INS MANUEL VÁZQUEZ MONTALBÁN

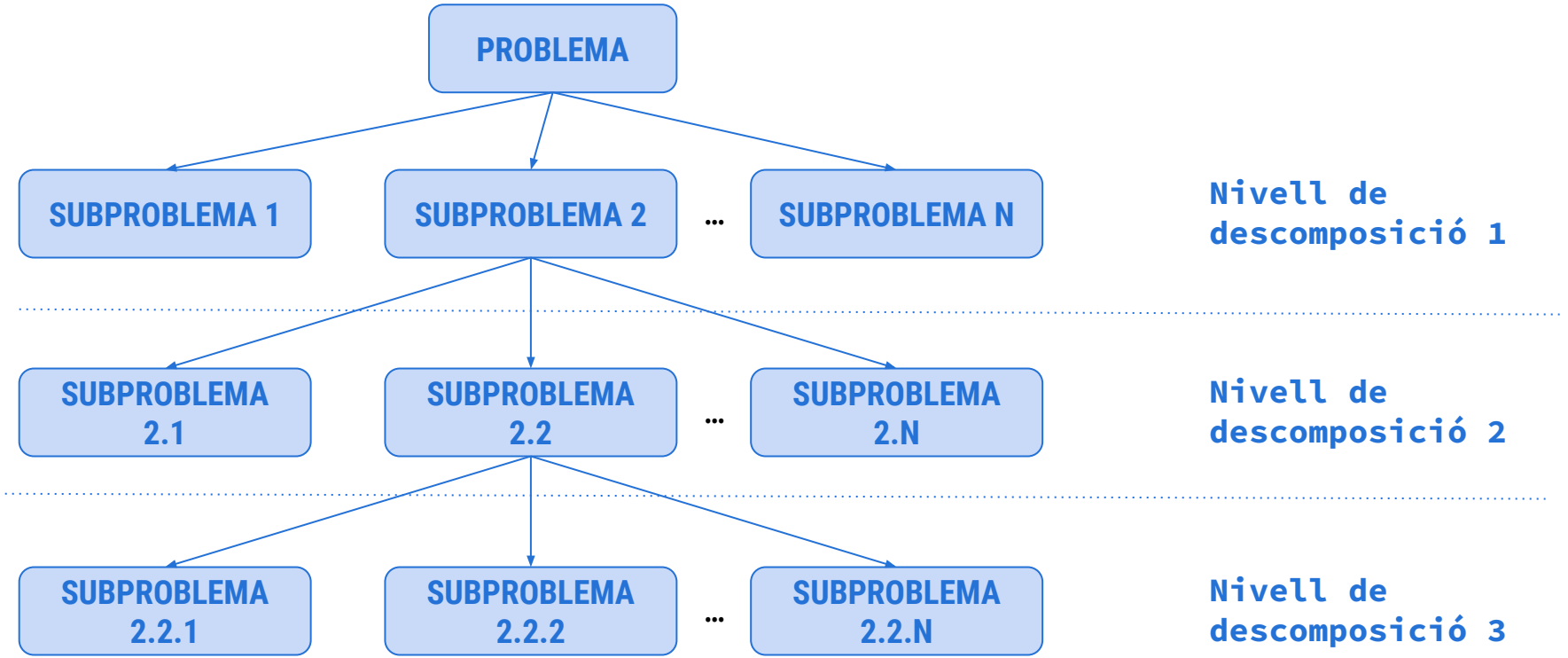
# M03. Programació bàsica

## UF2. Disseny modular

# 1. FUNCIONS

# Disseny modular. Descomposició del problema

Tal i com vam veure a la UF1, a l'hora de resoldre un problema, apliquem el disseny top-down (descendent).



# Què és una funció?

Un **bloc de sentències** que executa una tasca específica i al que fem referència mitjançant un **nom.**

En java, aquesta funcionalitat s'anomena **mètode.**

# Què s'ha d'especificar a l'hora d'escriure un mètode?

## Nom

La funció ha de tenir un nom per tal de poder-la cridar

## Cos

Conté les operacions (declaracions, condicions, iteracions) que ha d'executar la funció

## Arguments

Paràmetres que ha de rebre (o no) la funció per tal d'executar les operacions del cos

## Retorn

Cal indicar el tipus del resultat que retornarà (si en retorna) o si no retorna res

# Implementació del disseny modular

```
function
    var
        integer num1, num2
    endvar
    num1 = validate()
    num2 = validate()
    if num1 > num2 then
       write(“num1 és major”)
    else
       if num1 < num2 then
          write(“num2 és major”)
       else
          write(“Són iguals”)
       endif
    endif
endfunction
```

