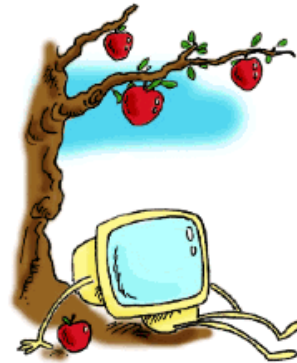


- ◆ 7.1 Introducció a les estructures (registres)
- ◆ 7.2. Anidament d'estructures
- ◆ 7.3 Pas d'estructures
- ◆ 7.4 Exemples



7.1 Introducció a les estructures (I)

- ◆ **Registre**
 - Conjunt d'elements heterogenis (anomenats camps) que comparteixen una estructura (sota un mateix nom i tipus) que poden ser tractats com a una unitat.
 - És a dir, permeten posar sota un denominador comú diferents elements relacionats entre ells.
 - Declaració de registres:

```
struct identificador_estructura
{
    tipus identificador_1;
    ...
    tipus identificador_n;
};
```
 - **Exemple .**

```
struct fitxa_alumne
{
    long dni;
    char[20] nom;
};
```

Aleshores es poden declarar variables del tipus creat. Internament cada variable conté tots els camps de l'estructura.

7.1 Introducció a les estructures (II)



- ◆ **Definició de variables estructura**
 - Es fa com en qualsevol tipus de variable.
 - **Exemple:** `fitxa_alumne fitxes[100]`
fitxes és un array de 100 elements que cadascun conté l'estructura definida.
 - Les estructures es poden passar entre elles, a l'igual que passa amb les variables normals.
 - **Exemple:** copiar la fitxa de l'alumne 1 al 10 es tan senzill com fer `fitxes[10]=fitxa[1];`
 - Si es vol copiar només el nom de la fitxa de l'alumne 1 al nom de la fitxa 10 aleshores es fa `fitxes[10].nom = fitxa[1].nom;`
- ◆ **Inicialització d'estructures.**
 - Es pot fer dintre el programa o es pot declarar la variable estructura, posant els camps dintre claus.
 - **Exemple:** `fitxa_alumne alumne1={35353535, "Pere Primer"};`
 - El tamany de l'estructura s'obté emprant la funció ja coneguda `sizeof()`
 - **Exemple:** `tamany = sizeof(alumne1);`
- ◆ **Accés a una estructura de dades**
 - Per a accedir a una estructura de dades es pot emprar
 - l'operador punt (`.`). La sintaxi és molt simple:
`<nom variable estructura>.<nom membre> = dada`
 - l'operador punter (`->`), útil quan es treballa amb estructures apuntadors. Sintaxi:
`<apuntador estructura> -> <nom membre> = dada`Aquest operador es veurà en més profunditat en el proper capítol.

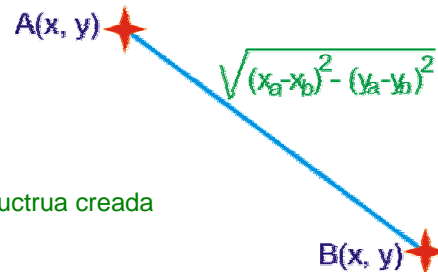
7.1 Introducció a les estructures (III)



- **En exemple complet (emprant l'estructura punt):**

```
/*
Trobar la distància entre dos punts
*/
struct coord          //declarem l'estructura coordenada
{
    float x, y;
};

void main()
{
    coord a, b;        //declarem variables de l'estructura creada
    float d;
    printf("\tOPERANT AMB COORDENADES\n\n");
    printf("\tEntra les coordenades x, y del punt A: ");
    scanf("%f%f", &a.x, &a.y);    //definim la variable a
    printf("\tEntra les coordenades x, y del punt B: ");
    scanf("%f%f", &b.x, &b.y);    //definim la variable b
    d=sqrt((a.x-b.x)*(a.x-b.x)+(a.y-b.y)*(a.y-b.y));
    printf("\n\tLa distancia entre els dos punts es %3.2f\n\n", d);
}
```



7.2 Anidament en estructures



- Una estructura pot contenir altres estructures.

