



Jean-Louis BALAS

Maths-Sciences teacher
French T³

jl.balas@orange.fr



Teachers Teaching with Technology™

Abir MARINA

Maths teacher
French T³

abir.marina@ac-nancy-metz.fr



Allemagne

Belgique

Luxembourg

France

Suisse

Nancy

Limoges



Google

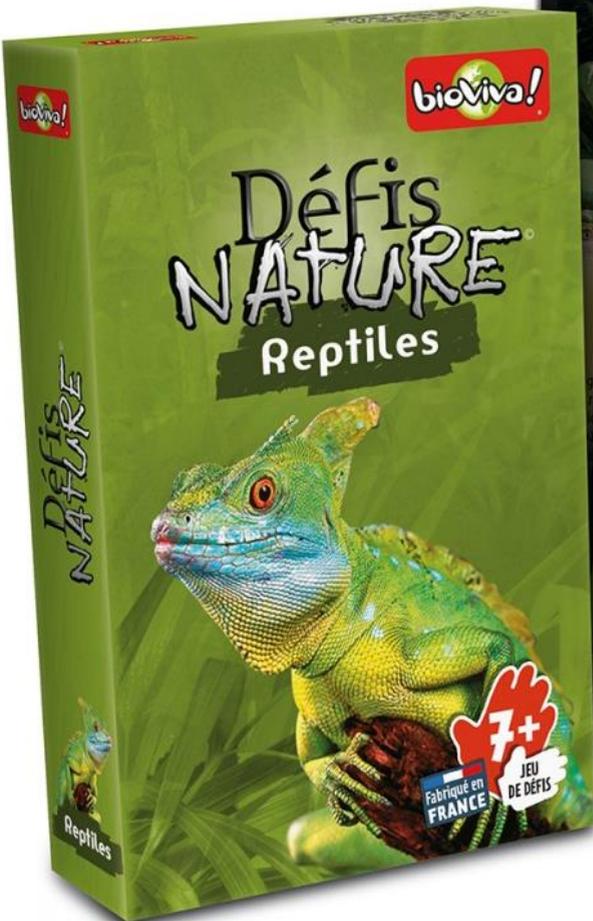
Thanks John !





TEXAS INSTRUMENTS

What is Python ?



What is Python ?



Guido van Rossum

- ✓ Born in 1991
- ✓ Completely free, complete and powerful language in many fields
- ✓ Syntax remains very simple and very readable
- ✓ Allows a modular and object-oriented approach to programming
- ✓ Learning object programming, system administration scripts or text file analysis, Web, realization of graphical user interfaces, scientific computing

What is Python ?



python



Tous Images Actualités Vidéos Shopping Plus Paramètres Outils

Environ 468 000 000 résultats (0,46 secondes)

Welcome to Python.org

<https://www.python.org/> Traduire cette page

The official home of the Python Programming Language.

Rechercher sur python.org



Download Python

Python 3.7.2 - Windows - Python
3.7.0 - Python 3.5.0 - Python 2.7.15

Python 3.7.2

Python 3.7.2 is the second
maintenance release of Python ...

Python For Beginners

BeginnersGuide -
BeginnersGuide/Download - IDEs -
Python Editors

Windows

Note that Python 3.5.5 cannot be
used on Windows XP or earlier ...

Python (langage) — Wikipédia

[https://fr.wikipedia.org/wiki/Python_\(langage\)](https://fr.wikipedia.org/wiki/Python_(langage))

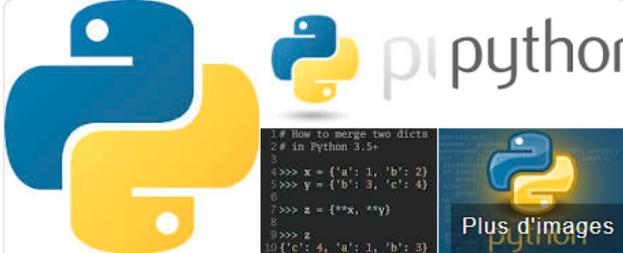
Python est un langage de programmation interprété, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et ...

