Dades Estructurades (I)

Què treballarem en aquest laboratori?

- Tipus de Dades Estructurades Homogènies
  - Taules d'una dimensió o Vectors
  - Un cas concret de les taules de caràcters (Strings)

- Exemples

Vectors

Recordem alguns aspectes sobre els vectors i la notació de C

Vector és un conjunt de valors d'un mateix *tipus* (homogenis).

Declaració:

tipusDades nom[N]; (N és nº de posicions que té el vector, que ha de ser una constant!).

Atenció: Amb aquesta declaració no s'inicialitza el vector, només es reserva espai a la memòria. Ex:

int nom1[5] ⇒ (5 x 4bytes/enter)= 20 bytes

char nom2[5] ⇒ (5 x 1byte/caràcter)= 5 bytes

Inicialització: es pot fer de dues maneres,

- A la definició ⇒ int vector[5]= {0,0,0,0,0};
- Per programa ⇒ for(i=0;i<5;i++) vector[i]=0;

Accés: a cada element del vector,

- a=nom1[i]; per llegir el contingut, recordeu que i ∈[0..4]
- nom1[i]=a; per guardar al vector (a la posició i) el valor de a
- scanf("%d",&nom1[i]) per guardar al vector el valor llegit de teclat

Cadenes de text

- Una cadena de text és una seqüència de caràcters que es tracten com un únic valor.
- Es tradueix i coneix com String.
- Es pot implementar mitjançant una taula d'una dimensió i treballar amb les operacions descrites per a aquest cas. Tanmateix com que s'utilitza molt, la majoria de llenguatges defineixen alguna llibreria adaptada a operacions a realitzar amb aquest tipus d'informació.
- En el llenguatge C una cadena és una seqüència de cero o més caràcters seguits per un caràcter ***NULL*** o ***\0*** (que fa de sentinella!).
- És important preservar el caràcter de terminació ***NULL*** ja que amb aquest és com C defineix i manipula les longituds de les cadenes. Totes les funcions de la llibreria estàndard de C ho requereixen per una operativa satisfactòria.
- Totes les funcions per manipulació de cadenes estan definides a la llibreria *string*, per tant si es volen utilitzar al programa en C cal fer l'include corresponent.
- Estudiarem algunes de les operacions que C permet aplicar a Strings.

Strings

Vectors de caràcters (String)

•Exemple:

```
char nom1[5]=”hola”;
char nom2[5];
```

Al posar el text entre “” implícitament s’afegeix el caràcter NULL al final (a la posició nom1[4]).

No és correcte	Si és correcte
nom2=nom1;	for(i=0;i<5;i++) nom2[i]=nom1[i];

Treballant amb taules:

- L’accés al vector sempre és “element a element”
- Els operadors són els propis del tipus “char”.
- Altres funcionalitats disponibles a la llibreria: **string.lib**

Vectors de caràcters (String)

• És un conjunt de valors de *tipus* caràcter “char”.

Declaració:

char nom[n];

- n-1és nº de caràcters que té el vector
- nom[n-1] ha de tenir '\0' (Caràcter Sentinella)

Inicialització:

A la definició

- char nom[12]=”juan es feo”;
- char nom[12]={'j','u','a','n',' ','e','s',' ','f','e','o', NULL};

Per programa

- nom[0]='j'; nom[1]='u'; nom[2]='a';...; nom[12]='\0';

Accés:

a=nom[i]; per llegir el contingut, recordeu que i ∈[0..11]

nom[i]=a; per guardar al string (a la posició i) el valor de a

scanf(“%s”,nom); per guardar al string el text llegit de teclat. Atenció: aquesta operació no es pot fer en pseudocodi!!

Strings

- La llibreria **string.lib** conte un conjunt de procediments per manipular cadenes: copiar, canviar caràcters, comparar cadenes, etc.

- Els procediments més elementals són:
strcpy (*c1*, *c2*); Copia ***c2*** en ***c1.***
strcat (*c1*, *c2*); Afegeix ***c2*** al final de ***c1.***
int strlen (*cadena*); retorna la longitud de la *cadena.*
int strcmp (*c1*, *c2*); Retorna zero si ***c1*** és igual a ***c2***; no-cero en cas contrari.

- Per a treballar amb aquests procediments, a l'inici del programa s'ha d'afegir:

```
#include <string.h>
```

Exemples

```
#include <stdio.h>
#include <string.h>
int main()
{
char complet [64];
char nom[32];
char cognoms [32];

   printf(“ Introdueix el teu nom: “);
   scanf(“%s”, nom);                    /* suposant que introdueix: Pere */
   printf(“ Introdueix els teus cognoms: “);
   scanf(“%s”, cognoms);                /* suposant que introdueix: Rovira Masdeu */
   strcpy ( complet, nom );             /* complet <- "Pere" */
   strcat ( complet, " ");              /* complet <- "Pere " */
   strcat ( complet, cognoms);          /* complet <- "Pere Rovira Masdeu" */
   printf ( "El nom complet es %s\n", complet );
}
```

El procediment scanf llegeix tot el text fins que l'usuari introdueix el salt de línea (tecla INTRO) o un espai blanc, i ho guarda al string que indiquem, ficant com a sentinella el caràcter NULL.

Fixeu-vos, davant de *nom* no hi ha **&** !!

...explicació a continuació

Exemples

Exemple ús d'un string:

```
char nom[32], c;
int m;

scanf("%s", nom);
scanf("%c", &c);
scanf("%d", &m);
```

Per que quan s'utilitza un string en el procediment scanf no es posa **&** *davant del nom de la variable i si que es posa per qualsevol altre tipus simple de variable?*

Perquè en el procediment scanf el segon paràmetre ha de ser sempre una variable per referència, que en C s'expressa posant & davant del nom de la variable. Pel cas de variables de tipus taula (com és el string), la taula en si ja es una referència i per tant no s'ha de posar &.

Exemples

Fent servir paràmetres per referència en procediments:

```
int main()
{
  char lletres[20];
  llegir_cadena(lletres);
  printf("%s", lletres);
  return(0);
}
```

```
void llegir_cadena(char *a)
{
  scanf("%s", a);
}
```

Quan es pasa per paràmetre una taula:

*char *a* equival a *char a[ ]*

```
int main()
{
  int num;
  llegir_enter(&num);
  printf("%d", num);
  return(0);
}
```

```
void llegir_enter(int *b)
{
  scanf("%d", b);
}
```

No es posa & perquè la variable ja és una referència.

Exemples

Fent servir paràmetres per referència en procediments:

```
int main()
{
  char lletres[20];
  …
  llegir_escriure_cadena(lletres);
  …
  return(0);
}
```

```
void llegir_escriure_cadena(char *a)
{ int i;
  scanf("%s", a);
  i=0;
  while ( a[i] != '\0') {
    printf("%c", a[i]);
    i++;
  }
}
```

```
int main()
{
  int num;
  …
  llegir_ escriure_enter(&num);
  …
  return(0);
}
```

```
void escriure_enter(int *b)
{
   scanf("%d", b);
   printf("%d",*b);
}
```

Cal posar * !

*b conté la dada guardada a b

Strings

Vectors de caràcters (String)

- **String.lib**

Informació sobre la llibreria estàndard per tractar strings:

http://c.conclase.net/librerias/index.php?ansilib=string#inicio

memchr	memcmp	memcpy	memmove
memset	strcat	strchr	strcmp
strcoll	strcpy	strcspn	strerror
strlen	strncat	strncmp	strncpy
strpbrk	strrchr	strspn	strstr
strtok	strxfrm		