Tema 1. Introducció a la programació

Marta Tarrés Puertas
mtarres@dipse.upc.edu
Departament de Disseny i Programació de Sistemes Electrònics
Universitat Politècnica de Catalunya - EPSEM

Resolució de problemes de programació

Program solving: Descripció de dades i quins resultats es volen obtenir a partir d'aquestes dades, i es demana un programa que expressi i calculi aquesta transformació. Propietats dels programes:

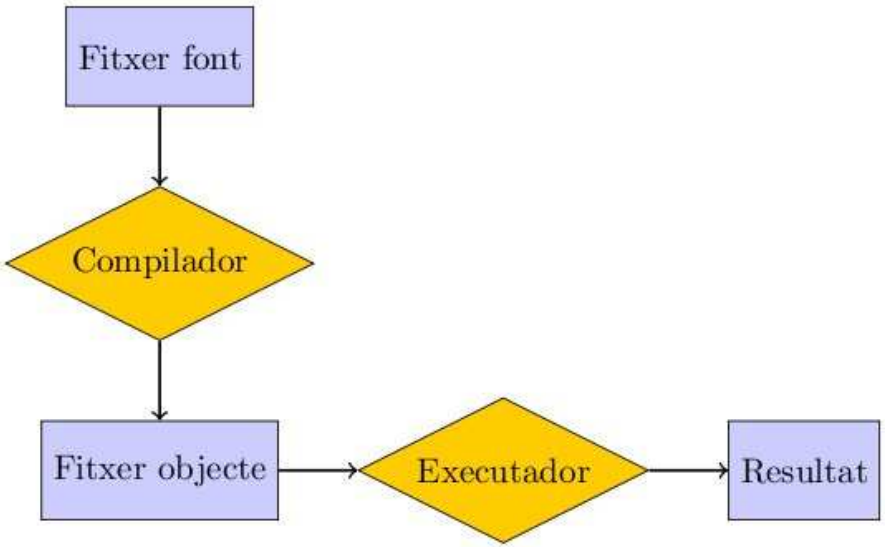
- Correctesa: un programa és correcte si per qualsevol valor possible de les dades, calcula un resultat correcte.
- No ambigüitat
- Finit

Llenguatges de programació i Python

1. Llenguatges de programació d'alt nivell (high-level languages)
2. Llenguatges de programació de baix nivell (low-level languages), llenguatge màquina (machine languages), llenguatge ensamblador (assembly languages)

Figura 1.1: Procés d'interpretació

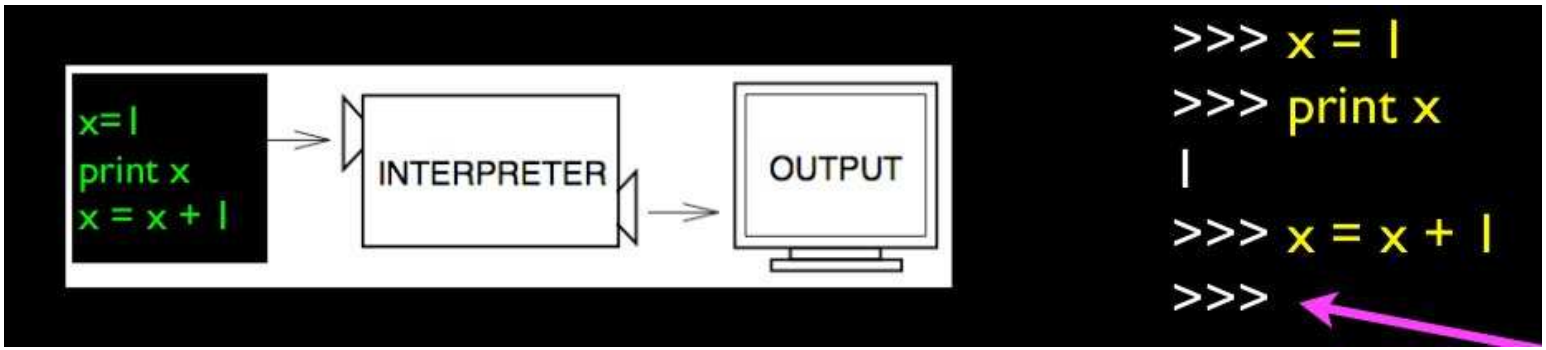
- intèrprets



Python

2 maneres d'utilitzar l'intèrpret de python

1. utilitzar el shell directament.
   http://datamech.com/devan/trypython/trypython.py



