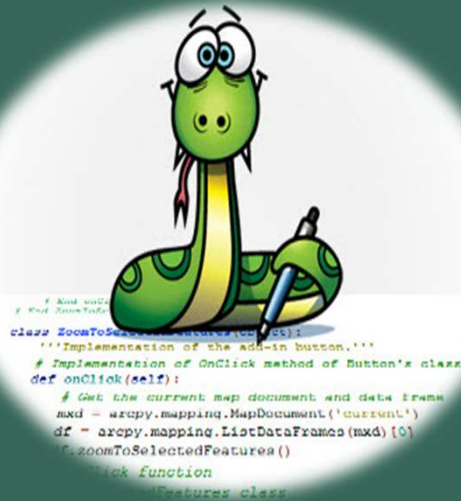


FUNCIONS

FUNCIONS, PROCEDIMENTS, FUNCIONS AMB PARÀMETRES



M^a Belén Tortosa Pedrón

INTRODUCCIÓ



- Una funció és un bloc de codi que pot ser executat les vegades que considerem necessàries.
- S'utilitzen quan necessitem que un programa executi el mateix bloc d'instruccions varies vegades.
- Existeixen dos tipus de funcions:
 - **Les creades pel programador:** Tenen el codi específic que nosaltres necessitem.
 - **Les que venen incloses amb Python:** Serveixen per crear procediments bàsics, com per exemple, convertir un **int** en una **string**.

PROGRAMACIÓ MODULAR



- Aquesta manera de treballar té moltes avantatges, doncs ens permet treballar amb problemes menors i més manejables.
- Tanmateix, a nivell tècnic ens caldrà introduir nous conceptes, que són els que es recullen sota el nom de **PROGRAMACIÓ MODULAR**. Precisament això fa referència a dividir un programa complex en MODULS menors. Les avantatges d'aquest paradigma de programació són moltes:
 - El codi és **més senzill de dissenyar** i escriure
 - Com a conseqüència directa, **és més senzill llegir codi**, i per tant també és **més fàcil de depurar** errors.
 - **Reutilització:** Un cop un mòdul està fet i verificat, el podem fer servir tantes vegades com vulguem i en qualsevol programa. Per tant necessitarem menys temps de desenvolupament i cometrem menys errors.
 - **Treball en equip:** Podem repartir millor la feina. Un programa pot desenvolupar-se de forma simultània en diferents mòduls, que poden ser assignats a equips diferents.
 - **No Redundant:** Evitem repetir trossos de codi.



**"COPY PASTE NO ES
TU AMIGO"**

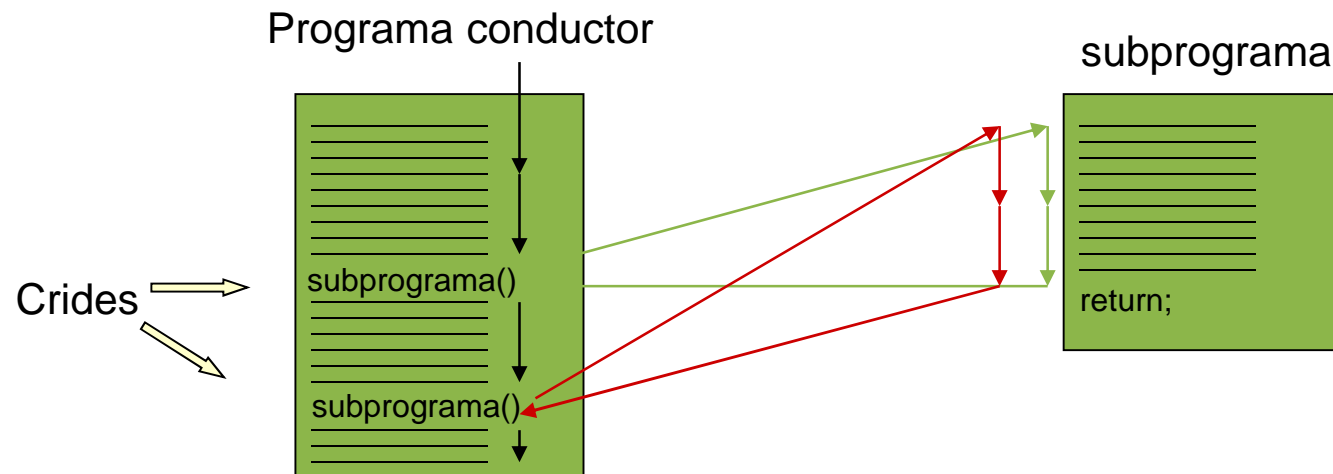


SUBPROGRAMES

■ Els subprogrames....

