Vectores
conceptos

- También se llaman **arrays** de una dimensión.
- Es un conjunto de valores **de un mismo tipo** (llamados elementos del vector).
- Cada elemento del vector se identifica por el **nombre del vector** y su posición (**índice**).
- El índice del primer elemento es el **cero**.
- Los elementos del vector se encuentran en posiciones de memoria contiguas.

Vectores
Declaración e inicialización

DECLARACIÓN: tipo nombre_vector [tamaño];

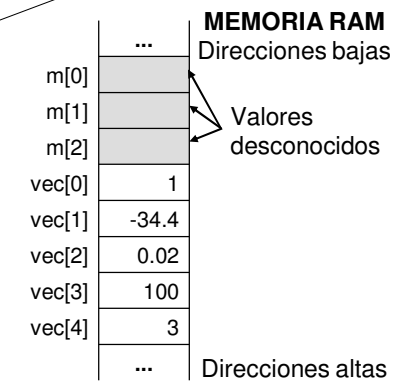
INICIALIZACIÓN: tipo nombre_vector [tamaño] = {valor1, valor2, ... };

El tamaño en la inicialización es **opcional**. Si no se especifica, el vector tendrá tantos elementos como valores.

EJEMPLOS:

```
float m[3];
float vec[] = {1, -34.4, 0.02, 100, 3};
```

Indice	0	1	2	3	4
Valor	1	-34.4	0.02	100	3
	vec[0]	vec[1]	vec[2]	vec[3]	vec[4]



Vectores
Acceso a los elementos (1)

```
nombre_vector[ índice ]
```

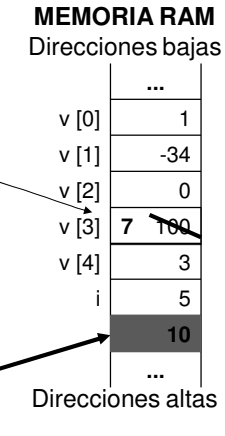
Cualquier expresión que de cómo resultado un valor entero

EJEMPLOS:

```
int v[5] = {1, -34, 0, 100, 3};
int i = 5;
v[i-2] = 3 * v[v[2]] + 4;
v[6] = 10;
if (v[4] > 50) {
      ...
}
scanf( "%d", &(v[0]) );
```

v[v[2]]
v[0]

¡PELIGRO!: Se está accediendo fuera del vector



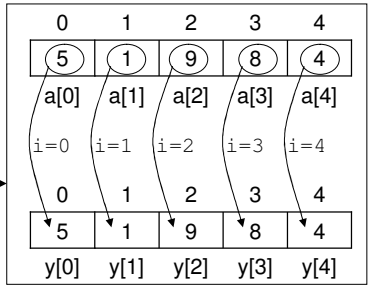
Vectores
Acceso a los elementos (2)

- No se puede operar sobre todos los elementos del vector a la vez
- Sólo podemos acceder a los elementos de uno en uno
- Ejemplo:

```
#define TAM 5
void main() {
        float x[TAM], y[TAM];
        float a[TAM] = {5,1,9,8,4};
        float b[TAM] = {5,4,3,2,1};
        int i;

NO      y = a;
        x = a + b;

SI      for (i=0; i<TAM; i++)  {
              y[i] = a[i];
              x[i] = a[i] + b[i];
        }
}
```

La variable **i** toma valores desde 0 hasta **TAM – 1.**

Para cada valor de **i**, se accede a un elemento diferente de los vectores.