Type: Fort, dynamique, duck typing

Auteur: Guido van Rossum

Extension de fichier: py, pyc, pyd, pyo et pyw

Développeurs: Python Software Foundation



Python
Langage de programmation

Python est un langage de programmation interprété, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Wikipédia

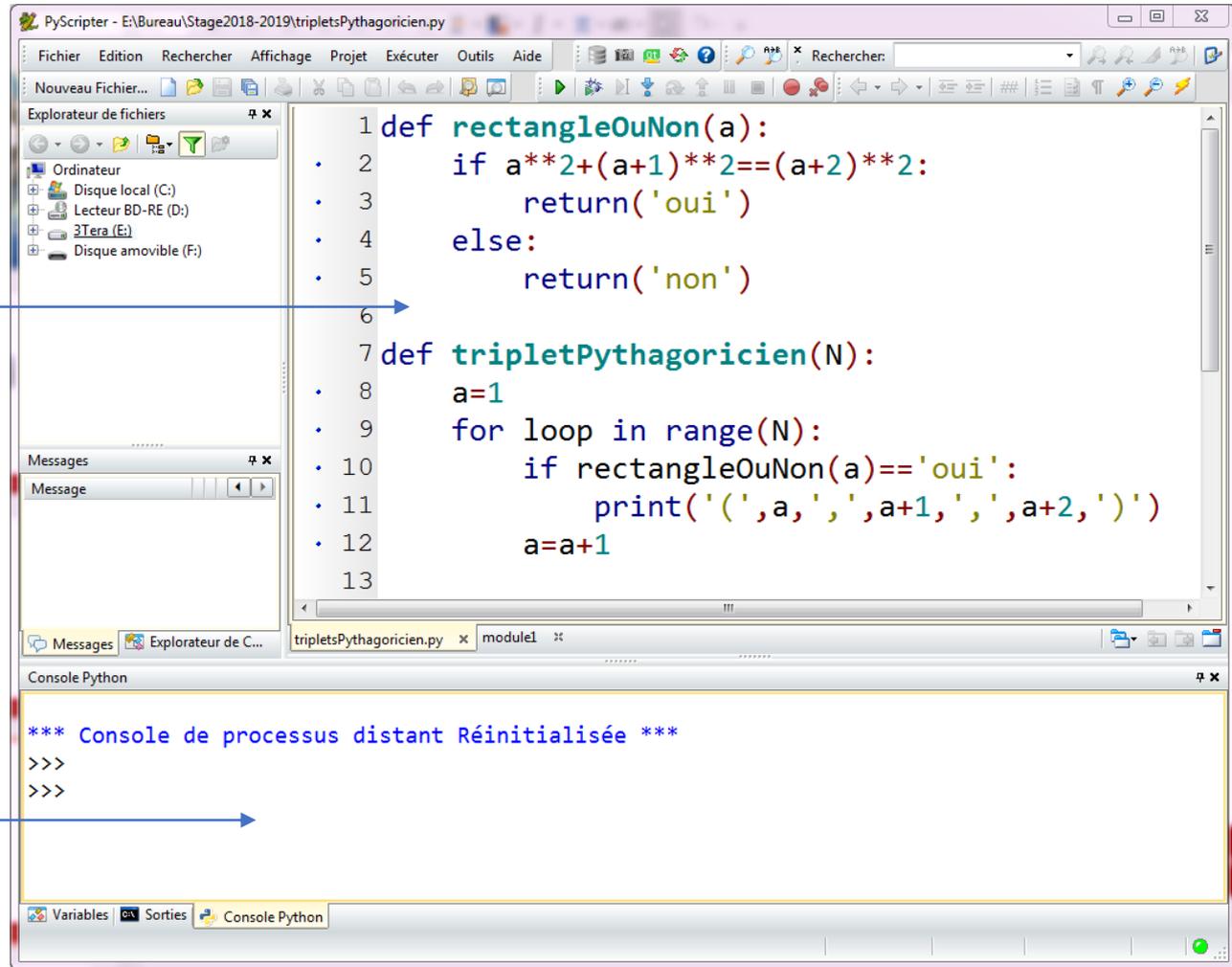
Conçu Par : Guido van Rossum

Date de première version : le 20 février 1991, il y a 27 ans

Dernière version : 2.7.15 (1^{er} mai 2018); 3.7.2 (24 décembre 2018)

Software

Script editor

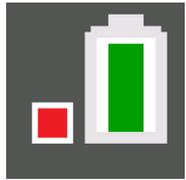


Shell

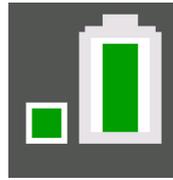
TI-Python adapter



TI-Python adapter

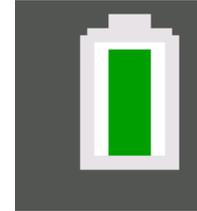


↑
Please wait...