- es poden cridar tantes vegades com sigui necessari
- Mentre el subprograma està en execució, el programa conductor **espera** a que aquest acabi.
- El subprograma **retorna el control** al programa conductor un cop finalitza la seva feina i ho indica mitjançant la instrucció **return** o la finalització del codi.

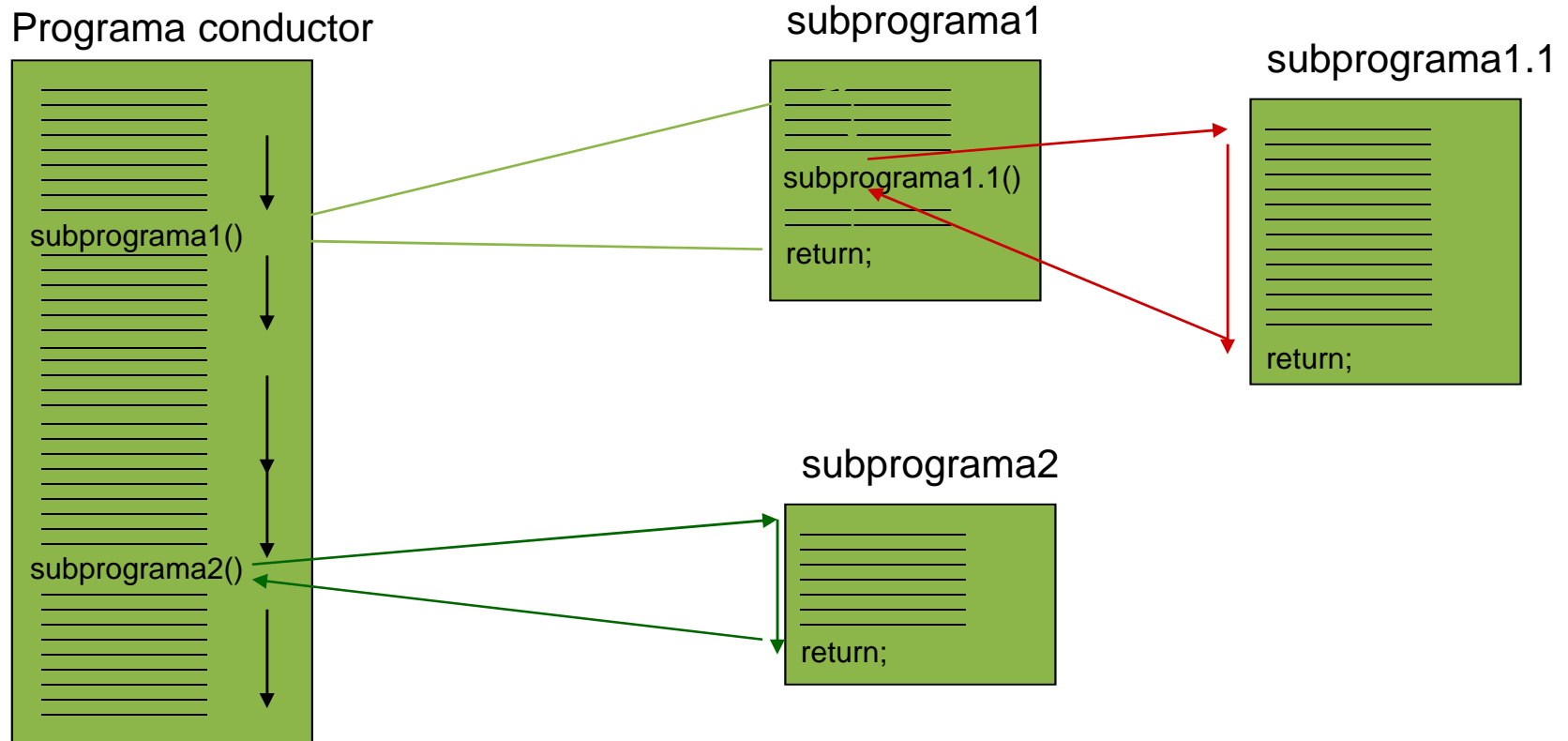
■ Observeu el fil d'execució quan un programa crida a un altre:



