Vectores y matrices en C

Mg. Ing. Facundo S. Larosa

Informática I

Instituto Universitario Aeronáutico
Universidad de la Defensa Nacional (UNDEF)

Ejemplo disparador

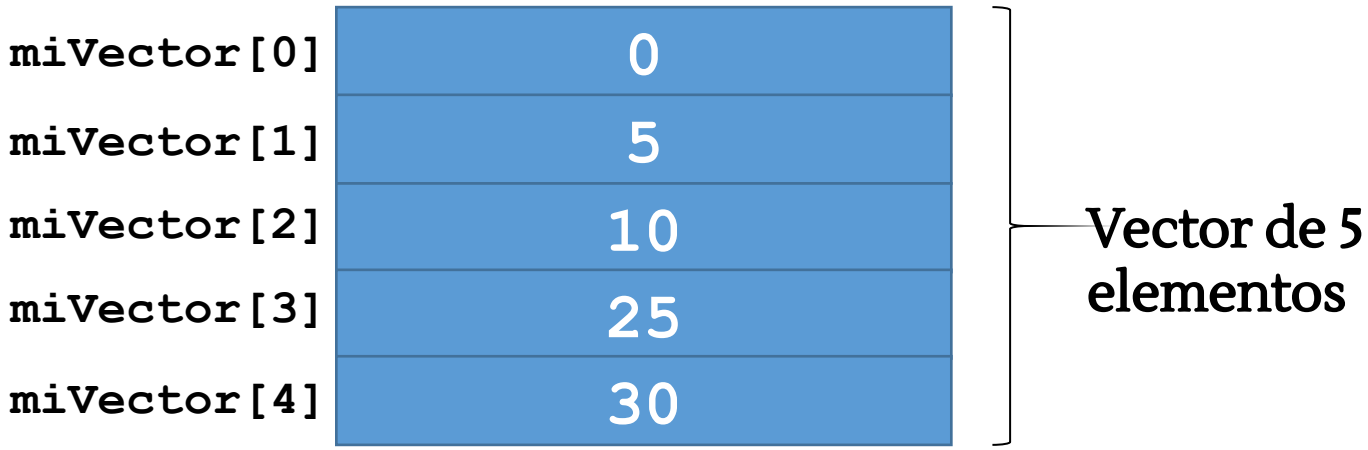
Diseñe un programa en el cual se ingresan 100 muestras de temperaturas y se imprime el valor máximo, el valor mínimo, el promedio y las temperaturas ordenadas de menor a mayor.

¿Cómo hacemos?

Vector: Definición

Es un arreglo unidimiensional de variables del mismo tipo, cada una identificadas con un subíndice numérico.

```
//Declaración de un vector inicializado
int miVector[5]={0,5,10,25,30};
```



A programar

1) Ingrese 10 valores numéricos que representarán muestras de temperatura provenientes de una máquina. Luego, imprimir solamente aquellos que superan un determinado umbral.

2) Ingrese 30 valores numéricos que representan las notas de un curso de Informática I. Finalmente imprima aquellos valores que superan la media del curso separados de aquellos que están por debajo.