2. crear un script

Un exemple d'script: La divisió d'enters

```
divisor=input("Entra divisor: ")
dividend=input("Entra dividend: ")
quocient=divisor/dividend
residu=divisor%dividend
print "El quocient de la divisio es: ",quocient," i el residu es: ",residu
```

Bugs i debugging. Errors sintàctics.

Debugging: Cerca i eliminació d'errors en els programes

```
divisor=inpu("Entra divisor: ")
dividend=input("Entra dividend: ")
quocient=divisor/dividend
residu=divisor%dividend
print "El quocient de la divisio es: ",quocient," i el residu es: ",residu
```

Execució de l'script:

```
python exemple.py
Traceback (most recent call last):
  File "exemple.py", line 1, in <module>
    divisor=inpu("Entra divisor: ")
NameError: name 'inpu' is not defined
```

Errors semàntics.

```
divisor=input("Entra divisor: ")
dividend=input("Entra dividend: ")
quocient=divisor*dividend
residu=divisor%dividend
print "El quocient de la divisio es: ",quocient," i el residu es: ",residu
```

Execució de l'script:

```
python exemple.py
Entra divisor: 10
Entra dividend: 5
El quocient de la divisio es:  50  i el residu es:  0
```

Errors d'execució.

```
divisor=input("Entra divisor: ")
dividend=input("Entra dividend: ")
quocient=divisor/dividend
residu=divisor%dividend
print "El quocient de la divisio es: ",quocient," i el residu es: ",residu
```

Error d'execució divisió per zero:

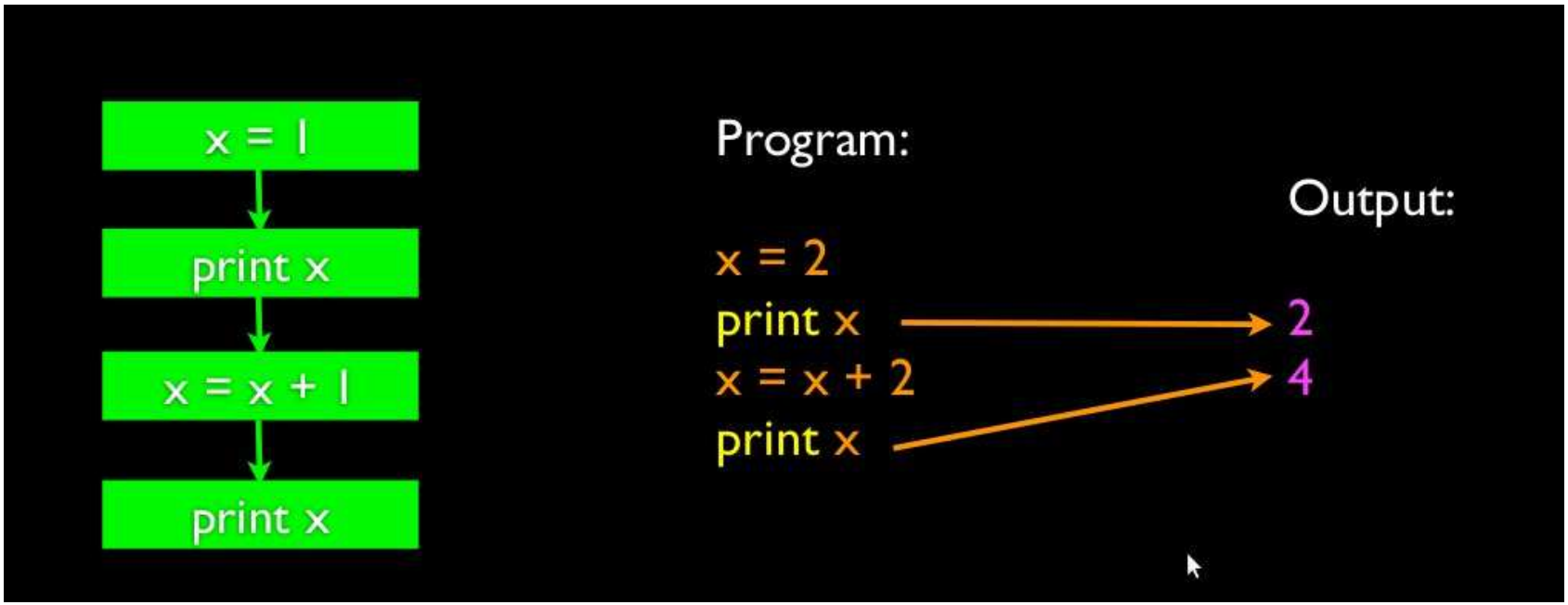
```
python exemple.py
Entra divisor: 20
Entra dividend: 0
Traceback (most recent call last):
  File "exemple.py", line 3, in <module>
    quocient=divisor/dividend
ZeroDivisionError: integer division or modulo by zero
```