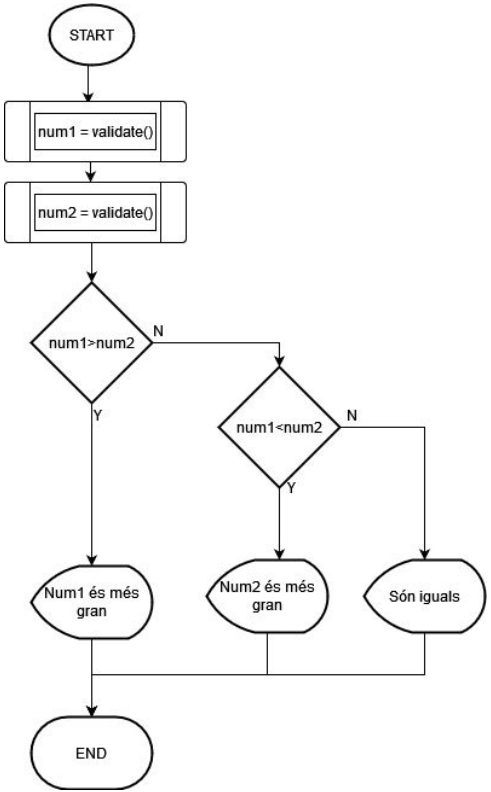
```
function validate()
    var
      integer num
    endvar
    do
      write(“Introdueix un nombre natural”)
      reaw(num)
    while num<1
    return num
endfunction
```

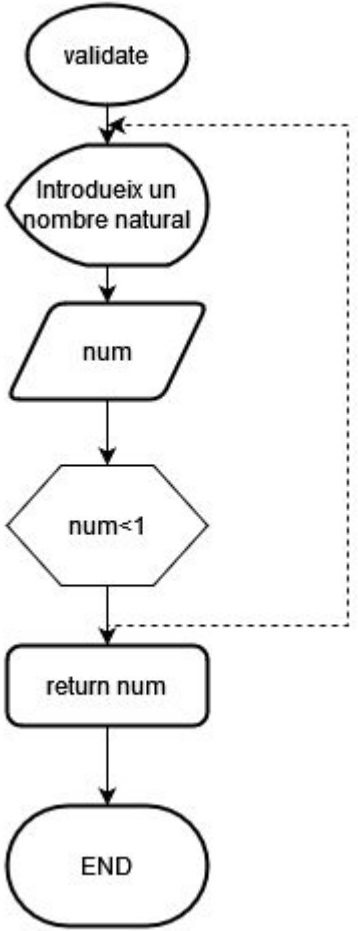
# Àmbits d'una variable

Una variable és **global** quan aquesta es declara en un programa fora de qualsevol bloc (és accessible des del seu punt de definició o declaració fins al final del codi font).
És a dir, existeix i té valor des del començament fins al final de l'execució del programa.

Una variable és **local** quan la seva declaració es fa dins d'un bloc (l'accés a aquesta variable queda limitat a aquest bloc i als blocs continguts dins d'aquest per sota del seu punt de declaració).
És a dir, és accessible només dins del bloc al qual pertany.

# Implementació del disseny modular



# Mètodes en Java

# Definició d'un mètode

```
tipus nomMetode (llistaArguments){
    (declaracions de variables locals);
    operacions (amb els paràmetres i variables locals);
    (retorn)
}
```

Les variables declarades en el cos de la funció són **locals** (només són accessibles dins el mètode).
El paràmetre *tipus* indica de quin tipus serà el valor retornat pel mètode (int, float, char, objectes,...)

# Definició d'un mètode

**`tipus nomMetode (llistaArguments)`**

```
void nomMetode ()
void nomMetode (int)
int nomMetode ()
int nomMetode (int)
```

La **declaració d'un mètode** (**prototipus**) indica:

- el nom del mètode
- quants arguments té i de quin tipus són
- el tipus del valor retornat

**Important! Podem declarar un mètode amb el mateix nom i diferent tipus d'arguments (overloading, OOP).**

# Invocació de mètodes. Exemple

```
public class Metodes {
    public int val1 = 3, val2 = 4;

    public static void main(String[] args) {
        Metodes method = new Metodes();
        System.out.println(method.sumar());
    }

    public int sumar(){
        return val1+val2;
    }
}
```

# Invocació de mètodes. Exemple II

```java
//mètode que retorna la suma dels dos arguments
public int sumar(int num1, int num2){
    return num1+num2;
}

//mètode que retorna la suma dels dos valors globals
public int sumar(){
    return val1+val2;
}
```

# Invocació de mètodes

Tenint en compte els exemples anteriors, quina de les opcions és millor a l'hora de declarar un mètode?

Cal tenir en compte els següents punts:

- **Àmbit**: els valors amb els que treballaran els mètodes seran locals o globals?
- **Retorn**: el resultat de les operacions executades dins del mètode ha de retornar al programa principal o a un altre mètode o no?

Un cop resolts els punts anteriors, podem defnir els nostres mètodes segons una de les 4 combinacions possibles.

# Exercicis I

**Exercici 1**

Crea un programa que demani un nombre per teclat i validi si aquest és natural.

**Exercici 2**

Crea un programa que demani per teclat un el radi d'una circumferència (nombre enter) i retorni la seva longitud.