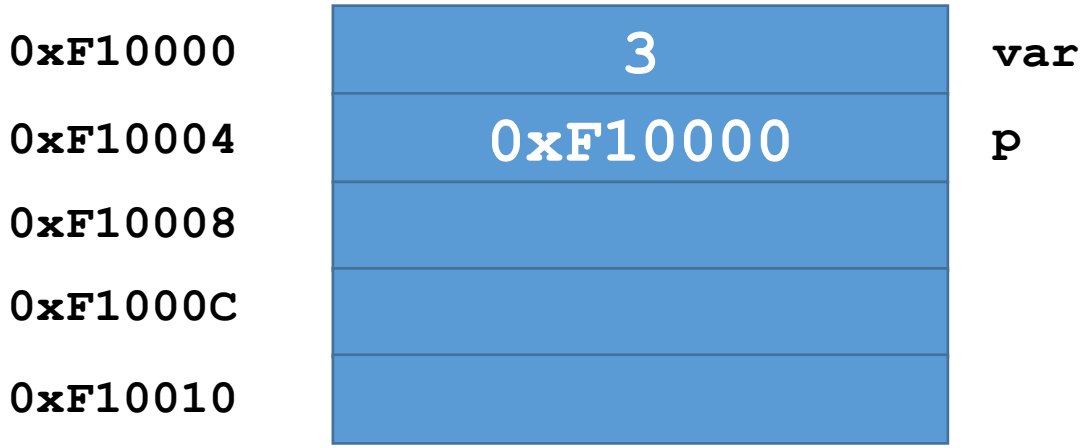
Punteros

Un puntero es un tipo de variable destinado a guardar una dirección de memoria.

```
//Declaración de un puntero
int * p;
int var=3;
//Asignación de un valor a un puntero
p=&var;
printf("El contenido de var es: %d",*p);
//*p significa "el contenido de lo apuntado por p"
```

Punteros

```
//Declaración de un puntero
int * p;
int var=3;
//Asignación de un valor a un puntero
p=&var;
//&var se lee "la dirección de memoria de var"
printf("El contenido de var es: %d",*p);
//*p se lee "el contenido de lo apuntado por p"
```



Fragmento del mapa de memoria

Se dice que "**p** apunta a **var**"

El contenido de lo apuntado por p equivale a var

***p ≡ var**

Punteros
Pasaje de variables a funciones por valor

```
//¿Qué imprime el siguiente ejemplo?

void incrementar (int var)
{
   var=var+1;
}

int main (void)
{
   int var=5;
   incrementar(var);
   printf("var=%d",var);
   return 0;
}
```

Punteros
Pasaje de variables a funciones por valor

```
/* El programa imprime "var=5" ya que la variable se pasa por valor, es
decir, que la variable original del main no se modifica */

void incrementar (int var)
{
   var=var+1;
}

int main (void)
{
   int var=5;
   incrementar(var);
   printf("var=%d",var);
   return 0;
}
```

Punteros
Pasaje de variables a funciones por referencia

```
//¿Qué imprime el siguiente ejemplo?

void incrementar (int *punt)
{
   *punt=*punt+1;
}

int main (void)
{
   int var=5;
   incrementar(&var);
   printf("var=%d",var);
   return 0;
}
```

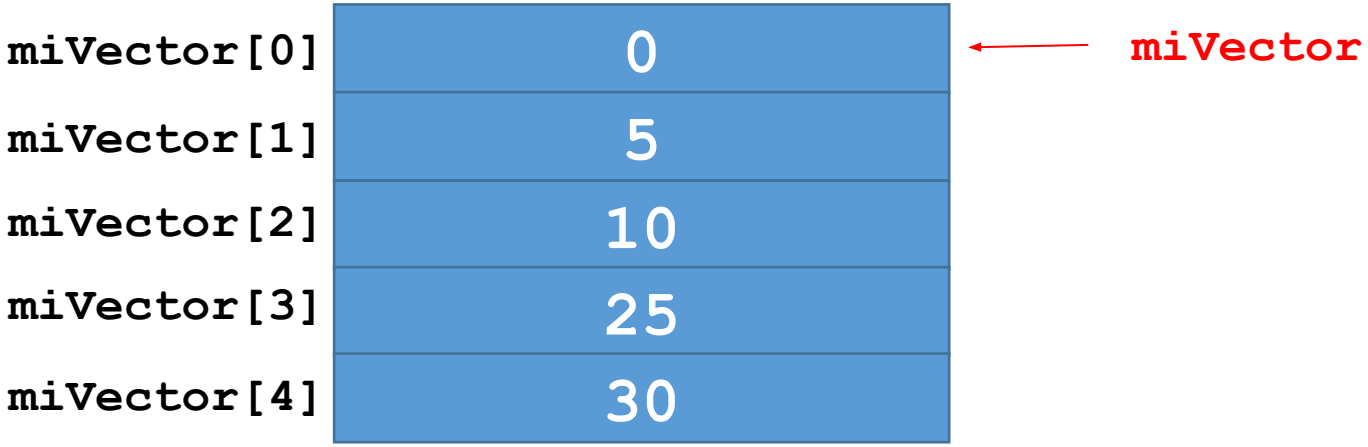
Punteros
Pasaje de variables a funciones por referencia

```
/* El programa imprime "var=6" ya que la variable se pasa por referencia,
es decir, se le pasa a la función un puntero a la variable y esta se
modifica directamente dentro del cuerpo de la función*/

void incrementar (int *punt)
{
   *punt=*punt+1;
}

int main (void)
{
   int var=5;
   incrementar(&var);
   printf("var=%d",var);
   return 0;
}
```