Llenguatges de programació i elements

- Llenguatges naturals
- Llenguatges formals: Llenguatges de programació. Elements dels llenguatge:
  - Tokens
  - Regles sintàctiques: Parsing (anàlisi de l'estructura sintàctica)

Variables

- Les variables són contenidors de valors.
- Identificador de les variables
- Operador d'assignació
- El seu valor pot canviar al llarg del programa.



Paraules reservades (keywords)

```
and   del   for   is   raise
assert   elif   from   lambda   return
break   else   global   not   try
class   except   if   or   while
continue   exec   import   pass   yield
def   finally   in   print
```

Variables. Exemple I

```
print 'Lectura com a paraules'
numero=raw_input("Entra un enter ")
numero2=raw_input("Entra un real ")

print 'Conversio a nombres enter i a real'
num1=int(numero)
num2=float(numero2)

print 'Els nombres son: ',num1,' i ',num2,' la suma es: ',num1+num2
print numero
print numero2
print 'Resultat de concatenacio de paraules ',numero+numero2
```

Resultat d'execució de l'script:

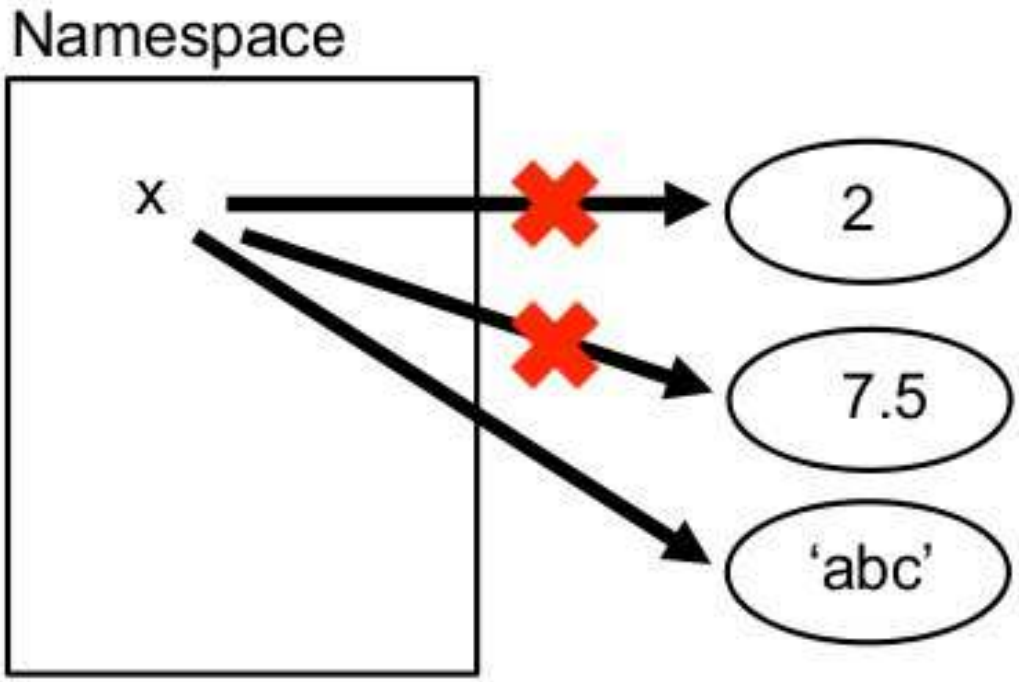
```
Lectura com a paraules
Entra un enter -5
Entra un real 4.89
Conversio a nombres enter i a real
Els nombres son:  -5  i  4.89  la suma es:  -0.11
-5
4.89
Resultat de concatenacio de paraules  -54.89
```