↑
connected

```
NORMAL FLOTT AUTO RÉEL RAD MP
APPLICATIONS
8↑Periodic
9:PlvSmlt2
0:Prob Sim
█:PyAdaptr
:SciTools
:Transfrm
```



TI-Python app

File manager



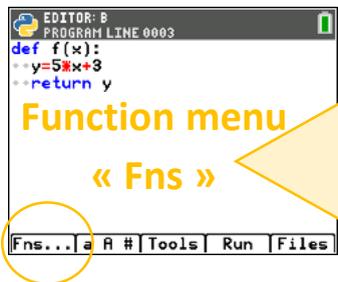
Editor



Shell



TI-Python app



The image displays six TI-Python app editor windows arranged in a 2x3 grid. Each window has a title bar 'EDITOR: B' and a menu bar with options: 'Func', 'Ctl', 'Ops', 'List', 'Type', 'I/O', and 'Modul'. The windows show the following content:

- Top-left:** A code snippet for a function: `1: def function():`
`2: return`
- Top-middle:** A code snippet for a loop and conditional: `1: if ..`
`2: if .. else ..`
`3: if .. elif .. else`
`4: for i in range(size):`
`5: for i in range(start, stop):`
`6: for i in range(strt, stp, step):`
`7: for i in list:`
`8: while condition:`
`9: elif :`
`0: else:`
- Top-right:** A code snippet for comparison operators: `1: x=y [sto]`
`2: x==y equal`
`3: x!=y not equal`
`4: x>y`
`5: x>=y`
`6: x<y`
`7: x<=y`
`8: and`
`9: or`
`0: not`
- Bottom-left:** A list of built-in functions: `1: []`
`2: list(sequence)`
`3: len()`
`4: max()`
`5: min()`
`6: .append(x)`
`7: .remove(x)`
`8: .insert(index, x)`
`9: sum()`
`0: sorted()`
- Bottom-middle:** A list of basic data types: `1: int()`
`2: float()`
`3: str()`
- Bottom-right:** A list of basic I/O and evaluation functions: `1: print()`
`2: input()`
`3: eval()`

TI-Python app

Maths library

The image shows three editor windows for the 'math module' and one for the 'random module'. Each window has a title bar with a Python logo and the text 'ÉDITEUR : AIREDEF' and the module name. The first window shows the 'math module' with a list of functions: Math, Const, Trig, fabs(), sqrt(), exp(), pow(x,y), log(x,base), fmod(x,y), ceil(), floor(), and trunc(). The second window shows 'Math', 'Const', and 'Trig' with 'e' and 'pi'. The third window shows 'Math', 'Const', and 'Trig' with 'radians()', 'degrees()', 'sin()', 'cos()', 'tan()', 'asin()', 'acos()', 'atan()', and 'atan2(y,x)'. The fourth window shows the 'random module' with 'random' and functions: random(), uniform(min,max), randint(min,max), choice(séquence), randrange(début, fin, pas), and seed(). A green arrow points from the 'Libraries « Modul »' text to the 'Modul' button in the first window.

```
ÉDITEUR : AIREDEF
math module
Math Const Trig
1:from math import *
2:fabs()
3:sqrt()
4:exp()
5:pow(x,y)
6:log(x,base)
7:fmod(x,y)
8:ceil()
9:floor()
0↓trunc()
Modul
```

```
ÉDITEUR : AIREDEF
math module
Math Const Trig
1:e
2:pi
Modul
```

```
ÉDITEUR : AIREDEF
math module
Math Const Trig
1:radians() degré→radians
2:degrees() radians→degré
3:sin()
4:cos()
5:tan()
6:asin()
7:acos()
8:atan()
9:atan2(y,x)
Modul
```

```
ÉDITEUR : AIREDEF
random module
random
1:from random import *
2:random()
3:uniform(min,max)
4:randint(min,max)
5:choice(séquence)
6:randrange(début, fin, pas)
7:seed()
Modul
```

The image shows the main interface of the TI-Python app. It has a menu bar with 'Func', 'Ctl', 'Ops', 'List', 'Type', 'I/O', and 'Modul'. The 'Modul' menu is open, showing 'math...' and 'random...'. A green arrow points from the 'Libraries « Modul »' text to the 'Modul' menu.

```
EDITOR: B
Func Ctl Ops List Type I/O Modul
1:math...
2:random...
Esc
```