Vectores y punteros

Cuando se declara un vector, el identificador utilizado es en realidad un puntero al tipo de datos del vector.

```
//Declaración de un vector inicializado
int miVector[5]={0,5,10,25,30};
```



`miVector` es un puntero a int (es del tipo int *) y contiene la dirección de inicio del vector

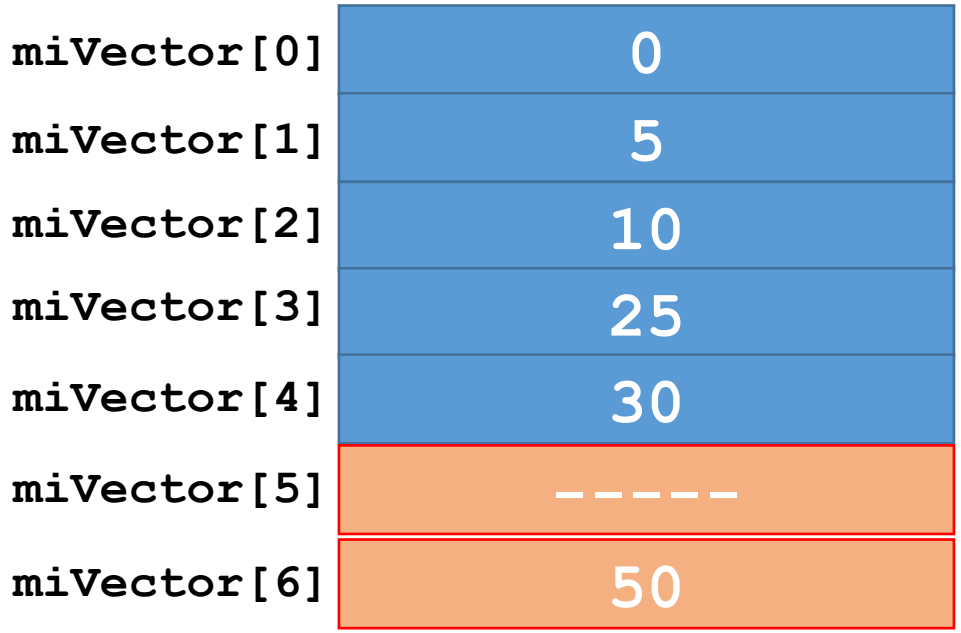
`miVector[i] (con i entre 0 y 4)` es una variable entera (del tipo int)

Vectores y punteros: No comprobación de contornos

Cuidado: En C no se realiza comprobación de contornos, es decir, el compilador no chequea si se están accediendo a elementos fuera del vector.

```
//Ejemplo de no comprobación de contornos
int miVector[5]={0,5,10,25,30};
miVector[6]=50;
```

