Tema 4: Tipos Avanzados de Datos

Oscar Perpiñán Lamigueiro - David Álvarez

Vectores en C

Definición

Conjunto de valores numéricos del mismo tipo

Código

```
tipo identificador[tamaño];
```

tipo Tipo de los elementos del vector (`int`, `float`, etc.).

identificador Nombre del vector.

longitud Número de elementos (no puede ser una variable).

Ejemplos

```
// Declara un vector llamado miVector compuesto por
// tres elementos de tipo int.
int miVector[3];

// Declara un vector e inicializa todos sus elementos
int miVector[3] = {2, 23, 0};

// Declara un vector e inicializa el primer elemento
// (resto quedan a 0)
int miVector[3] = {2};

// Declara un vector sin dimensión.
// La dimensión queda determinada a partir del numero de elementos de inicializaci
    ón
int miVector[] = {2, 23, 24};
```

Elementos de un vector

- Se referencian con el nombre del vector seguido de un subíndice entre corchetes.
- El subíndice representa la posición del elemento dentro del vector.
- La **primera posición** del vector tiene el **subíndice 0**.

```
#include <stdio.h>

int main(){
  int miVector[3];
  miVector[0] = 10;
  miVector[1] = 2 * miVector[0];
  miVector[2] = miVector[0] + miVector[1];

  printf("Posicion 0 = %i\n", miVector[0]);
  printf("Posicion 1 = %i\n", miVector[1]);
  printf("Posicion 2 = %i\n", miVector[2]);

  return 0;
}
```

Asignación de valores

No se pueden asignar vectores completos, solo dato a dato

```c
#include <stdio.h>
int main() {
  int v1[5];
  // Error
  v1 = {1, 2, 3, 4, 5};
  return 0;
}
```

```c
#include <stdio.h>
int main()
{
  int v1[5] = {1, 2, 3, 4, 5}, v2[5];
  // Error
  v2 = v1;
  return 0;
}
```

Longitud del vector

La dimensión de un vector es un valor constante: **no puede usarse una variable**.

```c
# define N 10
int main() {
  // Correcto: el precompilador sustituye N por el valor constante 10
  int miVector[N];

  return 0;
}
```

```c
#include <stdio.h>
int main() {
  int n = 10;
  // INCORRECTO, n es una variable
  int miVector[n];

  return 0;
}
```

Acceso a datos de un vector

```c
#include <stdio.h>

int main()
{
  float temp[5] = {2.1, 4.9, 0.51, 4.3, 9.01};
  float suma = 0;
  int i;
  // Es común el uso de bucles for para recorrer un vector. // Es importante
      recordar que el primer elemento
  // tiene índice 0.
  for (i = 0; i < 5; i++)
  {
    printf("El elemento %i es %f\n", i + 1, temp[i]);
    suma = suma + temp[i];
  }
  printf("La temperatura media es: %f\n", suma/5);
  return 0;
}
```

Vectores de Dimensión desconocida

Solución provisional: definir un vector de dimensión suficientemente elevada y emplear sólo un número reducido de elementos.

```
#include <stdio.h>
int main() {
  int i, n;
  // Definimos un vector de dimension muy grande
  int vect[100];

  printf("Nº datos? ");
  //El usuario debe teclear un n < 100
  scanf("%i", &n);
  //Utilizamos solo las n primeras
  for(i = 0; i < n; i++)
  {
    scanf("%i", &vect[i]);
  }
  return 0;
}
```

Matrices

Una matriz es un conjunto de valores del mismo tipo (`int`, `char`, `float`, etc.), de dos o más dimensiones

```
tipo identificador[dimension_1][dimension_2] ... [dimension_n];
```

tipo	Tipo de los elementos de la matriz.
identificador	Nombre de la matriz.
dimensión_n	Dimensión n-ésima de la matriz.

Ejemplo

```
// Crea una matriz de datos enteros, llamada
// tabla, de dos dimensiones y 9 elementos.
int tabla[3][3];
```

Elementos de una matriz

Se referencian con el nombre de la matriz seguido de tantos subíndices, entre corchetes, como dimensiones tenga la matriz.

```
#include <stdio.h>
int main (){
  int matriz[2][2]; // Matriz 2 x 2
  int fila, columna;
  // Inicializacion de elementos
  matriz[0][0] = 1;
  matriz[0][1] = 2;
  matriz[1][0] = 3;
  matriz[1][1] = 4;
  // Recorre matriz con un bucle for anidado
  for(fila = 0; fila < 2; fila++) {
     for(columna = 0; columna < 2; columna++)
        printf("%i\t", matriz[fila][columna]);
     printf("\n\n");
  }
  return 0;
}
```

Inicialización de una matriz

Los elementos de una matriz pueden iniciarse en el momento de la declaración.

```
#include <stdio.h>
int main() {
  int matriz[2][3] = // Matriz 2 x 3
  {
     {10, 20, 30}, // 1a fila
     {40, 50, 60} // 2a fila
  };
  int fil, col;
  // Recorremos con bucle anidado
  for(fil = 0; fil < 2; fil++){
     for(col = 0; col < 3; col++)
       printf("%i\t",matriz[fil][col]);
     printf("\n\n");
    }
  return 0;
}
```

Inicialización de una matriz

Los elementos de una matriz pueden iniciarse en el momento de la declaración.

```
#include <stdio.h>
int main()
{
  // Matriz de dos filas, tres columnas
  int matriz[2][3] = {10, 20, 30, 40, 50, 60};
  int fil, col;
  // Recorremos con bucle anidado
  for(fil = 0; fil < 2; fil++)
  {
    for(col = 0; col < 3; col++)
      printf("%i\t",matriz[fil][col]);
    printf("\n\n");
  }
  return 0;
}
```

Operaciones con matrices: suma

```c
#include <stdio.h>
int main() {
  int i,j;
  int m1[2][3] = {1, 2, 3, 4, 5, 6};
  int m2[2][3] = {4, 5, 12, 23, -5, 6};
  int m3[2][3]; // Matriz resultado
  // Realiza la suma con bucle anidado
  for(i = 0; i < 2; i++) // Filas
    for(j = 0; j < 3; j++) // Columnas
      m3[i][j] = m1[i][j] + m2[i][j];
  // Imprime resultado con bucle anidado
  for(i = 0; i < 2; i++) // Filas
  {
    for(j = 0; j < 3; j++) // Columnas
      printf("%i\t",m3[i][j]);
    printf("\n");
  }
  return 0;
}
```