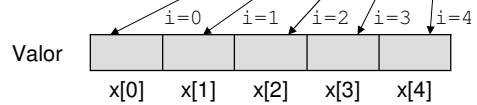
Vectores
Ejemplo

Programa que lee los valores del un vector desde teclado.

```
#define TAM 5
void main() {
       int i;
       float x[TAM];
       for (i=0; i<TAM; i++) {
              printf( "Introduce el elemento %d: ", i );
              scanf( "%f", &(x[i]) );
       }
}
```

La variable **i** toma valores desde **0** hasta **TAM - 1**

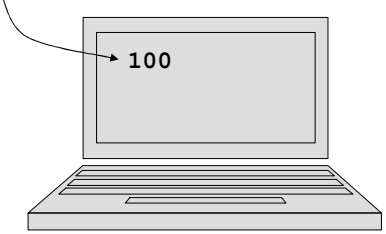
En cada iteración del bucle, se lee por teclado un valor y se guarda en el elemento **i**



Vectores
Parámetros formales de tipo vector (1)

- Cuando usamos el nombre de un vector sin corchetes, obtenemos la dirección de memoria donde comienza el vector.
- Ejemplo:

```
main() {
    int v[5] = {5,4,3,2,1};
    printf( "%X\n", v );
}
```



MEMORIA RAM

Dirección	Contenido	
	...	
0x100	5	v[0]
0x104	4	v[1]
0x108	3	v[2]
0x10C	2	v[3]
0x110	1	v[4]
	...	

Vectores
Parámetros formales de tipo vector (2)

- Parámetros formales de tipo vector
  - No se reserva espacio en memoria para el vector
  - No es necesario indicar el tamaño del vector
  - La dirección de memoria donde estará el vector se le pasa en la llamada a la función
- Ejemplos de declaración:

```
void funcion1( int vec[10] );
void funcion2( float m[] );
```

El parámetro `m` es un vector de `float`'s que se encontrará en la dirección de memoria que se le pase en la llamada a `funcion2`.

El parámetro `vec` es un vector de enteros que se encontrará en la dirección de memoria que se le pase en la llamada a `funcion1`.

Vectores
Parámetros formales de tipo vector: Ejemplo 1a

```
#define TAM 10
void ceros( float p[] );

main() {
       float v1[TAM], v2[15];
       ceros(v1);
       ceros(v2);
}

void ceros( float p[] ) {
       int i;
       for (i=0; i<TAM; i++)
              p[i] = 0;
}
```

MEMORIA RAM

Dirección	Contenido	
	...	
0x100		v1[0]
0x104		v1[1]
...	...	...
0x124		v1[9]
	...	
0x2F0		v2[0]
0x2F4		v2[1]
...	...	...
0x328		v2[14]
	...	

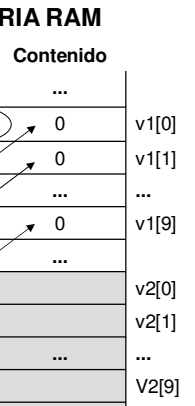
v1 **v2**

Las casillas grises contienen valores desconocidos

Vectores
Parámetros formales de tipo vector: Ejemplo 1b

```
#define TAM 10
void ceros( float p[] );

main() {
       float v1[TAM], v2[15];
       ceros(v1);
       ceros(v2);
}

void ceros( float p[] ) {
       int i;
       for (i=0; i<TAM; i++)
              p[i] = 0;
}
```

0x100

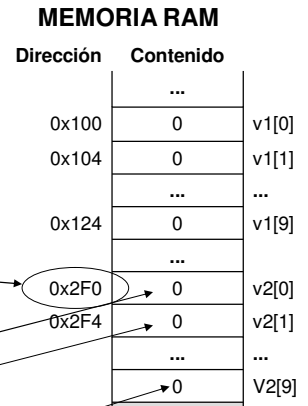


El parámetro **p** toma la dirección de memoria de **v1**

Vectores
Parámetros formales de tipo vector: Ejemplo 1c

```
#define TAM 10
void ceros( float p[] );

main() {
       float v1[TAM], v2[15];
       ceros(v1);
       ceros(v2);
}

void ceros( float p[] ) {
       int i;
       for (i=0; i<TAM; i++)
              p[i] = 0;
}
```

0x2F0



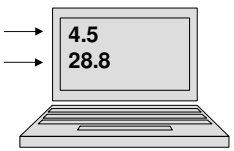
El parámetro **p** toma la dirección de memoria de **v2**