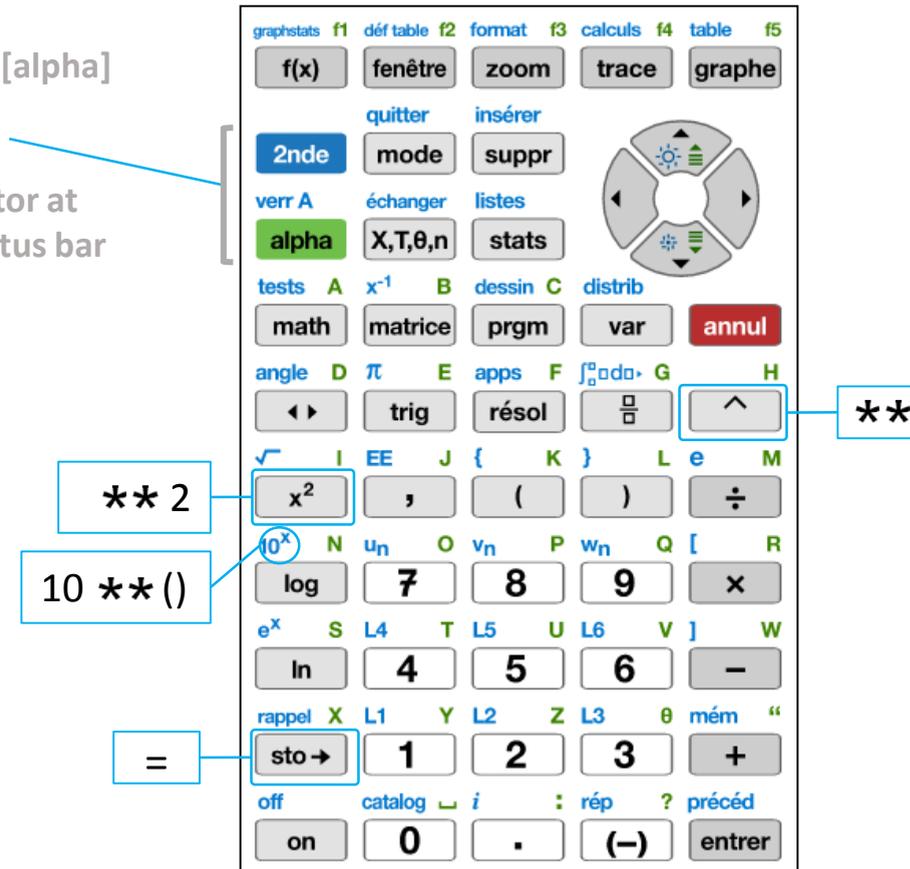
Libraries
« Modul »

Random library

TI-Python app

[2nde][alpha] & [alpha]

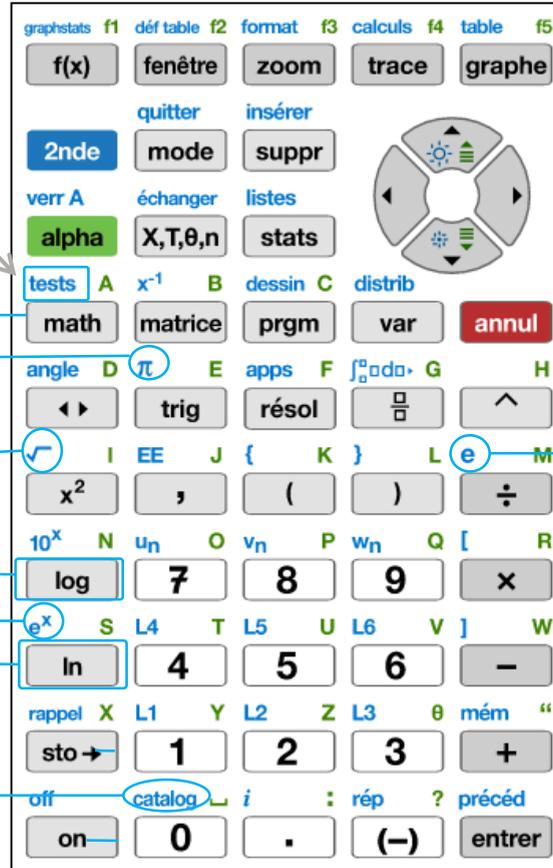
- abcde...
- ABCD...
- alpha indicator at cursor or status bar



TI-Python app

```
ÉDITEUR : FNS_EGL
E/S Type Ctl Ops Fonc List Modul
1: x=y [sto →]
2: x==y égal
3: x!=y différent de
4: x>y
5: x>=y
6: x<y
7: x<=y
8: and
9: or
0: not
Échap
```

```
EDITOR: SCRIPT01
I/O Type Ctl Ops Fonc List Modul
1: math...
2: random...
library
```



library

pi

sqrt()

log(x,10)

exp()

log(x)

all instructions

e

TI-Python app



```
PYTHON SHELL
>>> 34//13
2
>>> 34%13
8
>>>
>>> 5**3
125
>>> from math import *
>>> sqrt(13)
3.60555
>>> |
Fns... a A # Outils Éditer Script
```

```
PYTHON SHELL
>>> a,b=4,9
>>> a
4
>>> b
9
>>> |
Fns... a A # Outils Éditer Script
```

```
PYTHON SHELL
@ alpha
>>> c="TI Python"
>>> len(c)
9
>>> c[4]
'y'
>>> "abc"+"def"
'abcdef'
>>> |
Fns... a A # Outils Éditer Script
```