El valor de les variables pot canviar al llarg del flux d'execució.

```
x=8
x=x+3
x=-2
x=input("Introdueix nombre: ")
```

El tipus de les variables pot canviar al llarg del flux d'execució.

```
x = 2
x = 7.5
x = 'abc'
```



Assignació múltiple.

```
x, y = 2, 3      # assigns x=2 and y=3
print x, y       # prints 2 3
```

Tipus de les variables.

- `4/3` is 1 (integer division)
- `4.0/3.0` is 1.3333333 (float division)
- What is `4/3.0`?
  - No mixed-type operations. Must convert.
  - Python automatically converts to float.
    Thus 4 → 4.0 so the result is 1.3333333

Variables. Són correctes els següents noms de variables? Per què?

- x32y
- 32xy
- *ab
- is
- "plz
- a$3
- b78
- a_b
- _ab
- a-b
- a#b
- a?b
- a]b

Instruccions (statements)

- Expressions: Combinació de valors, variables i operacions
  - Integer (int)
    - addition and subtraction: `+`, `-`
    - multiplication: `*`
    - division
      - quotient: `/`
      - remainder: `%`
  - Floating point (float)
    - add, subtract, multiply, divide: `+`, `-`, `*`, `/`

Expressions numèriques

```
>>> xx = 2
>>> xx = xx + 2
>>> print xx
4
>>> yy = 440 * 12
>>> print yy
5280
>>> zz = yy / 1000
>>> print zz
5
```

```
>>> jj = 23
>>> kk = jj % 5
>>> print kk
3
>>> print 4 ** 3
64
```

```
      4 R 3
    ---------
5 | 23
    20
    ---
     3
```

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Power
%	Remainder

Ordre d'avaluació d'expressions

Per pensar-hi...

```
>>> print 10 / 2
5
>>> print 9 / 2
4
>>> print 99 / 100
0
>>> print 10.0 / 2.0
5.0
>>> print 99.0 / 100.0
0.99
```

Per pensar-hi...

```
>>> ddd = 1 + 4
>>> print ddd
5
>>> eee = 'hello ' + 'there'
>>> print eee
hello there
```

Assignació composta

```
y += x          # equivalent to y = y + x
```

Others

```
-=, *=, /=, %=
```

Quin és el resultat?

```
x, y = 2, 3
print x, y
x, y = y, x
print x, y
```

Quin és el resultat?

```
a = 5
b = 13
c = a+b
a = a-b
b = b-2*a+ c
```

Conversió de tipus

```
>>> sval = '123'
>>> type(sval)
<type 'str'>
>>> print sval + 1
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: cannot concatenate 'str' and 'int'
>>> ival = int(sval)
>>> type(ival)
<type 'int'>
>>> print ival + 1
124
>>> nsv = 'hello bob'
>>> niv = int(nsv)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int()
```

Quin és el resultat?

```
a = -5.3
b = int(a)
b = float(b)
a = a-b
b = str(b)
```

