ALGORISMES I PROGRAMACIÓ (GESTIÓ AERONÀUTICA)
SESSIÓ 18/04/2005

1. Funcions

Una funció de C és una porció de codi o de programa que realitza una determinada tasca. La seva utilitat principal es la de dividir un programa (**modularització**) en una sèrie de mòduls molt més petits i manejables. La idea és la de dividir un programa gran en un conjunt de subprogrames o funcions més petites que poden ser utilitzades en qualsevol part del programa. Els principals avantatges de l'ús de funcions són:

1. ***Estructuració i Simplificació de codi***. Cada funció té una missió molt concreta (un producte de matrius, lectura de dades ...). que pot ser requerida molts cops al llarg del programa. Per tant, escriure-la en una funció fa que el número total de línies de codi disminueixi i es simplifiqui el codi del programa principal.

2. ***Localització d'errors***. Com una funció pot ser desenvolupada i comprovada per separat, disminueix la probabilitat d'introduir errors i la seva localització és més senzilla.

De forma genèrica una funció necessita un nom (*identificador*), amb el que ens referirem des de la resta del programa, les dades que manipularà (*arguments d'entrada*) i el que retornarà (*argument de sortida)*. Aquest ***prototipus*** de la funció s'especifica en la seva ***declaració.***

1.1 Declaració

De la mateixa manera que es necessari declarar totes les variables, també s'han de declarar totes les funcions abans d'utilitzar-les. La declaració de les funcions mitjançant els prototipus sol fer-se al principi del fitxer, després dels **#define** i **#include**.

En molts casos –particularment en programes grans amb moltes funcions–, es pot crear un fitxer (amb l'extensió **.h**) amb tots els prototipus de las funcions utilitzades i incloure'l amb un **#include** en tots els fitxers en que s'utilitzin aquestes funcions. La sintaxis genèrica és:

Sintaxis:	**Exemple:**
tipus_valor_retorn nom_funció (llista tipus arguments);	char Pes_ideal (int, float);

Descripció:

tipus_valor_retorn és el tipus de les dades que retorna la funció. Si la funció no retorna cap valor, com a tipus_valor_retorn es posa '***void***'.

nom_funció és el nom que s'utilitzarà per referir-se a ella des de la resta del programa

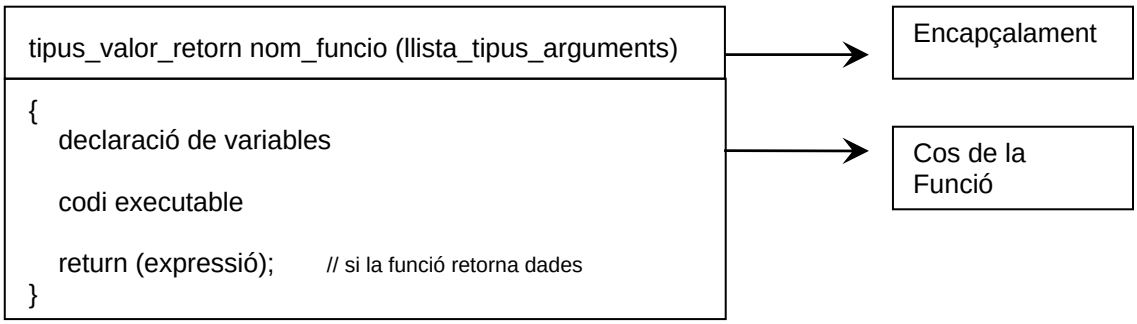
llista_tipus_arguments són els tipus de cadascun dels arguments separats per comes. Si la funció no rep cap argument, entre els parèntesis es posa '***void***'.

Important! – No us oblideu de posar el punt i coma al final de la declaració.

1.2 Definició d'una Funció

La **definició d'una funció** consisteix en l'escriptura del codi de la funció. La seva estructura es la mateixa que la de la funció **main:**

Sintaxis:



Exemple:

```
char  Pes_ideal (int altura, float coeff)
{
   char res;

   //sentències

  return res;
}
```

Descripció:

La primera línia o ***encapçalament*** (header) ha de coincidir amb el prototipus especificat a la declaració de la funció en el nom de la funció i el tipus dels arguments. L'única diferència és que en la **llista_tipus_arguments** hem d'especificar el nom de les variables on es guardaran aquests arguments.

La sentència ***return*** retorna l'execució al punt del programa que ha cridat a la funció. Si el ***return*** va seguit d'una *expressió o variable*, es retorna el seu valor**.**

Important - Si volem fer ús d'aquest valor al programa principal hem de guardar-lo en una variable.