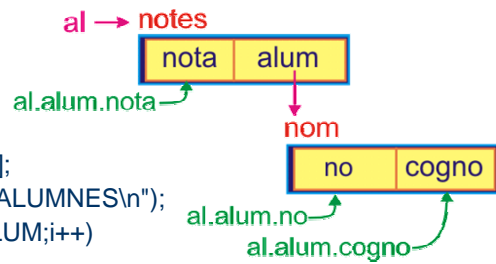
- Exemple:

- L'estructura

```
#include <stdio.h>
#define NALUM 20
/*
Informació personal
*/
struct nom //Nom alumne
{
    char no [20], cogno [20];
};
struct notes //estructura notes
{
    float nota;
    nom alum;
};
```

//El programa

```
void main()
{
    notes al[NALUM];
    printf("\tNOTES ALUMNES\n");
    for (int i=0;i<NALUM;i++)
    {
        printf("\tEntra nom, cognom i nota alumne %i: ", (i+1));
        fflush(stdin);
        scanf("%s%s%f", &al[i].alum.no, &al[i].alum.cogno, &(al[i].nota));
    }
    printf("\n\tEL RESULTAT HA ESTAT:\n");
    for (i=0;i<NALUM;i++)
    {
        printf("\tAlumne %i:\n", (i+1));
        printf("\t\tNom: %s%s\n", al[i].alum.no, al[i].alum.cogno);
        printf("\t\tNota: %3.1f\n", al[i].nota);
    }
}
```



7.3 Pas d'estructures (I)



- El pas d'estructures es pot fer per valor o per referència. En estructures grans cal fer el pas per referència, ja que passar per valor podria omplir la memòria.
- Es veurà com es treballa amb tres exemples, que treballaran sobre l'estructura data:

```
struct data
{
    int dia;
    int mes;
    int any;
};
```

- En els tres casos es definiran tres funcions:
 - Entrada d'una data
 - Calcular una nova data
 - Imprimir data
- És important entendre com es passen els paràmetres i quins són els resultats en cada cas.

7.3 Pas d'estructures (II)



- **Pas per valor:** no es modifiquen els continguts de les variables en les funcions.

```
data calcular(data); //Modifica la data sobre la que s'envia (es modifica en el procediment)
void imprimir (data, int); //Procediment d'impressió: només agafa l'estructura per a imprimir-la
void main()
{
    data d2, d1 = {21, 2, 2001};
    imprimir(d1, 1);
    d2=calcular(d1);
    imprimir(d1, 2);
    imprimir(d2, 3);
}
data calcular(data d) //definició procediment modificar data
{
    d.dia = d.dia+1;
    d.mes = d.mes+1;
    d.any = d.any+1;
    return (d);
}
void imprimir (data d, int i) //definició procediment imprimir
{
    printf("Data cas %i: %i/%i/%i\n", i, d.dia, d.mes, d.any);
}
```

Resultat:

Data cas 1: 21/2/2001

Data cas 2: 21/2/2001

Data cas 3: 22/3/2002

7.3 Pas d'estructures (III)



- **Pas per referència:** es modifiquen els continguts de les variables en les funcions.

```
data calcular(data d); //El procediment modifica el contingut de la variable que s'envia
void imprimir (data, int); //Procediment d'impressió: només agafa l'estructura per a imprimir-la
void main()
{
    data d2, d1 = {21, 2, 2001};
    imprimir(d1, 1);
    d2=calcular(d1);
    imprimir(d1, 2);
    imprimir(d2, 3);
}
data calcular(data &d) //definició procediment modificar data
{
    d.dia = d.dia+1;
    d.mes = d.mes+1;
    d.any = d.any+1;
    return (d);
}
void imprimir (data d, int i) //definició procediment imprimir
{
    printf("Data cas %i: %i/%i/%i\n", i, d.dia, d.mes, d.any);
}
```

Un petit canvi posar & ens canvia el resultat :

→ pas per referència

→ els canvis introduïts en d (d1) es propaguen!

Resultat:

Data cas 1: 21/2/2001

Data cas 2: 22/3/2002

Data cas 3: 22/3/2002

7.3 Pas d'estructures (IV)



- **Pas per adreça:** es modifiquen els continguts de les variables en les funcions.

```
data calcular(data d); //Modifica la data sobre la que s'envia (es modifica en el procediment)
void imprimir (data d, int i); //Procediment d'impressió: només agafa l'estructura per a imprimir-la
void main()
{
    data d2, d1 = {21, 2, 2001};
    imprimir(d1, 1);
    d2=calcular(&d1);
    imprimir(d1, 2);
    imprimir(d2, 3);
}
data calcular(data d) //definició procediment modificar data
{
    (*d).dia = (*d).dia+1;
    (*d).mes = (*d).mes+1;
    (*d).any = (*d).any+1;
    return (*d);
}
void imprimir (data d, int i) //definició procediment imprimir
{
    printf("Data cas %i: %i/%i/%i\n", i, d.dia, d.mes, d.any);
}
```

Pas per referència amb apuntadors:

→ es passa una adreça

→ es recull amb un apuntador!

