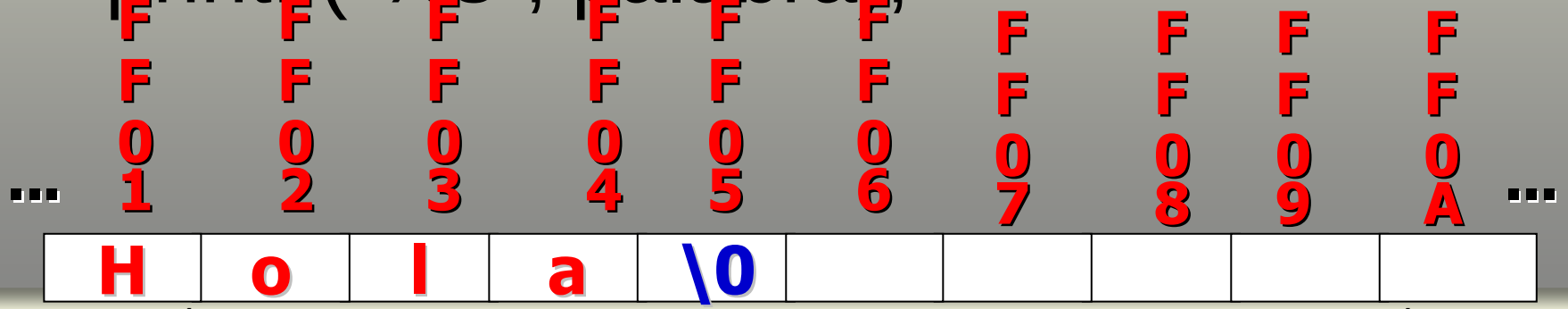
```
char palabra[10] = “Hola”;
```

- El caracter \0 determina el final de la cadena



Cadenas de Caracteres

- Son Vectores de tipo char (caracteres)
  - **Para mostrar un string se utiliza %s en el printf()**
  - **Muestra todos los caracteres hasta que encuentra el caracter \0.**
  - printf (“%s”, palabra);



Cadenas de Caracteres

- Son Vectores de tipo char (caracteres)
  - **No se puede cargar el contenido de un string en otro utilizando el operador =.**
  - **Hay que utilizar la función strcpy de la librería string.h**
  - **char palabra[10] = “Hola”;**
  - **palabra = “Chau”;** 
  - **strcpy(palabra, “Chau”);**

Libreria string.h

- Algunas Funciones:
- strcat
  - **Concatena una cadena al final de otra**
  - **Ejemplo:**
    - **char c_destino[20] = “Hola”;**
    - **char c_origen[20] = “Mundo”;**
  - **strcat (c_destino, c_origen);**
  - **printf(“%s”,c_destino);** ⟶ HolaMundo

Libreria string.h

- Algunas Funciones:
- strchr
  - **busca un caracter dentro de una cadena**
  - **Ejemplo:**

```
char cadena[20] = “Hola Mundo”;
char car = ‘M’;
if (strchr (cadena, car) )
    printf(“%c esta en la cadena %s”, car, cadena);
else
    printf(“%c no esta en la cadena %s”, car, cadena);
```

Libreria string.h

- Algunas Funciones:
- strstr
  - **busca una cadena dentro de otra cadena**
  - **Ejemplo:**

```
char cadena[20] = “Hola Mundo”;
char c_buscar[20] = “nd”;
if (strstr (cadena, c_buscar) )
    printf(“%s esta en la cadena %s”, c_buscar, cadena);
else
    printf(“%s no esta en la cadena %s”, c_buscar, cadena);
```

Libreria string.h

- Algunas Funciones:
- strcmp
  - **compara dos cadenas y devuelve:**
    - **<0 si s1 < s2**
    - **=0 si s1 = s2**
    - **>0 si s1 > s2**
  - **la comparación se realiza caracter a caracter y a través del código ASCII.**