Functions

```
EDITOR: FONCTION
PROGRAM LINE 0003
def f(x):
  y=2*x+3
  return y
```

Fns... a A # Tools Run Files

- ✓ Use « return » not « print »
- ✓ The two points starts instructions
- ✓ Indent is automatic and necessary

```
PYTHON SHELL
>>>
>>> # Shell Reinitialized
>>> # L'exécution de CALC
>>> from CALC import *
>>> f(3)
9
>>> f(5)
13
>>> |
```

Fns... a A # Outils Éditer Script

- ✓ In the shell  will copy last line
- ✓ In the shell, you can call a function with

```
NORMAL FIXE9 AUTO RÉEL RAD MP
PROGRAM: CALC
:Effécran
:Fixe 2_
:Prompt X
:2*X+3→Y
:Output(5,5,"Y=")
:Output(5,7,Y)
```

distrib

var

Functions

Exercise 1

Rewrite this algorithm as concisely as possible using a function.

```
NORMAL FIXES AUTO REEL RAD MP
EDIT MENU: [a,1pha] [f5]
PROGRAM:ACT1
:Input "Xa ",A
:Input "Ya ",B
:Input "Xb ",C
:Input "Yb ",D
:(A+C)/2→I
:(B+D)/2→J
:Disp I,J
:
:
```

```
EDITOR: MILIEU
PROGRAM LINE 0005
from math import *
def milieu(xa,ya,xb,yb):
    xi=(xa+xb)/2
    yi=(ya+yb)/2
    return xi,yi
Fns... a A # Tools Run Files
```

```
PYTHON SHELL
>>> milieu(2,4,4,6)
(3.0, 5.0)
>>> |
Fns... a A # Outils Éditer Script
```

A function is a way to avoid rehearsal

Functions

Find the smallest integer n such as $c_n \leq 1$ with $c_0 = 3,4$ et $c_{n+1} = 0,8$

```
n ← 0
C ← 3,4
While C ≥ 1
    n ← n + 1
    C ← 0,8 × C
```

```
NORMAL FIXE2 AUTO RÉEL RAD MP
EDIT MENU: [a.Tpha.] [f5]
```

```
PROGRAM: SEUIL
: Effécran
: 0 → N
: 3.4 → C
: While C ≥ 1
: 1 + N → N
: 0.8 * C → C
: End
: Disp "N=", N
```

```
EDITOR: SEUIL
PROGRAM LINE 0007
def annee(c0, seuil):
    n = 0
    c = c0
    while c >= seuil:
        n = n + 1
        c = 0.8 * c
    return n_
```

Fns... a A # Tools Run Files

A function is a way to avoid rehearsal

Conditional statement

```
EDITOR: SI
PROGRAM LINE 0006
def min(a,b):
  if a<b:
    return a
  else:
    return b
-
Fns... | a A # | Tools | Run | Files
```

```
EDITOR: SI
TOOLS
Tools
1: Indent >
2: Indent <
3: Undo Clear
4: Insert Line Above
5: Cut Line
6: Copy Line
7: Paste Line Below
8: Go to Program Line...
9: Go to New Shell
0: Return to Shell
Esc
```

- ✓ « Then » doesn't exist in Python. Indent replace it.
- ✓ Indent close the instruction,

```
NORMAL FIXE2 AUTO RÉEL RAD MP
EDIT MENU: [a,1pha] [f5]
PROGRAM: SI
: Effécran
: Prompt A,B
: If A<B
: Then
: Disp "MIN=",A
: Else
: Disp "MIN=",B
: End
```

- ✓ Tools menu contains lots options like copy/paste or indent

Conditional statement

Exercise 2

A photo printing website offers prints at 0.11€ each. The price is reduced to 0.08€ each for orders of more than 200 photos.