Resultat:

Data cas 1: 21/2/2001

Data cas 2: 22/3/2002

Data cas 3: 22/3/2002

7.4 Exemples (I)



- **Creació d'un llistín telefònic (un exemple complet! – val la pena entendre-ho)**

- Es vol crear un llistín telefònic de fins a 100 adreces ordenades per nom.
- Cada adreça constarà del nom de la persona i del número de telèfon
- Les operacions que s'hauran de permetre amb el llistín són:
 - ♦ insertar nova adreça
 - ♦ eliminar adreça
 - ♦ cercar per nom
 - ♦ imprimir el llistín

- **Estructura del programa**

- Per a crear el llistín es crea l'estructura

```
struct llist
{
    char nom[30]; //Nom de l'abonat
    long tel; //Telèfon de l'abonat
};
```

- El programa s'estructura en un cos principal i 4 funcions, que corresponen a cada acció:

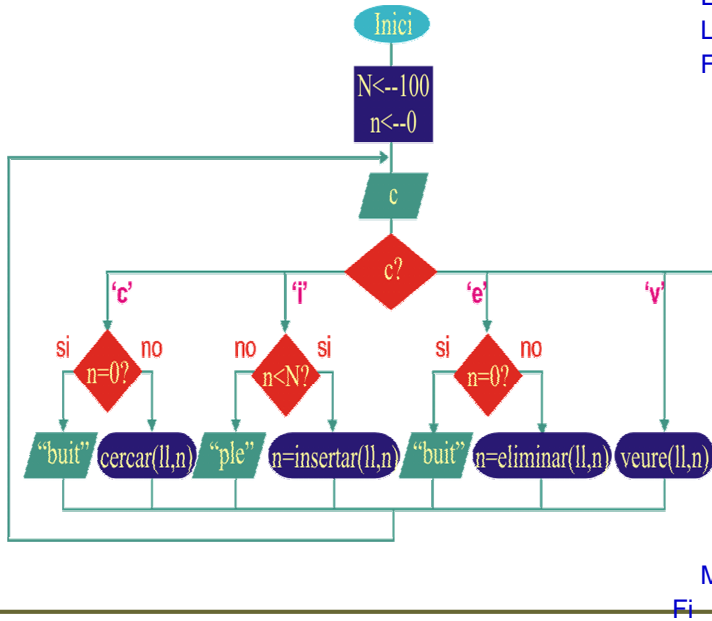
```
void cercar (llist [], int); //Cercar un abonat per nom
int insertar (llist [], int); //Insertar un nou abonat (si el nom no existeix!)
int eliminar (llist[], int); //Eliminar un abonat
void veure ( llist[], int); //Veure tot el llistín
```

Es pot veure com, en cada crida, es passa l'array que conforma el llistín.

7.4 Exemples (II)

El cos principal

- La seva missió és inicialitzar variables i distribuir les tasques.
- El seu diagrama de flux ve donat per:



Pseudocodi

Caracter c

l·list ll[N]

Inici

n ← 0

Escriure("Què vols fer c, i, e, v, s?")

Llegir(c)

Fer

EnCasDe (c)

'c': Si (n <> 0) Fer cercar(ll, n)

SiNo Escriure("buit")

Fi_c

'i': Si (n < N) Fer insertar(ll, n)

SiNo Escriure("ple")

Fi_i

'e': Si (n <> 0) Fer eliminar(ll, n)

SiNo Escriure("buit")

Fi_e

'v': Si (n <> 0) Fer veure(ll, n)

Fi_v

's': Continuar

FiEnCasDe

Mentre (c <> 's');

Fi

Estructures

11

7.4 Exemples (III)

Funció insertar

- Inserta un abonat en el llistín.
- De no existir, ha d'entrar el nom i el número de telèfon.

Pseudocodi

Funció Enter insertar (l·list a[], Enter n)

Enter i, j

Cadena cad //Nom a insertar

Inici

i = 0

Llegir (cad)

Mentre (i < n i a[i].nom <> cad) Fer i ← i + 1

Si (a[i].nom = cad) Escriure ("Ja existeix!")