SUBPROGRAMES



- Des d'un programa o subprograma podem cridar a d'altres subprogrames:





SUBPROGRAMES: FUNCIONS I PROCEDIMENTS

Hi ha dos tipus de subprogrames:

Funcions:

■ Són aquells subprogrames que realitzen una sèrie de processos i finalment retornen **UN ÚNIC** valor al programa que les ha cridat. Típicament, la funció `suma(x,y)` accepta dos números, `y` i `x` com a *paràmetres*, i ens retorna el valor de la suma d'ambdós números.

```
def suma(a,b):  
    resultat=a+b  
    return resultat
```

Procediments:

Són aquells subprogrames que un cop són cridats realitzen una sèrie d'accions, i que no tornen cap resultat *de forma directa*.

Per exemple el procediment `imprimirMenu(idioma)` en C seria:

```
def imprimirMenu(idioma):  
    print("=====")  
    if(idioma==0):  
        print("1.- Llista de clients")  
        print("2.- Edició de clients")  
        print("3.- Sortir")  
    else:  
        print("1.- Lista de clientes")  
        print("2.- Edición de clientes")  
        print("3.- Salir")
```



FUNCIONS: DECLARACIÓ

```
def suma(a,b):  
    resultat=a+b  
    return resultat
```

Paraula
reservada per
declarar
funcions

Llista de paràmetres,
separats per comes.

Cadascun indica el
tipus que té i el nom.

return fa que es torni
un valor com a
resultat. Ha de ser del
mateix tipus que el
valor de retorn declarat

Les funcions han de tenir sempre
UN RETURN COM A MÍNIM en el
cos de codi.



FUNCIONS: CRIDA

- Per a cridar un subprograma, necessitarem:
 - indicar el nom del subprograma
 - posar entre parèntesis la llista **d'arguments**, que **ha de coincidir amb tipus i número amb els paràmetres declarats a la funció**.
 - En el cas de les funcions, recollir el resultat (si el necessitem)

Declaració amb tipus de retorn i paràmetres

```
def suma(a,b):  
    resultat=a+b  
    return resultat
```

```
res=0;
```

```
res = suma( 2, 1.5 );
```

```
res = res + suma( 3.1, 2.9 );
```

```
res = suma( res, res );
```

```
suma( 3, res );
```

arguments
de crida
constants

arguments
de crida
usant variables



FUNCIONS CREADES PER L'USUARI

- En Python, la definició de funcions es realitza mitjançant la instrucció **def** més un nom de funció descriptiu seguit de parèntesi d'apertura i tancament.
- La definició de la funció finalitza amb dos punt (:) i les instruccions que ha de realitzar la funció aniran identades.

```
def mi_funcio():  
    #Aqui s'escriben les instruccions
```

- Una funció no s'executa fins que no és invocada.
- Per invocar una funció, simplement es posa el seu nom i els parèntesis.
Exemple: **mi_funcio()**

EXAMPLE:



```
def mi_funcio():  
    print("Hola Mundo")
```

```
mi_funcio()
```

```
C:\_ C:\WINDOWS\system32\cmd.exe
```

```
Hola Mundo
```



FUNCIONS AMB RETORNO DE DADES

- Quan una funció fa un retorn de dades, aquest retorn, pot ser assignats a una variable:

```
def mi_funcio():  
    return "Hola Mundo"  
  
saludo=mi_funcio()
```

```
C:\WINDOWS\system32\cmd.exe  
Hola Mundo
```

- La paraula reservada que permet que una funció faci un retorn de dades és:
return