Create an algorithm which gives the total price for a number n of prints.



```
NORMAL FIX9 AUTO REAL RADIAN MP
EDIT MENU: [a,Tp,q,] [f5]

PROGRAM:ACT2
:Input "Number ",N
:If N<200
:Then
:0.11*N→M
:Else
:0.08*N→M
:End
:Disp M
:█
```

```
EDITOR: SITE
PROGRAM LINE 0006

def photo(n):
    if n<200:
        M=0.11*n
    else:
        M=0.08*n
    return M_

Fns... a A # Tools Run Files
```

```
PYTHON SHELL

>>> photo(165)
18.15
>>> photo(314)
25.12
>>> |

Fns... a A # Outils Éditer Script
```

Closed loop

```
EDITOR: FOR
PROGRAM LINE 0005
def f(n):
    x=3
    for i in range(1,n+1):
        x=x+4
    return x

Fns... | a A # | Tools | Run | Files
```

```
EDITOR: FOR
Func Ctrl Ops List Type I/O Modul
1:if ..
2:if .. else ..
3:if .. elif .. else
4:for i in range(size):
5:for i in range(start,stop):
6:for i in range(strt,stp,step):
7:for i in list:
8:while condition:
9:elif :
0:else:
Esc
```

- ✓ Range (a,b) do the loop for $a \leq n < b$
- ✓ Range(n+1) are integers from 0 to n
- ✓ Indent start instructions

```
NORMAL FIXE2 AUTO RÉEL RAD MP
EDIT MENU: [a]pha [f5]
PROGRAM:FOR
:Effécran█
:Prompt N
:3→X
:For(I,1,N)
:X+4→X
:Disp "X=",X
:Wait 1
:End
```

- ✓ 4 choices

Take care of the length of the interval

Closed loop

Exercise 3

The population of a village is 2300 today. As the village is growing, its population increases each year by 150 inhabitants.

Design an algorithm which gives the number of inhabitants of this village in n years from today.



```
NORMAL FIXE2 AUTO RÉEL RAD MP
ÉDIT MENU: [a] [pha] [f5]

PROGRAM: ACT3
:Effécran■
:Input "Population ? ",N
:2300→P
:For(I,1,N)
:P+150→P
:End
:Disp P
```

```
ÉDITEUR : POP
LIGNE DU SCRIPT 0005
def population(n):
    p=2300
    for i in range(1,n+1):
        p=p+150
    return p_
```

```
PYTHON SHELL
>>> population(28)
6500
>>> |
```

Take care of the length of the interval

Open loop

- ✓ Indent starts instructions
- ✓ « return » stop the program, take care to the indent

```
NORMAL FIXE2 AUTO RÉEL RAD MP
EDIT MENU: [a] [pha] [f5]

PROGRAM:REBONDS
:Effécran
:Input "INITIAL HEIGHT (CM
) ",H
:0→R
:While H≥1
:3/5*H→H
:R+1→R
:End
:Disp "R=",R
```

- ✓ « tests » is accessible with the shortcut   permet de

Take care to the indent

La boucle non bornée

Exercise 4

On the first January 2018 the price of a new car was 20 000€. Each year the value of the car diminishes by 20%.

Write an algorithm which calculates the number of years which takes the value of the car to below 2000€.



```
NORMAL FIX9 AUTO REAL RADIAN MP
EDIT MENU: [a]Tpha] [f5]

PROGRAM:ACT4
:Input "Price ? ",V
:0→N
:While V≥2000
:0.8*V→V
:1+N→N
:End
:Disp N■
```

```
EDITOR: VOITURE
PROGRAM LINE 0006

def prix(v):
    n=0
    while v>=2000:
        v=0.8*v
        n=n+1
    return n

Fns... | a A # | Tools | Run | Files
```

```
PYTHON SHELL

>>> prix(20000)
11
>>> |

Fns... | a A # | Outils | Éditer | Script
```

Débugger

```
ÉDITEUR : BUGS1
LIGNE DU SCRIPT 0004
def somme(x,y)
z=x+y
return z

# faire l'appel somme(3,4) en console
# Faire l'appel somme('3',4) en console
# faire l'appel some(3,4) en console
# faire l'appel somme(3) en console
Fns... a A # Outils Exéc Script
```

```
ÉDITEUR : BUGS2
LIGNE DU SCRIPT 0007
def petitCheval():
....dé=randint(1,6)
....if dé==6:
....message='cheval sorti'
....else:
....message='cheval non sorti'
....return message

# faire l'appel petitCheval() en console
# faire l'appel petitCheval en console
Fns... a A # Outils Exéc Script
```

```
ÉDITEUR : BUGS3
LIGNE DU SCRIPT 0008
def sommeCarrés(N):
....S=0
....entier=1
....while entier !=N+1:
....S=S+entier*2
....entier=entier+1
....return S

# faire l'appel sommeCarrés(3) en console
# qui doit donner 12+22+32=14
Fns... a A # Outils Exéc Script
```

```
ÉDITEUR : BUGS1C0
LIGNE DU SCRIPT 0004
def somme(x,y):
....z=x+y
....return z

# faire l'appel somme(3,4) en console
# Faire l'appel somme('3',4) en console
# faire l'appel some(3,4) en console
# faire l'appel somme(3) en console
Fns... a A # Outils Exéc Script
```

```
ÉDITEUR : BUGS2C0
LIGNE DU SCRIPT 0009
from random import*
def petitCheval():
....dé=randint(1,6)
....if dé==6:
....message='cheval sorti'
....else:
....message='cheval non sorti'
....return message

# faire l'appel petitCheval() en console
Fns... a A # Outils Exéc Script
```

```
ÉDITEUR : BUGS3C0
LIGNE DU SCRIPT 0008
def sommeCarrés(N):
....S=0
....entier=1
....while (entier !=N+1):
....S=S+entier**2
....entier=entier+1
....return S

# faire l'appel sommeCarrés(3) en console
# qui doit donner 12+22+32=14
Fns... a A # Outils Exéc Script
```