SiNo Fer

a[i].nom = cad

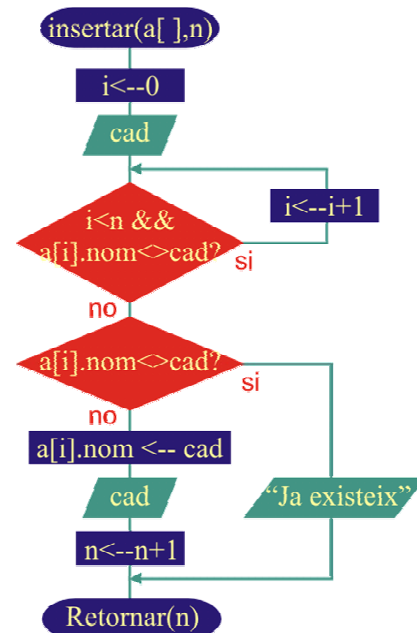
Llegir (a[i].tel)

n ← n + 1

FiSi

Retornar (n)

Fi



Estructures

12

7.4 Exemples (IV)

Funció eliminar

- Elimina un abonat en el llistín.
- De no existir, h'ha d'informar.

Pseudocodi

Funció Enter eliminar (llist a[], Enter n)

Enter i

Cadena cad //Nom a insertar

Inici

i=0

Llegir (cad)

Mentre (i<n i a[i].nom<cad) Fer i ← i+1

Si (a[i].nom <> cad) Escriure ("No existeix!")

SiNo Fer

Per (i = i; i<n)

a[i] ← a[i+1]

i ← i+1

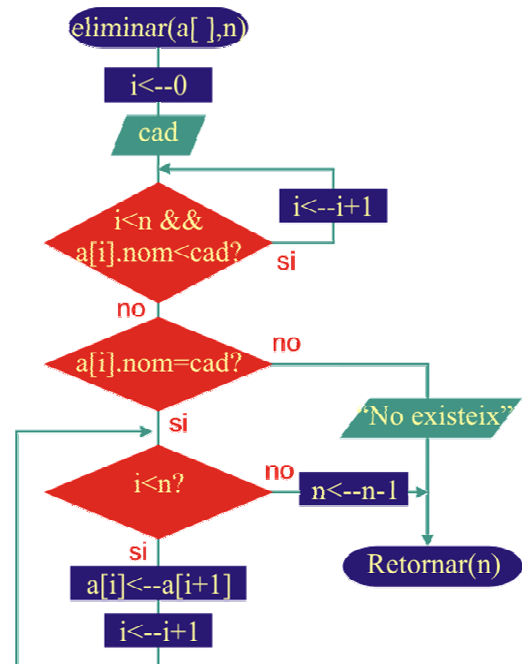
FiPer

n ← n-1

FiSi

Retornar (n)

Fi



7.4 Exemples (V)

Funció cercar

- Cerca un abonat en el llistín.

Pseudocodi

Funció cerca (llist a[], Enter n)

Enter i

Cadena cad //Nom a insertar

Inici

i=0

Llegir (cad)

Mentre (i < n && a[i].nom <> cad) Fer i ← i+1

Si (i = n) Escriure ("No existeix!")

SiNo Escriure (i, a[i].tel)

FiSi

Fi

Funció veure

- Llista el contingut del llistín.

Pseudocodi

Funció veure (llist a[], Enter n)

Enter i

Inici

Si (n=0) Escriure ("Buit")

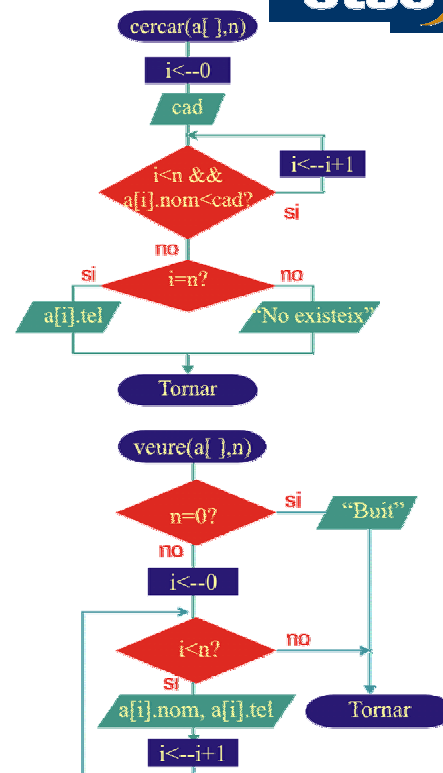
SiNo Per (i=0; i<n) Fer

Escriure (i, a[i].nom, a[i].tel)

i ← i+1

FiPer

Fi



7.4 Exemples (VII)



- Creació d'un llistín telefònic (codificació en C)
- Inicialització

```
/*  
Llistín de N adreces de telèfon ordenades des del conjunt buit (n=0).  
Crear el següent conjunt de funcions sobre l'array:  
Cercar nom  
Eliminar adreça  
Insertar adreça (sempre i quan no existeixi)  
Imprimir l'array  
Sortir  
Es util treballar amb les funcions de strings strcpy() i strcmp()  
*/
```