Vectores
Parámetros formales de tipo vector: Ejemplo 2

Función para calcular la media de los elementos de un vector de `double`'s.

```
double media( double vec[], int tam ) {
       int i;
       double total = 0;
       for (i=0; i<tam; i++)
              total += vec[i];
       return total/tam;

}

void main() {
       double valores1[10] = {2,2,3,4,3,6,5,8,9,3};
       double valores2[5] = {10,11,23,100,0};
       printf( "%lf\n", media( valores1, 10 ) );
       printf( "%lf\n", media( valores2, 5 ) );

}
```

El parámetro `tam` contiene el tamaño del vector



Vectores
Parámetros formales de tipo vector: Ejemplo 3

Función para calcular el valor de un polinomio cuyos coeficientes están en un vector de `double`'s.

```
double polinomio( double x, double a[], int tam ) {
       int i;
       double total = 0;
       double pot_x = 1;
       for (i=0; i<tam; i++) {
              total += a[i] * pot_x;
              pot_x = pot_x * x;
       }
       return total;

}

void main() {
       double c[10] = {2,0,-3};
       printf( "%lf\n", polinomio( 3.4, c, 3 ) );

}
```

El vector a contiene los coeficientes del polinomio:
$a_0 + a_1 \cdot x + a_2 \cdot x^2 + ... + a_{tam-1} \cdot x^{tam-1}$

calculamos el polinomio $2 - 3 \cdot x^2$ para x=3.4

Cadenas de caracteres (1)

- Carácter y cadena de caracteres:
  - `'a'` es un carácter (tipo **`char`**)
  - `"Hola mundo"` es una cadena de caracteres
  - Internamente (en memoria) todas las cadenas de caracteres terminan con el **carácter nulo** (carácter `'\0'`, código `ASCII 0`)

Carácter	H	o	l	a		m	u	n	d	o	\0
ASCII	72	111	108	97	32	109	117	110	100	111	0

  - No debe confundirse el carácter nulo `'\0'` (código `ASCII 0`) con el carácter cero `'0'` (código `ASCII 48`)
- Equivalencia entre el tipo **`char`** y el código ASCII, ejemplos:

```
char c;

c = '0';          equivalente      c = 48;
c = '9';                           c = 57;
c = 'a';                           c = 97;
c = '\0';                          c = 0;
c = '0' + 1;                       c = 48 + 1;
```

CARACTER	Código ASCII
'\0' NULO	0
...	...
'0'	48
'1'	49
...	...
'9'	57
...	...

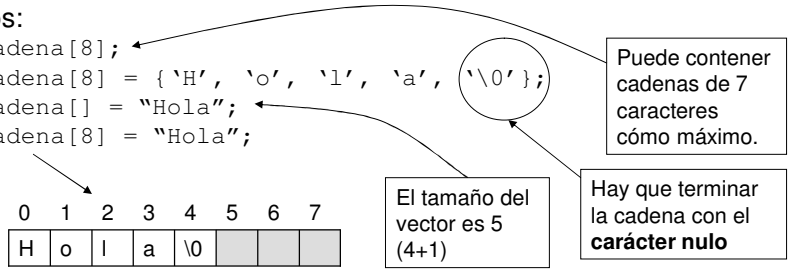
Cadenas de caracteres (2)

- Las cadenas de caracteres se almacenan en **vectores de caracteres**.
- Declaración e inicialización
  - Se declaran y se inicializan como cualquier vector
  - Existe una inicialización especial para cadenas de caracteres:

```
char nombre_cadena[ tamaño ] = "cadena de caracteres";
```

- Ejemplos:

```
char cadena[8];
char cadena[8] = {'H', 'o', 'l', 'a', '\0'};
char cadena[] = "Hola";
char cadena[8] = "Hola";
```



Cadenas de caracteres
Ejemplo 1

Rellenar una cadena con las 6 primeras letras de alfabeto.

```
#define TAM 7
void main() {
       char cadena[TAM];
       int i;
       for (i=0; i<TAM-1; i++) {
              cadena[i] = 'A' + i;
       }
       cadena[i] = '\0';
       printf( "%s\n", cadena );

}
```

i toma valores desde **0** hasta **5**

Índice	0	1	2	3	4	5	6
Carácter	A	B	C	D	E	F	
ASCII	65	66	67	68	69	70	

Cuando termina el bucle, **i** vale **6**. Añadimos el carácter nulo para finalizar la cadena

Índice	0	1	2	3	4	5	6
Carácter	A	B	C	D	E	F	\0
ASCII	65	66	67	68	69	70	0

Cadenas de caracteres
Ejemplo 2a

Función que convierte las letras de una cadena de minúsculas a mayúsculas.