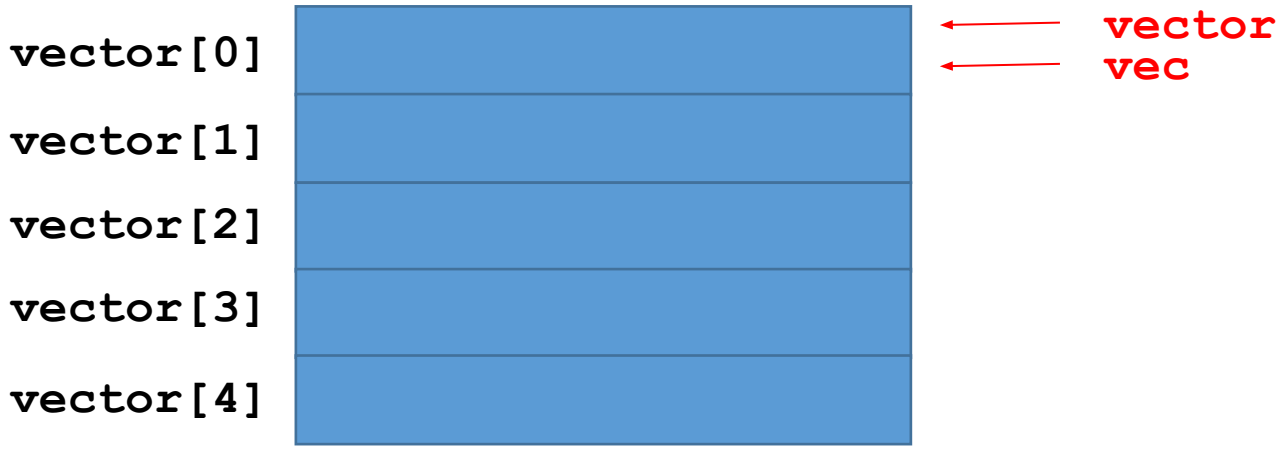
Si hago `miVector[6]` el programa intentará acceder a una posición que no está dentro del espacio de memoria reservado para el vector pudiendo sobreescribir un área de memoria generando un comportamiento inesperado en el código.



Pasaje de vectores a funciones

Cuando se utiliza un vector como argumento de una función, se envía el puntero al vector. Dentro del cuerpo de la función se tendrá acceso al vector original.

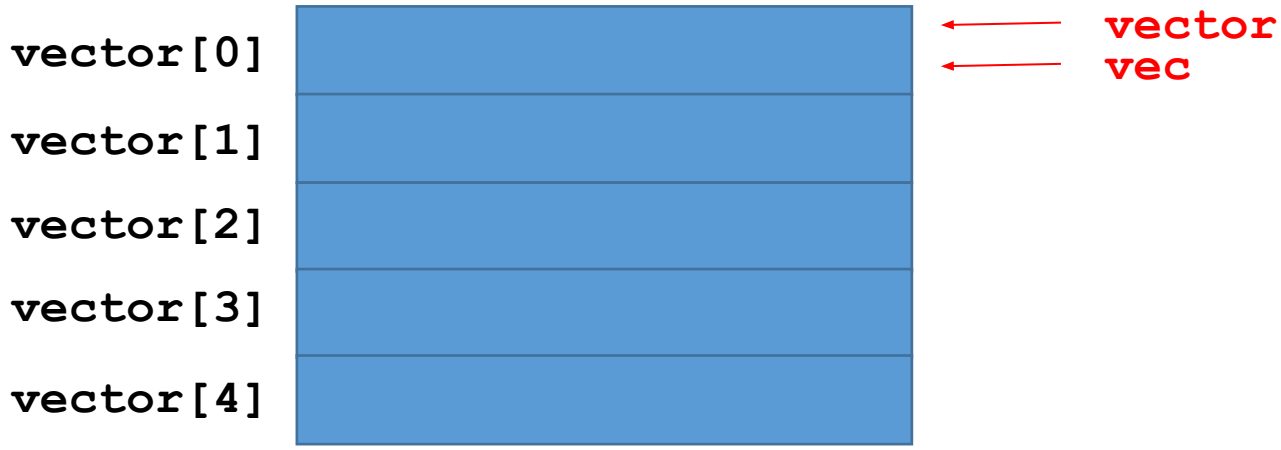
```
void borrar (int vec[])
{
   int i;
   for(i=0;i<5;i++) vec[i]=0;
}

int main (void)
{
   int vector[5];
   borrar(vector);
   //¿Cómo queda vector?
   return 0;
}
```