Explicació programa

```
#include <stdio.h>  
#include <string.h>  
#define N 100 //Nombre elements llistin  
#define LON 30 //Longitud de l'array
```

Llibreries i macros

```
struct llist //Estructura base  
{  
    char nom[LON];  
    long tel;  
};
```

Estructura llistín

```
void cercar (llist [], int); //Declaració funció cercar per nom  
int insertar (llist [], int); //Declaració funció insertar abonats  
int eliminar (llist[], int); //Declaració funció eliminar abonats  
void veure ( llist[], int); //Declaració funció veure llistin
```

Declaració funcions

Estructures

15

7.4 Exemples (VIII)



- Programa principal

```
void main()
```

```
{  
    char c;  
    llist ll[N]; //Declaració de l'array d'abonats  
    int n=0; //Numero abonats entrats en l'array  
    printf("\t--- LLISTIN ---\n");  
    do
```

Inicialització

```
{  
    printf("\n\t...(c)ercar, (i)nsertar, (e)liminar, (v)eure, (s)ortir? ");  
    fflush(stdin);  
    scanf("%c", &c);
```

```
    switch(c) //Rutina principal
```

Selecció d'operació

```
{  
    case 'c': if (n!=0) cercar (ll, n);  
              else printf("\t\t --> ARRAY BUIT!\n");  
              break;  
    case 'i': if (n<N) n = insertar (ll, n);  
              else printf("\t\t --> ARRAY PLE\n");  
              break;  
    case 'e': if (n!=0) n = eliminar (ll, n);  
              else printf("\t\t --> ARRAY BUIT!\n");  
              break;  
    case 'v': veure (ll, n);  
}
```

```
}while (c!='s');
```

```
}
```

Estructures

16

7.4 Exemples (IX)



■ Funció insertar abonat

```
int insertar (llist a[], int n) //Definició funció insertar
{
    int i=0, j;
    char cad[LON];
    printf("\t\tEntra nom: ");
    fflush(stdin);
    gets(cad);
    while (i<n && strcmp(a[i].nom,cad)<0) i++;
    if (strcmp(cad,a[i].nom)==0) printf("\t\t --> Nom ja entrat %i\n", i);
    else
    {
        for (j=n; j>i; j--) a[j]=a[j-1];
        strcpy(a[i].nom, cad);
        printf("\t\tEntra telefon: ");
        fflush(stdin);
        scanf("%i", &a[i].tel);
        n++;
        printf("\t\t --> Insertat en posicio %i de %i abonats\n", i+1, n);
    }
    return(n);
}
```

Inicialització

Entrar abonat a insertar

Existeix abonat ?

Inserció abonat

Estructures

17

7.4 Exemples (X)



■ Funció eliminar abonat

```
int eliminar (llist a[], int n) //Definició funció eliminar
{
    int i;
    char cad[LON];
    printf("\t\tEntra nom: ");
    fflush(stdin);
    gets(cad);
    i=0;
    while (i<n && strcmp(a[i].nom,cad)<0) i++;
    if (strcmp(a[i].nom,cad)!=0) printf("\t\t --> Nom no existent!\n", i);
    else
    {
        for (; i<n; i++) a[i]=a[i+1];
        n--;
        printf("\t\t --> Queden %i abonats en l'array!\n", n);
    }
    return(n);
}
```

Inicialització

Entrar abonat a eliminar

Existeix abonat ?

Inserció abonat

Estructures

18

7.4 Exemples (XI)



■ Funció cercar llistín

```
void cercar (llist a[], int n) //Definició funció cercar
{
    int i=0;
    char cad[LON];
    printf("\t\t --> Entra el nom: ");
    fflush(stdin);
    gets(cad);
    while (i<n && strcmp(a[i].nom,cad)!=0) i++;
    if (i==n) printf("\t\t --> Nom no existent!\n");
    else printf("\t\t Lloc %i. Tel: %i\n", i+1, a[i].tel);
}
```

Inicialització

Cercar abonat

Veure si existeix

■ Funció cercar llistín

```
void veure (llist a[], int n) //Definició funció veure
{
    if (n==0) printf ("\t\t --> ARRAY BUIT\n");
    else for (int i=0; i<n; i++) printf("\t %i. %s -> %i\n", i, a[i].nom, a[i].tel);
}
```

Impressió