Exemple> El programa “Pes Ideal”

Programa 1.

```
main()
{
  char edat,res;
  float coeff, rate;
  int altura;


 printf(“Digues el teu grup d’edat:\n”);
 printf(“ a. 0-15       b. 15-30      c. +30\n”);
 scanf(“%c”,&edat);
 printf(“La teva alçada en centímetres:\n”);
 scanf(“%d”,&altura);

 switch (edat)
{
case ‘a’:
     coeff=1.5;
     rate=coeff*altura/100;
     if (rate>5 )
        res=’s’;
     else
        res=’n’;
     break;

case’b’:
     coeff=1;
     rate=coeff*altura/100;
     if (rate>5 )
        res=’s’;
     else
        res=’n’;
     break;

default:
     coeff=0.5;
     rate=coeff*altura/100;
     if (rate>5 )
        res=’s’;
     else
        res=’n’;
     break;
}

if (res==’s’)
     printf(“Estàs gras”);
}
```

Programa 2.

```
char  Peso_ideal (int, float );  //declaración

main()
{
 char edat,res;
 float coeff, rate;
 int altura;

 printf(“Digues el teu grup d’edat:\n”);
 printf(“ a. 0-15       b. 15-30      c. +30\n”);
 scanf(“%c”,&edat);
 printf(“La teva alçada en centimetros:\n”);
 scanf(“%d”,&altura);

  switch (edat)
 {
    case ‘a’:
        coeff=1.5;
        res=Pes_ideal(altura,coeff);
        break;

    case ‘b’:
        coeff=1;
        res=Pes_ideal(altura,coeff);
        break;

    default:
        coeff=0.5;
        res=Pes_ideal(altura,coeff);
        break;

 }

  if (res==’s’)
     printf(“Estàs gras”);

 }


 char  Pes_ideal (int altura, float coeff)
 {
  float rate;
  char resultat;

   rate=coeff*altura/100;
   if (rate>5)
       resultat=’s’;
   else
       resultat=’n’;

   return resultat;
   }
```

Exercici #1> Definiu funcions que facin el mateix que el codi en negreta i modifiqueu els programes perquè cridin a les funcions en substitució del codi en negreta.

a)
```
void main( )
  {
     float a,b, div;

      printf ("Introdueix dos números \n");
     printf ("a1=");
     scanf ("%f",&a);
     printf ("a2=");
     scanf ("%f",&b);

     if (b!=0) div=a/b;
     else div = 0;
     printf("La divisió es: %f", div);
  }
```

b)
```
void main( )
{
   int x,i;

   i=5;

   do{
      printf ("introdueix un número entre 1 i 10\n");
      scanf("%d",&x);
    } while ((x<1) ||  (x>10));

    if(x==i)
         printf("Has encertat!!!"\n);
}
```

Exercici #2> Feu un programa que donats dos números, a (enter) i x (float), introduïts per l'usuari, mostri el següent càlcul:

$$(2x + a(x+1))/(x-1)$$

El càlcul l'ha de fer una funció de prototipus: *float funcio (float, int);*

2. Variables locals, variables globals i arguments d'una funció

Els arguments amb els que es crida a la funció són copiats a les variables especificades a la definició de la funció. Les variables que hi ha a la funció ***main()*** i les variables d'una funció són **variables diferents**. Per tant, **els canvis sobre les variables que realitza la funció no es transmeten a las variables (originals) del programa que ha cridat a la funció.** Veiem un exemple.

Exemple> Modificació dels valors de les variables d'un programa.

```
float valor_abs(float );

main()
{
  float x,i;              A
  i=2 ;                   B
  x=valor_abs(i) ;        C
  x=x+1;                  G
}

float valor_abs(float i)
{
   float x;

   if(i<0) x=-i;          D
   else x=i;

   i=x+2;                 E

   return i;              F
}
```

	VARIABLES DEL MAIN	***VARIABLES DE FUNCIONS AUXILIARS***
A	*Declaració variables:* *i= ?? x= ??*	
B	*Inicalització de i:* *i= -2 x= ??*	
C	*Crida a valor_abs amb el valor de i:* *i= -2 x= ??*	*Inici de la funció valor_abs:* *i= -2 x= ??*
D	*i=-2 x= ??*	*i=-2 x= 2*
E	*i=-2 x= ??*	*i= 0 x= 2*
F	*i= -2 x= 2*	*Tornem al main() retornant x*
G	*i= -2 x= 3*	