Libreria string.h

- Algunas Funciones:
- strcmp
  - **Ejemplo**

```
char cad1[20] = “progrmacion”;
char cad2[20] = “PROGRAMACION”;
if (strcmp (cad1, cad2) < 0)
     printf(“%s se menor que %s”, cad1, cad2);
else
   if (strcmp (cad1, cad2) > 0)
     printf(“%s se mayor que %s”, cad1, cad2);
   else
     printf(“%s y %s son iguales”, cad1, cad2);
```

Libreria string.h

- Algunas Funciones:
- strlwr
  - **Convierte una cadena a minúsculas**
  - **Ejemplo:**

```
char cadena[20]="HoLa MuNdO";
strlwr(cadena);
printf("%s",cadena);
```

Libreria string.h

- Algunas Funciones:
- strupr
  - **Convierte una cadena a mayúsculas**
  - **Ejemplo:**

```
char cadena[20]="HoLa MuNdO";
strupr(cadena);
printf("%s",cadena);
```

Libreria string.h

- Algunas Funciones:
- strlen
  - **Devuelve el largo en caracteres de una cadena, hasta el caracter '\0'.**
  - **Ejemplo:**

```
char cadena[20]="HoLa MuNdO";
int largo;
largo = strlen(cadena);
printf("%i",largo);
```

→ 10

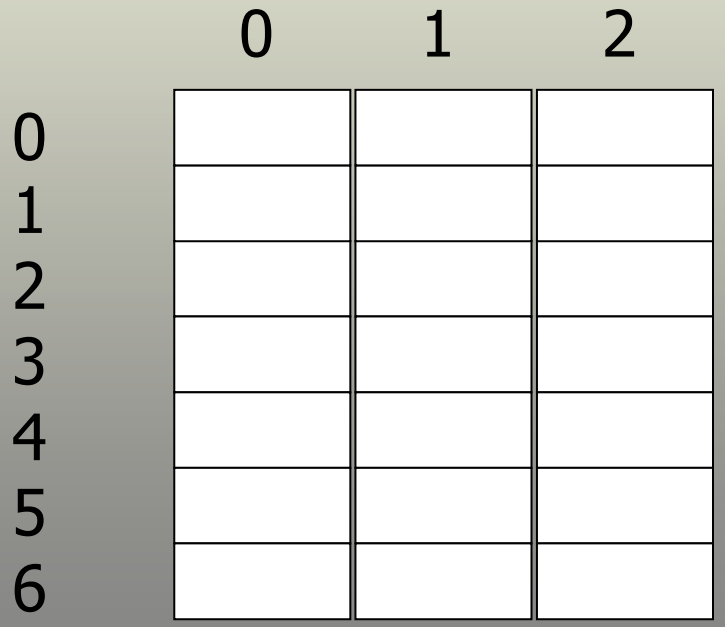
Matrices

Matrices

- Es un array pero de 2 o más dimensiones**.**
- **Al igual que los arrays:**
  - **Se almacenan en posiciones consecutivas de la memoria.**
  - **Cada posición de la matriz es definida por un conjunto de índices, uno para cada dimensión.**
  - **En cada posición se puede almacenar un dato.**
  - **Todos los datos corresponden al mismo tipo de dato**

Matrices

- **Ejemplo matriz de 2 dimensiones (7x3)**
  - **Cada posición de la matriz esta definida por dos índices (fila y columna).**

