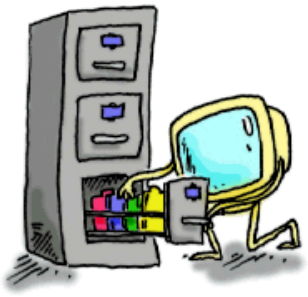
Tema 5: Programació estructurada



5.1 Programació estructurada i modular (I)

- Programació modular → Un programa consta de la unió de mòduls independents.
- Programació estructurada → Cada mòdul es construeix utilitzant mètodes estructurats.

Programació estructurada

- La programació estructurada es fonamenta en l'ús exclusiu de tres estructures de control: seqüència, selecció i iteració.
  - Un programa estructurat no utilitza sentències de salt incondicional. Exemple: goto
- Ús de recursos abstractes →
- Permet descomposar accions complexes en funció d'instruccions (accions simples) que són executades per la computadora
- Disseny top-down →
- Tot problema pot descomposar-se en nivells successius on cada nivell següent (inferior) és un refinament de l'anterior.
- Estructures bàsiques →
- Tal com demostraren Böhm i Jacopini (1966) emprant estructures seqüencials, selectives i iteratives es pot escriure qualsevol programa. A més, aquest programa compleix les característiques de tenir:
  - Un únic punt d'entrada i un de sortida (o fi) per control del programa.
  - Existeixen camins des de l'entrada del programa a la sortida que es poden seguir i que passen per totes les parts del mateix.
  - Totes les instruccions són executables i no existeixen bucles infinits (sense fi).

5.1 Programació estructurada i modular (II)

Programació modular

- La programació modular ajuda a construir programes 'ben fets': correctes (donen el resultat esperat), llegibles (per qualsevol programador), modificables i depurables.
  - Un mòdul és un conjunt d'instruccions que realitzen una tasca específica i tenen una interfície ben definida: un punt d'entrada i un de sortida
  - Interfície és el conjunt d'entitats necessàries per a utilitzar el mòdul
  - En un disseny modular és important descompondre el problema en subproblemes.
  - En la major part de llenguatges això passa per implementar el programa emprant funcions i procediments (subrutines): tots dos realitzen una tasca determinada retornant el control a qui els ha cridat.

Exemple de programació estructurada:

1. Posar el títol (comentari) del programa.
2. Directives del processador.
3. Explicar (capçalera) què fa el programa.
4. Definició de variables.
5. Declaració de constants numèriques.
6. Declaració de funcions.
7. Funció principal (main()).
8. Altres definicions (funcions i procediments).

Acompanyar el programa amb els comentaris que calguin

Exemple d'estructura modular simple.

Mòdul principal
- → B.1. Mòdul impressió de capçalera.
- → B.2. Mòdul de procés de dades.
- → B.3. Mòdul d'impressió.

5.1 Programació estructurada i modular (III)

Exemple

- Programa que calcula la hipotenusa d'un triangle rectangle.

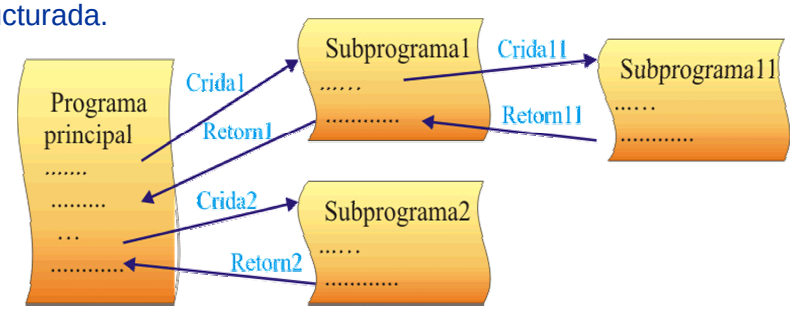
```
//Algorisme hipotenusa
/*Aquest programa calcula la hipotenussa
d'un triangle.
Consta del programa principal i d'una funció (retorna un
valor real) que calcula la hipotenusa*/
Real calculH(Real a, b)        //Declaració funció
Real x, y, h                   //Declaració variables
Inici                          //Programa principal
    Escriure("Programa calcul hipotenusa")
    Llegir(x,y)
    h <- calculH(x,y)
    Escriure(h);
Fi
Real calculH(Real a, b)    //Definició funció
Inici
    Tornar (arrel_quadrada(a^2+b*2))
Fi
```

```
//Algorisme hipotenusa
/*Aquest programa calcula la hipotenussa
d'un triangle.
Consta del programa principal i d'una funció (retorna un
    valor real) que calcula la hipotenusa*/
float calculH(float , float )      //Declaració funció
float x, y, h                      //Declaració variables
void main()                        //Programa principal
{
    printf("Programa calcul hipotenusa\n")
    printf("        Entra x i y: \n")
    scanf("%f%f", &x, &y)
    h = calculH(x,y)
    printf("        h = %f\n", h);
}
float calculH(float a, float b)    //Definició funció
{
    return (arrel_quadrada(a^2+b*2))
}
```

- Es pot observar clarament que, en la realització de la funció, hi ha tres etapes:
  - Declaració
  - Definició
  - Crida a la funció