Débugger

```
ÉDITEUR : BUGS4
LIGNE DU SCRIPT 0008
# calculs du N-ieme nombre trian
gulaire

def triangle(N):
    T=0
    for i in range(1,n+1):
        T=T+i
    return(T)

# faire l'appel triangle(10) en
console.
```

Fns... a A # Outils Exéc Script

```
ÉDITEUR : BUGS5
LIGNE DU SCRIPT 0006
def test():
    if 3*0,1==0,3:
        return('vrai')
    else:
        return('faux')

# faire l'appel test() en consol
e
```

Fns... a A # Outils Exéc Script

```
ÉDITEUR : BUGS6
LIGNE DU SCRIPT 0003
def f(x):
    return x**2-7*x+5

def dichotomie(a,b,precision):
    while (b-a)>precision:
        c=(a+b)/2
        if f(a)*f(c)<=0:
            b=c
        else:
            a=c
    return a,b
```

Fns... a A # Outils Exéc Script

```
ÉDITEUR : BUGS4C0
LIGNE DU SCRIPT 0008
# calculs du N-ieme nombre trian
gulaire

def triangle(N):
    T=0
    for i in range(1,N+1):
        T=T+i
    return(T)

# faire l'appel triangle(10) en
console.
```

Fns... a A # Outils Exéc Script

```
ÉDITEUR : BUGS5
LIGNE DU SCRIPT 0006
def test():
    if 3*0.1==0.3:
        return('vrai')
    else:
        return('faux')
```

```
PYTHON SHELL
>>> test()
'faux'
>>> |
```

Fns... a A # Outils Exéc Script

```
ÉDITEUR : BUGS6C0
LIGNE DU SCRIPT 0003
def f(x):
    return x**2-7*x+5

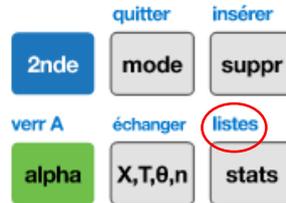
def dichotomie(a,b,precision):
    while (b-a)>precision:
        c=(a+b)/2
        if f(a)*f(c)<=0:
            b=c
        else:
            a=c
    return a,b
```

Fns... a A # Outils Exéc Script

Oubli des deux points, indentation, appel de bibliothèque, majuscule/minuscule, virgule...

Lists

Shortcut



```
ÉDITEUR : BUGS2
List
1: [ ]
2: list(séquence)
3: len()
4: max()
5: min()
6: .append(x)
7: .remove(x)
8: .insert(indice, x)
9: sum()
0↓sorted()
Échap
```

Number of objects in a list

Display elements asked

Add an object to a list

Sum all object of a list

Arrange in ascending order

Lists

```
PYTHON SHELL
>>> l=[7,6,9,2]
>>> l[0]
7
>>> l[2]
9
>>> |

Numbering starts to 0
```

Fns... a A # |Outils|Éditer|Script

```
PYTHON SHELL
>>> l=[7,6,9,2]
>>> len(l)
4
>>> l.sort()
>>> l
[2, 6, 7, 9]
>>> l.append(5)
>>> l
[2, 6, 7, 9, 5]
>>> |

Length of a list
Arrange in ascending order
Add a value
```

Fns... a A # |Outils|Éditer|Script

```
PYTHON SHELL
>>> l=range(5,20,2)
>>> list(l)
[5, 7, 9, 11, 13, 15, 17, 19]
>>> sum(l)
96
>>> |

Write a list with the function
« range »
```

Fns... a A # |Outils|Éditer|Script

```
PYTHON SHELL
>>> l=[1,2,4,1,1,4,5]
>>> l.count(1)
3
>>> l.count(2)
1
>>> |

Count the number of occurrence
```