Tipus

a) Quin és el tipus de la variable *a* en acabar el programa següent?

```
a = 4
a = type(4)
a = "hello"
a = -7.3
a = type(``hello'')
```

b) Quin creus que és el valor de id(a), id(b) i id(45.3). Fes la prova.

```
a = 45.3
b = a
```

c) En Python, són iguals el objectes 14.007 i 14.0069? I 14.000007 i 14.0000069? Afegeix zeros després del punt i esbrina a partir de quants zeros els objectes són considerats iguals.

d) De quins tipus són les variables *x,y,z,t* i *v* després de les assignacions següents? Comprova-ho amb la funció "type

```
x = "43"
y = 43
z = 43.0
t = 43,0
v = 43.0e-1
```

Imprimir per pantalla missatges amb format

```
>>> print '%(language)s has %(#)03d quote types.' % \
          {'language': "Python", "#": 2}
Python has 002 quote types.
```

Documentació Python:
http://docs.python.org/release/2.4.4/lib/typesseq-strings.html

Traça d'un programa. I

Traça = Estat del programa en cada pas

```
q = input ("Entra les hores: ")
h = q % 24
q = q / 24
d = q % 7
s = q / 7
print "Aixo son", s, "setmanes", d, "dies i", h, "hores."
```

Traça d'un programa. II

Traça = Estat del programa en cada pas

```
q = input ("Entra les hores: ")
print "q =", q
h = q % 24
print "q =", q, "h =", h
q = q / 24
print "q =", q, "h =", h
d = q % 7
print "q =", q, "h =", h, "d =", d
s = q / 7
print "q =", q, "h =", h, "d =", d, "s =", s
print "Aixo son", s, "setmanes", d, "dies i", h, "hores."
```

Online Python Tutor: http://people.csail.mit.edu/pgbovine/python/

Què és una funció?

Funcions: Introducció intuïtiva

Pas 1: Definició de la funció.

```
def elevar_quadrat(x):
    print "El quadrat del nombre x es: "
    print x*x
```

Pas 2: Crida a la funció tantes vegades com sigui necessari

```
elevar_quadrat(100)
z=input("Entra nombre per elevar al quadrat: ")
elevar_quadrat(z)
elevar_quadrat(200)
a=input("Entra nombre per elevar al quadrat: ")
elevar_quadrat(a)
```

Definició de funcions. Crida a funcions

- Sintaxi definició de la funció amb indentació
  - Paràmetres
  - Variables locals
  - Resultat de retorn (funcions fructíferes)
- Invocació (crida) particular d'una funció
  - Arguments

Funcions: Introducció intuïtiva. Funcions fructíferes.

```
def elevar_quadrat(x):
    print "El quadrat del nombre x es: "
    return x*x

print elevar_quadrat(100)
z=input("Entra nombre per elevar al quadrat: ")
print elevar_quadrat(z)
print elevar_quadrat(200)
a=input("Entra nombre per elevar al quadrat: ")
print elevar_quadrat(a)
```

Funcions: Introducció intuïtiva

Exemple 1 (funció fructífera)

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

x=suma_nombres(4,8)
print x

print suma_nombres(suma_nombres(2,3),4) #Un argument de la crida a una funció pot ser el resultat
                                        #d'una crida a una funció fructífera
```

Exemple 1 variació (funció no fructífera)

```
def suma_nombres(a,b):
    resultat=a+b
    print resultat

suma_nombres(4,8)
```

Funcions: Introducció intuïtiva

```
def demanaTemperatura():
    t=raw_input("Introdueix temperatura en graus Celsius: ")
    return float(t)

def celsius2Farenheit(t):
    resultat=t*1.0+32.0
    return resultat

def mostra(c,f):
    print "temperatura celsius: ",c
    print "equivalent farenheit: ",f
    print

temperatura=demanaTemperatura()
farenheit=celsius2Farenheit(temperatura)
mostra(temperatura,farenheit)
```

Utilitzar funcions del llenguatge python. Exemple: llibreria math

```
import math

radi=raw_input("Introdueix el radi del cercle: ")
radif=float(radi)

circumferencia=2*math.pi*radif
area=math.pi*radif*radif

print

print "La perimetre del cercle es: ",circumferencia, ", i l'area es: ",area
```

Hi ha funcions de python que poden utilitzar-se sense fer un import (Built-in Functions Python).