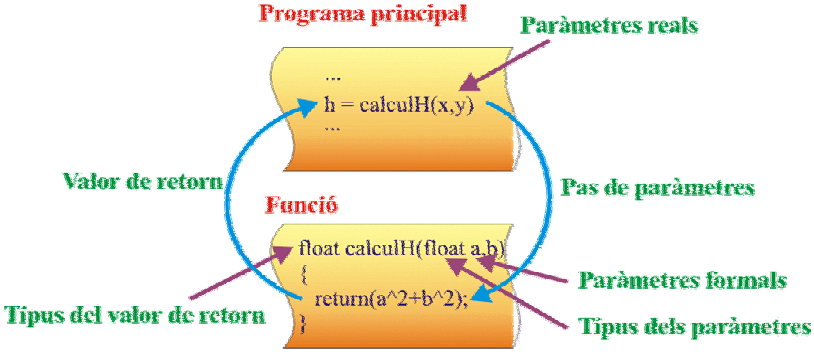
5.2 Procediments i funcions

- Disseny descendent o top-down →
  - És una forma de resoldre problemes complexos: divideix i conqueriràs
  - Al conjunt de subprogrames que resolen el problema principal se'ls anomena funcions i procediments
- Funcions i procediments →
  - Compleixen els axiomes prèviament descrits en la programació estructurada:
    - Tots els blocs (funcions i procediments) tenen un únic punt d'entrada.
    - Tots els blocs tenen un únic punt de sortida.
    - Quan s'arriba al final d'un bloc es continua amb un altre o acaba.
    - Un programa pot tenir diferents nivells de subprogrames:
  - En l'exemple anterior es fa ús de les propietats de la modularitat i la programació estructurada.



5.3 Funcions

- És una operació que rep un o més valors, anomenats arguments, i genera **un** valor de retorn anomenat resultat
  - Certs llenguatges (com el C) requereixen de la declaració prèvia de la funció abans del seu ús.
- La funció s'ha de declarar i definir. La declaració s'ha de fer previ crida de la mateixa.
- S'ha d'especificar el tipus de variables dels arguments i del resultat.
  - Exemple: com en l'exemple del càlcul de la hipotenusa.
- Paràmetres formals i reals.
  - Els paràmetres reals són els paràmetres que s'utilitzen en el càlcul en l'algorisme.
  - Els paràmetres formals són els emprats en la definició de la funció.
  - Les variables emprades pels paràmetres formals i els reals no tenen per què ser les mateixes.



5.4 Procediments (I)

- És un subprograma que executa un procés determinat.
- Es diferencia de les funcions en:
  - Les funcions retornen un valor. Els procediments en poden retornar zero, un o més .
  - El nom de procediment no està lligat a cap tipus de valor .
- A l'igual que les funcions:
  - Hi ha els paràmetres formals i els reals.
  - Les variables emprades pels paràmetres formals i els reals no tenen per què ser les mateixes.
  - Donat que no retorna valor de forma explícita, en C, s'indica amb el tipus de variable void.
- Exemple: Cas típic d'una rutina d'impressió de resultats.
  - Un procediment (funció) es pot fer servir en múltiples ocasions (és una rutina).

```
void imprimir(char[]);                  //Declaració
void main()
{
    imprimir("Comencem... \n");         //Ús
    ...
    imprimir("...aqui s'ha acabat! \n"); //Ús
    imprimir("Adeu!!\n");               //Ús
}
void imprimir(char[20] cadena);         //Definició
{
    printf("%s\n", cadena);
}
```


5.4 Procediments (II)

- Exemple
  - El següent exemple ordena de major a menor tres nombres. Observar:
    - Com el programa principal només fa de controlador de successos, cridant als tres procediments llegir_xs (entrada de dades), ordenar (mòdul de procés) i escriure (sortida de dades). Al seu temps, ordenar crida al procediment canvi per a efectuar la tasca rutinària d'ordenar els valors de dues dades.
    - El pas de paràmetres posicional en els procediments ordenar i canvi. Sense canviar el nom ni l'ordre de les variables emprades efectuem la ordenació dels nombres (continguts de les variables)!.
    - x1, x2 i x3 són variables globals del programa. La variable intern (en el proc. canvi) és local.

```
//Algoritme ordenació
Enter: x1, x2, x3
Inici
        llegir_xs()
        ordenar(x1, x2, x3)
        escriure()
Fi
Procediment llegir_xs
Inici
        Llegir (x1)
        Llegir (x2)
        Llegir (x3)
Fi llegir_xs
Procediment ordenar (var int: X, Y, Z)
Inici
        Si (X<Y) Fer canvi (X, Y)
        Si (Y<Z) Fer canvi (Y, Z)
        Si (X<Y) Fer canvi (X, Y)
Fi ordenar
```

```
Procediment canvi (var Enter: A, B)
var Enter: intern
Inici
        intern ← A
        A ← B
        B ← A
Fi canvi
Procediment escriure_xs
Inici
        Escriure (x1)
        Escriure (x2)
        Escriure (x3)
Fi escriure_xs
```