8. Registres

8.1. Introducció

8.2. Declaració de registres

8.3. Accés als camps dels registres

8.4. Còpia de registres

8.5. Comparació de registres

8.1. Introducció

• Els vectors permeten agrupar elements d'un mateix tipus. Cada element del vector és accessible a través d'un índex.

• En ocasions necessitaràs agrupar dades de diferents tipus o preferiràs accedir a diferents elements d'un grup de dades a través d'un identificador, no d'un índex.

• Els registres són agrupacions heterogènies de dades els elements del qual (anomenats camps) són accessibles mitjançant identificadors.

8.2. Declaració de registres

• Suposem que volem tenir les següents dades d'una persona:

-Nom, amb un màxim de 40 caràcters

-Edat, un enter

-DNI, una cadena de 9 caràcters

• Podem definir un registre 'persona' abans de l'aparició del programa principal, el main.

```
#define MAXNOM 40
#define LONDNI 9

struct Persona {
   char nombre[MAXNOM+1];
   int  edad;
   char dni[LONDNI+1];
}; // <- Fíjate en el punto y coma: es fácil olvidarse de ponerlo.
```

La definició d'un registre introdueix un nou tipus de dades en el nostre programa. En l'exemple hem definit el tipus struct Persona. Ara pots declarar variables del tipus struct Persona així:

struct Persona pepe, juan, ana;

8.3. Accés als camps dels registres

• Per accedir a cadascun dels camps d'una variable de tipus struct es fa separant amb un punt l'identificador de la variable del corresponent identificador del campo.

• Per exemple, pepe.edad és l'edat de Pepe (un enter), juan.nombre és el nom de Juan (una cadena) i ana.dni[9] és la lletra del DNI d'Ana (un caràcter).

• Cada variable de tipus struct Persona ocupa, en principi, 55 bytes: 41 per al nom, 4 per a l'edat i 10 per al DNI.

Aquest programa il·lustra com accedir als camps d'un registre llegint per teclat els seus valors i mostrant per pantalla diferents informacions emmagatzemades en ell:

```
13 int main(void)
14 {
15   struct Persona ejemplo;
16   char linea[81];
17   int i, longitud;
18
19   printf("Nombre: "); gets(ejemplo.nombre);
20   printf("Edad   : "); gets(linea); sscanf(linea, "%d", &ejemplo.edad);
21   printf("DNI    : "); gets(ejemplo.dni);
22
23   printf("Nombre leído: %s\n", ejemplo.nombre);
24   printf("Edad leída  : %d\n", ejemplo.edad);
25   printf("DNI leído   : %s\n", ejemplo.dni);
```

```
27    printf("Iniciales del nombre: ");
28    longitud = strlen(ejemplo.nombre);
29    for (i=0; i<longitud; i++)
30      if (ejemplo.nombre[i] >= 'A' && ejemplo.nombre[i] <= 'Z')
31        printf("%c", ejemplo.nombre[i]);
32    printf("\n");
33
34    printf("Letra del DNI: ");
35    longitud = strlen(ejemplo.dni);
36    if (ejemplo.dni[longitud-1] < 'A' || ejemplo.dni[longitud-1] > 'Z')
37      printf("No tiene letra.\n");
38    else
39      printf("%c\n", ejemplo.dni[longitud-1]);
40
41    return 0;
42  }
```

8.4. Còpia de registres

- Els registres es poden copiar íntegrament sense cap problema.

- El següent programa còpia el contingut d'un registro en un altre i passa a minúscules el nom de la còpia:

```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>

#define MAXNOM 40
#define LONDNI 9

struct Persona {
  char nombre[MAXNOM+1];
  int  edad;
  char dni[LONDNI+1];
};

int main(void)
{
  struct Persona una, copia;
  char linea[81];
  int i, longitud;

  printf("Nombre: "); gets(una.nombre);
  printf("Edad   : "); gets(linea); sscanf(linea, "%d", &una.edad);
  printf("DNI    : "); gets(una.dni);
```

```
23
24    copia = una; // Copia
25
26    longitud = strlen(copia.nombre);
27    for (i=0; i<longitud; i++)
28       copia.nombre[i] = tolower(copia.nombre[i]);
29
30    printf("Nombre leído: %s\n", una.nombre);
31    printf("Edad leída   : %d\n", una.edad);
32    printf("DNI leído    : %s\n", una.dni);
33
34    printf("Nombre copia: %s\n", copia.nombre);
35    printf("Edad copia  : %d\n", copia.edad);
36    printf("DNI copia   : %s\n", copia.dni);
37
38    return 0;
39 }
```

La copia s'efectua inclús quan els elements del registre són vectors. O siga que copiar vectors amb una assignació està prohibit, però copiar registres és possible.

8.5. Comparació de registres

• Els registres no es poden comparar.

• El següent programa, per exemple, efectua una còpia d'un registre en un altre per a continuació intentar dir-nos si els dos registres són iguals o no.

• Però no serà possible ni tan sols compilar-lo, la línia 24 donarà un error.

```
12 int main(void)
13 {
14   struct Persona una, copia;
15   char linea[81];
16   int i, longitud;
17
18   printf("Nombre: "); gets(una.nombre);
19   printf("Edad   : "); gets(linea); sscanf(linea, "%d", &una.edad);
20   printf("DNI    : "); gets(una.dni);
21
22   copia = una; // Copia
23
24   if (copia == una) // Comparación ilegal.
25     printf("Son iguales\n");
26   else
27     printf("No son iguales\n");
28
29   return 0;
30 }
```

Per poder saber si dos registres són iguals s'han de comparar cadascun dels camps corresponents dels dos registres que volem comparar.

```
12 int main(void)
13 {
14   struct Persona una, copia;
15   char linea[81];
16   int i, longitud;
17
18   printf("Nombre: "); gets(una.nombre);
19   printf("Edad   : "); gets(linea); scanf(linea, "%d", &una.edad);
20   printf("DNI    : "); gets(una.dni);
21
22   copia = una; // Copia
23
24   if (strcmp(copia.nombre, una.nombre)==0 && copia.edad==una.edad
25       && strcmp(copia.dni, una.dni)==0)
26     printf("Son iguales\n");
27   else
28     printf("No son iguales\n");
29
30   return 0;
31 }
```