```
a=input("Entra valor: ")
print abs(a)
print min(2,3,-8)
print max(4,2,5,122)
```

http://docs.python.org/library/functions.html

Ús 1 de funcions: Cridar a funcions dins el mateix script

Veure els exemples de les darreres transparències.

Ús 2 de funcions: Crear una llibreria de funcions i cridar-les des d'altres script

Script fun1.py

```
def comprovaPositiu(r):
    if r>0:
        print "Es positiu"
    else:
        print "Es negatiu o zero"

def comprovaParell(r):
    if r%2==0:
        print "Nombre parell"
    else:
        print "Nombre senar"
```

Script provaFuncions.py –> Importa les funcions de l'script fun1.py i realitza les crides que calguin.

```
from fun1 import *

dada=input("Entra valor: ")
comprovaPositiu(dada)
comprovaParell(dada)

comprovaPositiu(dada-8)

comprovaParell(pow(dada,2)) #Un argument pot ser el resultat d'una crida a una altra funció fructífera
```

Resultat d'execució de l'script provaFuncions.py

```
python provaFuncions.py
Entra valor: 13
Es positiu
Nombre senar
Es positiu
Nombre senar
```

Ús 3 de funcions: Crear una llibreria de funcions i cridar-les des del shell de python

Script fun.py

```
def suma2(a,b):
    return pow(a,2)+pow(b,2)

def suma3(a,b,c):
    return pow(a,2)+pow(b,3)+pow(c,2)
```

Importació de l'script de funcions des de l'intèrpret de python i crida a les funcions.

```
>>> from fun import *
>>> suma2(2,2)
8
>>> suma3(2,2,2)
16
>>> r1=input("Entra numero1 :")
Entra numero1 :24
>>> r2=input("Entra numero2 :")
Entra numero2 :10
>>> suma2(r1,r2)
676
>>> suma2(r1+2,r2)
776
```

Importació de funcions: from ... import * / import ...

La clàusula import. Permet que 2 o més llibreries de funcions tinguin funcions amb el mateix nom. Compte amb la sentència d'importació i en la crida a les funcions.

```
>>> import fun
>>> fun.suma2(2,3)
13
>>> fun.suma3(4,8,0)
528
```

Resum de continguts I

1. Assignacions
2. Tipus int, float, str
3. Variables: canvi de valor i tipus al llarg de l'execució del programa
4. Traces
5. Operadors, operands i operacions
6. Conversió de tipus (cast)
7. Ordre d'avaluació de les expressions
8. Funcions

Resum de continguts II

- Funcions
  - Capçalera
  - Llista de paràmetres
  - Bloc d'instruccions indentat
  - Variables locals
  - Funcions fructíferes
    - return
  - Crida a una funció
    - Llista d'arguments
    - Crida a funcions des del mateix script o des d'altres d'scripts
    - Crida a funcions des de l'intèrpret de comandes interactiu
    - Crida a funcions del llenguatge python
    - Clàusula from ... import * / Clàusula import ...
  - Llibreria de funcions
  - Composició de funcions (Laboratori: Una funció fructífera es passa com a argument d'una altra funció)
- Flux d'execució seqüencial

La Bíblia de l'assignatura.

- La "Bíblia de l'assignatura":
  - http://ocwitic.epsem.upc.edu/assignatures/inf/Apunts/introduccio-a-la-programacio.

En acabar aquest tema...

- Lectura acurada de La "Bíblia de l'assignatura" capítol Introducció a la programació:
  - http://ocwitic.epsem.upc.edu/assignatures/inf/Apunts/introduccio-a-la-programacio.
- Realització de tots els EOT Tema 1.
- Realització dels EOL Tema 1.
- Realització dels EOP Tema 1.
- Realització dels exemples de les transparències d'aquest tema.