Exercici #3> Debuga i comprova el valor de totes les variables del següent programa. Per entrar dins de la funció utilitzeu l'opció ***Step Into*** (F11) del Debugger quan esteu a sobre de la línia del codi que fa la crida a la funció.

```
#include <stdio.h>

double valor_abs(double); // declaració

void main (void)
{
        double z, i;

        i = -30.8;
        z = valor_abs(i) + i*i; // crida a la funció dins d'una expressió
}


double valor_abs(double x)
{
        double absX;

        if (x < 0.0)
                absX = -x;
        else
                absX = x;

        return absX;
}
```

En mode Debugger, podem veure el valor de les variables locals de cada funció. Apareixen a la pestanya ***Locals*** (*figura 1*). Al quadre ***Context*** ens apareix el nom de la funció a la qual pertanyen les variables i se'ns permet canviar a qualsevol altra funció que tingui variables en memòria. Utilitzeu aquest recurs per comprovar que quan entrem a la funció ***valor_abs*** tenim definides dues variables 'x', 'i', però associades a contextos diferents. Fixeu-vos que després de retornar al ***main( )***, les variables que pertanyen a la funció ***valor_abs*** han deixat d'existir. Són variables locals a la funció.

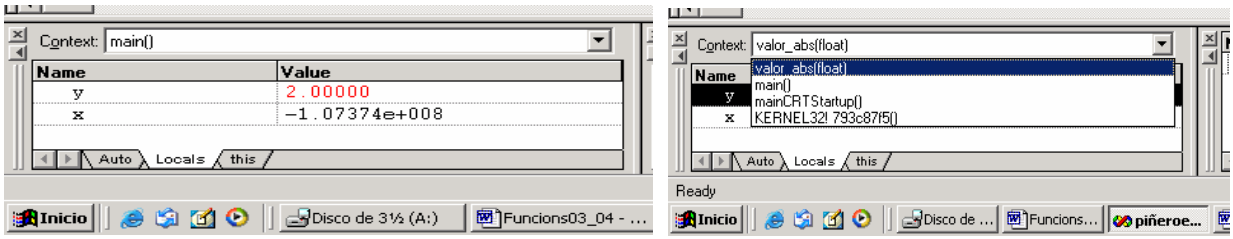


Figura 1. Variables locals

Acabem de veure que les variables de les funcions auxiliars del nostre programa només tenen existència dins del cos de la funció. Les úniques que sobreviuen al llarg de tot el programa són les variables del main, però el valor de les variables declarades dins d'una funció només es poden utilitzar en la funció que les ha declarat.

Si volem que una variable pugui ser usada per qualsevol funció, hem de fer una declaració global, això és, fora del main. Ara bé, cal tenir en compte una cosa: UTILITZAR VARIABLES GLOBALS ES MOLT PERILLÓS PERQUÈ EL SEU ÚS INDISCRIMINAT POT PROVOCAR EQUIVOCACIONS QUE US GENERIN ERRORS EN TEMPS D'EXECUCIÓ.

3. Altres exercicis

Exercici #4> Feu una funció que, donats dos números, els compari i informi a l'usuari si el primer número és major o menor que el segon. No mostreu cap d'ells. La funció ha de retornar TRUE si els dos números coincideixen i FALSE en cas contrari. Comproveu el seu bon funcionament.

Exercici #5> Feu una funció que donat un número màxim de jugades N, i un número secret, doni a l'usuari un màxim de N oportunitats per endevinar el número secret. Retornarà 's' o 'n' segons s'hagi endevinat el número secret o no. Utilitzeu la funció de l'exercici anterior.

Exercici #6> Modifiqueu el joc de l'exercici 5. L'usuari podrà jugar tants cops com vulgui, però en cada nova partida, podrà escollir el número d'oportunitats que pot tenir.

Exercici #7> Escriure un programa que llegeixi les dades de dues matrius de 3 files i 3 columnes, les sumi i escrigui el resultat. Utilitzeu tres funcions, una per llegir les dades, una altra per sumar les matrius i una tercera per escriure la matriu resultat.

Exercici #8> Definiu globalment l'estructura de CD's de la pràctica anterior, així com l'array de 10 CD's. Modifiqueu el programa de la pràctica anterior afegint les següents funcions:

- Una per l'entrada de dades dels CD's.
- Una altra que englobi el cos del while. Haurà de llegir el grup i comparar. La resposta serà retornada al main.