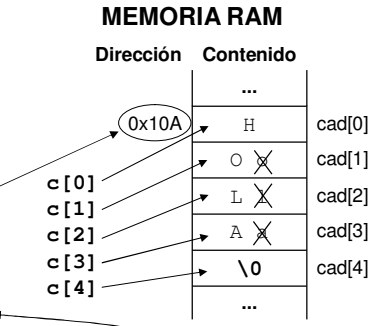
```
void mayusculas( char cadena[] );
void main() {
       char cad[] = "Hola";
       mayusculas( cad );
       printf( "%s\n", cad );

}

void mayusculas( char c [] ) {
       for (i=0; c[i] != '\0'; i++) {
              if (c[i] >= 'a' && c[i] <= 'z') {
                     c[i] = 'A' + c[i] - 'a';
              }
       }
}
```

MEMORIA RAM

Dirección	Contenido	
	...	
0x10A	H	cad[0]
0x10B	o	cad[1]
0x10C	l	cad[2]
0x10D	a	cad[3]
0x10E	\0	cad[4]
	...	

Cadenas de caracteres
Ejemplo 2b

Función que convierte las letras de una cadena de minúsculas a mayúsculas.




```
void mayusculas( char cadena[] );
void main() {
       char cad[] = "Hola";
       mayusculas( cad );          0x10A
       printf( "%s\n", cad );
}
void mayusculas( char c[] ) {
       for (i=0; c[i] != '\0'; i++) {
              if (c[i] >= 'a' && c[i] <= 'z') {
                     c[i] = 'A' + c[i] - 'a';
              }
       }
}
```

El bucle **for** termina cuando el elemento **i** de la cadena sea el carácter nulo.

Cadenas de caracteres
Funciones de la librería estándar

Función	Cometido	Valor devuelto
`char *gets (char cad[]);`	Lee de teclado una cadena de caracteres y la guarda en `cad`.	Se estudiará en el apartado de punteros
`int puts (char cad[]);`	Imprime en pantalla la cadena de caracteres `cad`. Después de la cadena imprime un salto de línea.	El número de caracteres escritos
`int strlen (char cad[]);`	Devuelve la longitud de la cadena de caracteres contenida en `cad`.	Longitud de la cadena
`char *strcat (char cad1[], char cad2[]);`	Añade la cadena `cad2` al final de la cadena `cad1`.	Se estudiará en el apartado de punteros
`int strcmp (char cad1[], char cad2[]);`	Compara alfabéticamente `cad1` con `cad2` (usando el código ASCII).	Un valor <0 si cad1 < cad2 Cero si cad1 == cad2 Un valor >0 si cad1 > cad2
`char *strcpy (char cad1[], char cad2[]);`	Copia la cadena de caracteres `cad2` en `cad1`.	Se estudiará en el apartado de punteros

Cadenas de caracteres
Funciones de la librería estándar: Ejemplo

```
void main() {
       char cad1[80] = "mundo";
       char cad2[100];
       printf( "cad1: %s\n", cad1 );
       strcpy( cad2, "hola " );
       puts( cad2 );
       strcat( cad2, cad1 );
       printf( "cad2: %s\n", cad2 );
       printf( "%d\n", strlen( cad2 ) );
       if (strcmp( cad1, cad2 ) < 0)
              puts( "cad1 < cad2" );
       else if(strcmp( cad1, cad2 ) == 0)
              puts( "cad1 == cad2" );
       else if (strcmp( cad1, cad2 ) > 0)
              puts( "cad1 > cad2" );
       gets( cad2 );
}
```