Fns... a A # |Outils|Éditer|Script

```
PYTHON SHELL
>>> l=[1,2,5,8,11]
>>> 3 in l
False
>>> 5 in l
True
>>> |

Check if an object is in a list
```

Fns... a A # |Outils|Éditer|Script

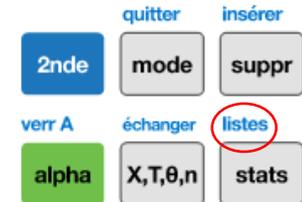
```
PYTHON SHELL
>>> x=[1,2]
>>> y=[4,5,6]
>>> x*y
[1, 2, 1, 2, 1, 2, 1, 2, 1, 2]
>>> [0]*5
[0, 0, 0, 0, 0]
>>> x+y
[1, 2, 4, 5, 6]
```

Fns... a A # |Outils|Éditer|Script

Lists

Write a function that return the list of the n first integer squared

Shortcut:



```
NORMAL FIX9 AUTO REAL RADIAN MP
EDIT MENU: [α][Phα][f5]
PROGRAM: CARRES
:ClrHome
:ClrList L1
:Input "Number of values "
,N
:For(I,1,N)
:I²→L1(I)
:End
:Disp L1
```

```
EDITOR: LISTE
PROGRAM LINE 0005
def carres(n):
    ++liste=[]
    ++for i in range(1,n+1):
    +++liste.append(i**2)
    ++return liste_
```

```
PYTHON SHELL
>>> carres(7)
[1, 4, 9, 16, 25, 36, 49]
>>> |
```

Application

Exercise 5 : the hare and the tortoise

One part of the hare and tortoise game goes like this : The distance to run is 6 squares. The die is thrown and if a six comes up the hare advances 6 squares, otherwise the tortoise goes forward one square.

- 1) Programme a simulation of this game using Python.
- 2) Write a piece of script which returns the number of wins of the hare and the tortoise.



```
NORMAL FLOTT AUTO RÉEL RAD MP
ÉDIT MENU: [a] [pha] [f5]
PROGRAM:ACT5
:Effécran
:Fixe 0
:0→N:0→D
:While D<6 et N<6
:nbrAléatEnt(1,6)→D
:N+1→N
:Disp D
:End
:If N=6
:Then
:Disp "The Turtle wins"
:Else
:Disp "The hare wins"
:End
:Fixe 9
```

```
ÉDITEUR : TORTUE
LIGNE DU SCRIPT 0002
from random import randint

def course():
    de=0
    case=0
    while de<6 and case<6:
        de=randint(1,6)
        case=case+1
        print (de)
    if case==6:
        return "la tortue a gagn
    else :
        return "le lièvre a gagn
```

```
PYTHON SHELL
>>> course()
4
3
3
2
5
4
'la tortue a gagné'
>>> |
Fns... a A # Outils Éditer Script
```

Application

Exercise 6 : Primeness test

Write an algorithm which tests for primeness and returns a boolean. Use the instruction `assert(n>=2)` (found in the instruction catalogue) to verify the hypothesis made in the argument.

```
NORMAL FIX9 AUTO REAL RADIAN MP
EDIT MENU: [a] [pha.] [f5]

PROGRAM:PRIMALIT
:Input "Number ",N
:For(I,2,N/2)
:If not(remainder(N,I))
:Then
:Disp "False"
:Stop
:End
:End
:Disp "True"
```

```
ÉDITEUR : PREMIER
LIGNE DU SCRIPT 0006

def premier(n):
    assert n>=2
    for i in range(2,n):
        if n%i==0:
            return False
    return True_

Fns... | a A # | Outils | Exéc | Script
```

```
PYTHON SHELL

>>> premier(9)
False
>>> premier(19)
True
>>> |

Fns... | a A # | Outils | Éditer | Script
```

T3 France contents

Les ressources pour travailler avec calculatrice



TI Codes

Découvrez le langage de programmation Python pas à pas avec TI Codes

[En savoir plus »](#)



Activités pour la classe en PDF

Téléchargez gratuitement des exemples d'activités prêtes à utiliser pour le lycée

BIENTÔT DISPONIBLE



CAHIER D'ACTIVÉS PYTHON AVEC LA TI-83 PREMIUM CE PAR EYROLLES

[CONSULTER DES EXTRAITS](#)



Tutoriels vidéos

Familiarisez-vous avec Python sur la TI-83 Premium CE avec nos tutoriels vidéos

[En savoir plus »](#)



Jean-Louis BALAS
Maths-Sciences teacher
French T³
jl.balas@orange.fr



Abir MARINA
Maths teacher
French T³
abir.marina@ac-nancy-metz.fr