Pasaje de vectores a funciones (alternativa)

Cuando se utiliza un vector como argumento de una función, se envía el puntero al vector. Dentro del cuerpo de la función se tendrá acceso al vector original.

```
void borrar (int * vec)
{
   int i;
   for(i=0;i<5;i++) vec[i]=0;
}

int main (void)
{
   int vector[5];
   borrar(vector);
   //¿Cómo queda vector?
   return 0;
}
```



A programar

1) Realice una función que recibe un vector de enteros y la cantidad de elementos del mismo. La función devuelve el módulo de los elementos del vector (es decir, si encuentra números negativos, los cambia por sus opuestos). Compruebe su correcto funcionamiento.

2) Realice una función que recibe un vector de enteros y la cantidad de elementos del mismo. La función debe invertir el orden de los elementos del vector. Realice un programa que compruebe su correcto funcionamiento.

3) Realice una función que recibe dos vectores de R3 y devuelve su producto escalar. Compruebe su correcto funcionamiento.

Matriz: Definición

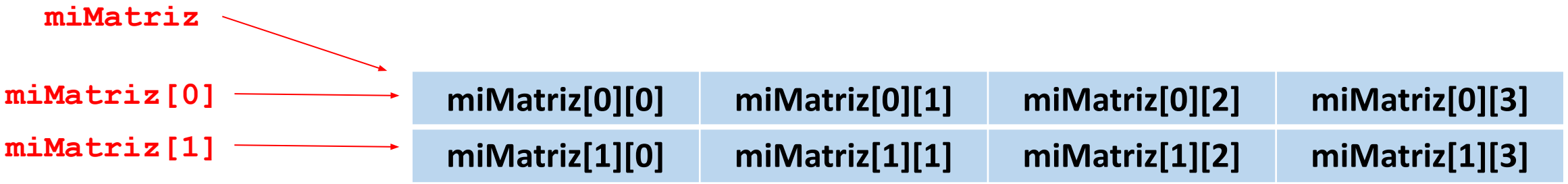
Es un arreglo multidimensional de variables del mismo tipo, cada una identificadas con dos o más subíndices numéricos.

```
//Declaración de una matriz inicializada
int miMatriz[2][4]={{1,2,3,4},{5,6,7,8}};
```

	Columna 0	Columna 1	Columna 2	Columna 3
Fila 0	miMatriz[0][0]	miMatriz[0][1]	miMatriz[0][2]	miMatriz[0][3]
Fila 1	miMatriz[1][0]	miMatriz[1][1]	miMatriz[1][2]	miMatriz[1][3]

	Columna 0	Columna 1	Columna 2	Columna 3
Fila 0	1	2	3	4
Fila 1	5	6	7	8

Matriz: Definición



`miMatriz` es del tipo **`int **`** y apunta al comienzo de la matriz

`miMatriz[0]` es del tipo **`int *`** y apunta al comienzo de la primera fila de la matriz

`miMatriz[1]` es del tipo **`int *`** y apunta al comienzo de la segunda fila de la matriz

A programar

1) Realice una función que recibe una matriz de 2x2 y devuelve el valor de su determinante

2) Realice una función que recibe dos matrices de 2x2 y una tercera matriz donde guardará el resultado del producto de ambas.

Ejercicio adicional ...

Realice una función de reciba un vector de enteros y su cantidad de elementos y devuelva el vector ordenado de menor a mayor.