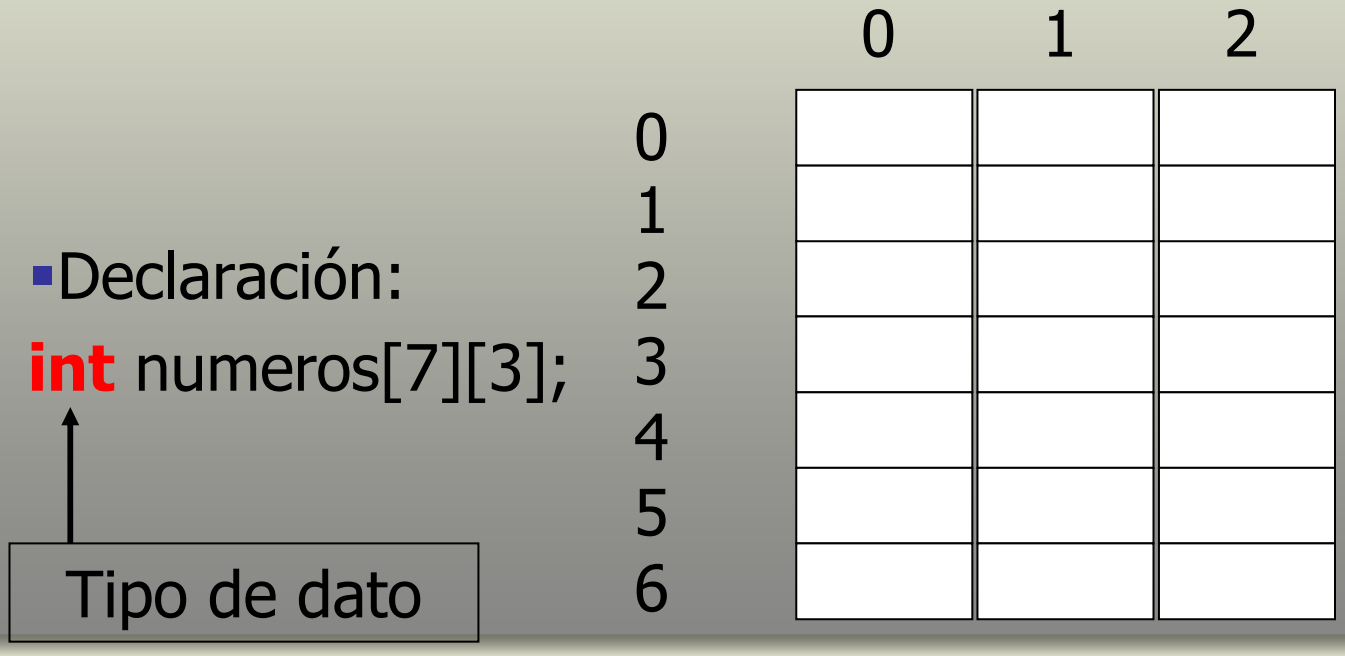


- Declaración:

int numeros[7][3];