FUNCIONS AMB PARÀMETRES

- Un paràmetre és un valor que la funció espera rebre quan sigui invocada, per tal d'executar accions en base al mateix.
- Una funció pot esperar un o més paràmetres, els quals, aniran separats per comes.

```
def mi_funcio(nom, cognoms):  
    return "Hola "+ nom+" "+cognoms  
  
saludo=mi_funcio("Pol", "Linares")  
  
print(saludo)
```

C:\WINDOWS\system32\cmd.exe
Hola Pol Linares

- Els paràmetres, s'indiquen entre parèntesis, a mode de variables, d'aquesta manera els podem utilitzar a dintre de la funció.
- Els paràmetres que una funció espera, seran utilitzats per aquesta, a mode de variables d'àmbit local. Si utilitzem els seus paràmetres des de fora de la funció obtindrem un error.

```
def mi_funcio(nom, cognoms):  
    return "Hola "+ nom+" "+cognoms  
  
saludo=mi_funcio("Pol", "Linares")
```

`print(nom)`

C:\WINDOWS\system32\cmd.exe

```
Traceback (most recent call last):  
  File "I:\Curs2016-2017\M3\AWS1\material\codi\28_FuncionsPropies", line 7, in <module>  
    print(nom)  
NameError: name 'nom' is not defined
```



FUNCIONS AMB PARÀMETRES

- *Al cridar a una funció, sempre se li han de passar els seus arguments en el mateix ordre que els espera.*

```
def resta(a,b):  
    resultat=a-b  
    return resultat
```

```
x=int(input("Entra un valor: "))  
y=int(input("Entra un valor: "))  
print(resta(x,y))
```

EXERCICIS



I.1) Definiu i programeu una funció que, donats un preu i un percentatge de descompte, ens torni el preu amb el descompte aplicat. La funció té dos paràmetres: preu i percentatge. Retorna el preu amb el descompte aplicat.

I.2) Feu una funció “calculaResidu” a la que passem per paràmetre el dividend i el divisor, i torna com a resultat el residu de la divisió. **calculaResidu(dividend, divisor)**

I.3) Creeu una funció “llegeixla10” que s’encarregui de demanar a l’usuari que introdueixi pel teclat un nombre entre 0 i 10. Fins que el nombre no està entre el 0 i el 10, continua demanat a l’usuari el número.

La funció no té paràmetres, i retorna un nombre sencer, que és el que s’ha llegit a la funció.

I.4) Creeu una funció “**lecturaInterval**” que faci una lectura d’una xifra dins d’un interval. A la funció li passarem el valor mínim del rang, el valor màxim, i ens tornarà un valor llegit del teclat que estigui dins d’aquest rang. Com a l’exercici anterior, fins que el nombre no estigui dins l’interval, la funció continuarà demanat el nombre a l’usuari.

lecturaInterval(minim , maxim)

I.5) Creeu una funció “esDigit” que, donat un caràcter qualsevol, ens digui si és numèric o no. La funció rep un paràmetre **que representarà un únic caràcter**, i retorna un valor **entero** (1 si el caràcter és un número, 0 si no ho és). **esDigit(caracter)**



EXERCICIS

1.6) Creeu una funció “*esLletra*” que, donat un caràcter qualsevol, ens digui si és una lletra o no. La funció rep un paràmetre, i retorna un valor **entero** (1 si és una lletra, 0 si no ho és).

1.7) Feu una funció “*aMajuscula*” que, donat un caràcter qualsevol:

- a) ens el torni convertit a majúscula en el cas de ser una lletra
- b) el deixi igual si no és una lletra.

aMajuscula(caracter)

1.8) Volem fer un programa per a fer càlculs. El programa mostra un petit menú i cada opció ens demanarà un o dos números per a fer un càlcul determinat. Exemple:

Funcions: *math.sqrt(num)* y *math.log(num)*

```
opcion='x'
while(opcion!='T'):
    print("=====")
    print("Pren S per a SUMAR")
    print("Pren R per a ARREL QUADRADA")
    print("Pren L per a LOGARITME")
    print("Pren A per a AJUDA")
    print("Pren T per a TERMINAR")
    opcion=input()
    if opcion=='S':
        suma()
    elif opcion=='R':
        arrel()
    elif opcion=='L':
        logaritme()
    elif opcion=='A':
        ajuda()
```