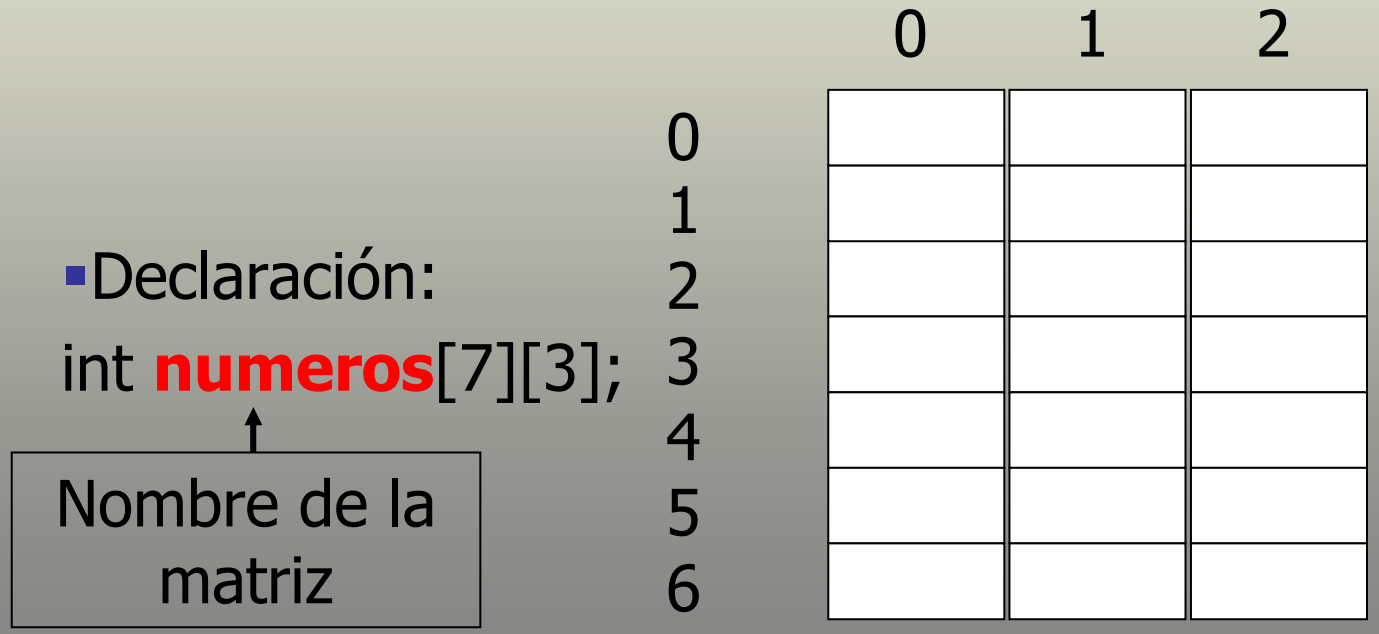
Matrices

- **Ejemplo matriz de 2 dimensiones (7x3)**
  - **Cada posición de la matriz esta definida por dos índices (fila y columna).**



Matrices

- **Ejemplo matriz de 2 dimensiones (7x3)**
  - **Cada posición de la matriz esta definida por dos índices (fila y columna).**



Matrices

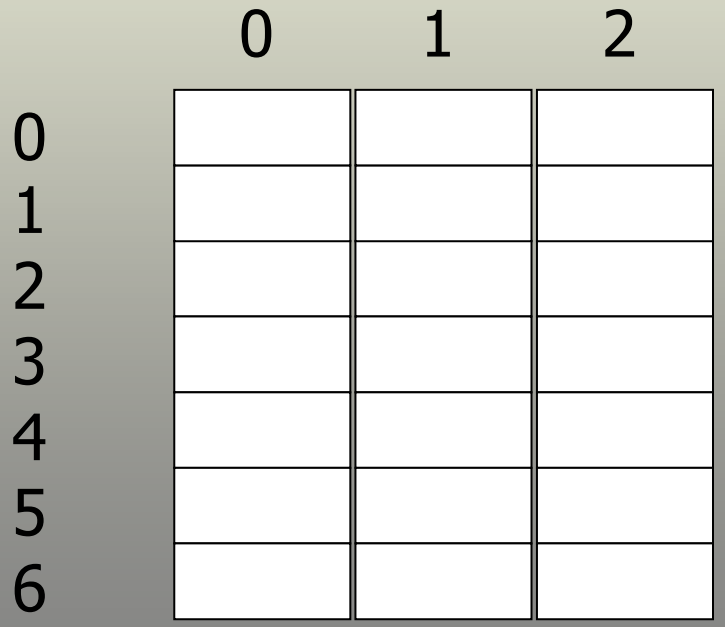
- **Ejemplo matriz de 2 dimensiones (7x3)**
  - **Cada posición de la matriz esta definida por dos índices (fila y columna).**

Tamaño segunda dimensión

- Declaración:

int **numeros**[7][3];

Tamaño primer dimensión

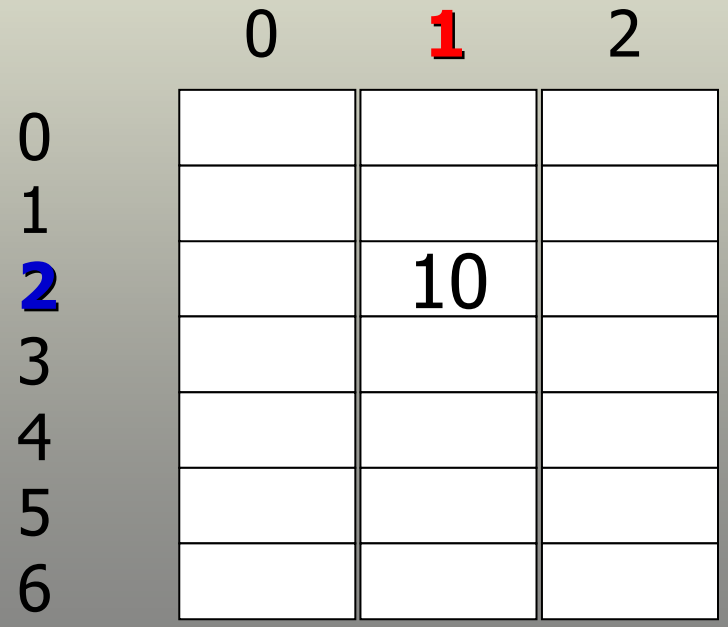


Matrices

- **Ejemplo matriz de 2 dimensiones (7x3)**
  - **Cada posición de la matriz esta definida por dos índices (fila y columna)**

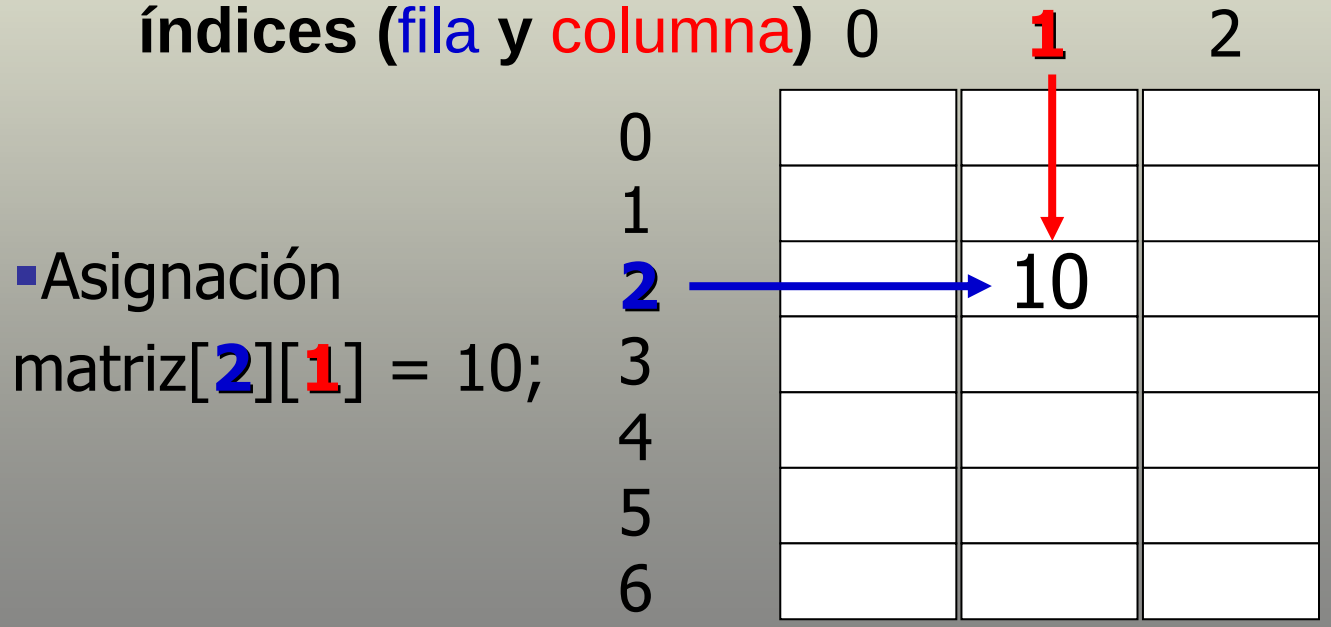
- Asignación

matriz[**2**][**1**] = 10;



Matrices

- Es un array pero de 2 o más dimensiones.
- **Ejemplo matriz de 2 dimensiones (7x3)**
  - **Cada posición de la matriz esta definida por dos índices (fila y columna)**



- Asignación

matriz[2][1] = 10;

Matrices

- **Ejemplo matriz de 2 dimensiones (7x3)**
  - **Al igual que los arrays hay que inicializar la matriz.**

```
int f, c, numeros[7][3] ;
for (f = 0; f < 7; f++)
    for ( c = 0; c < 3; c++)
        numeros [ f ][ c ]=0;
```