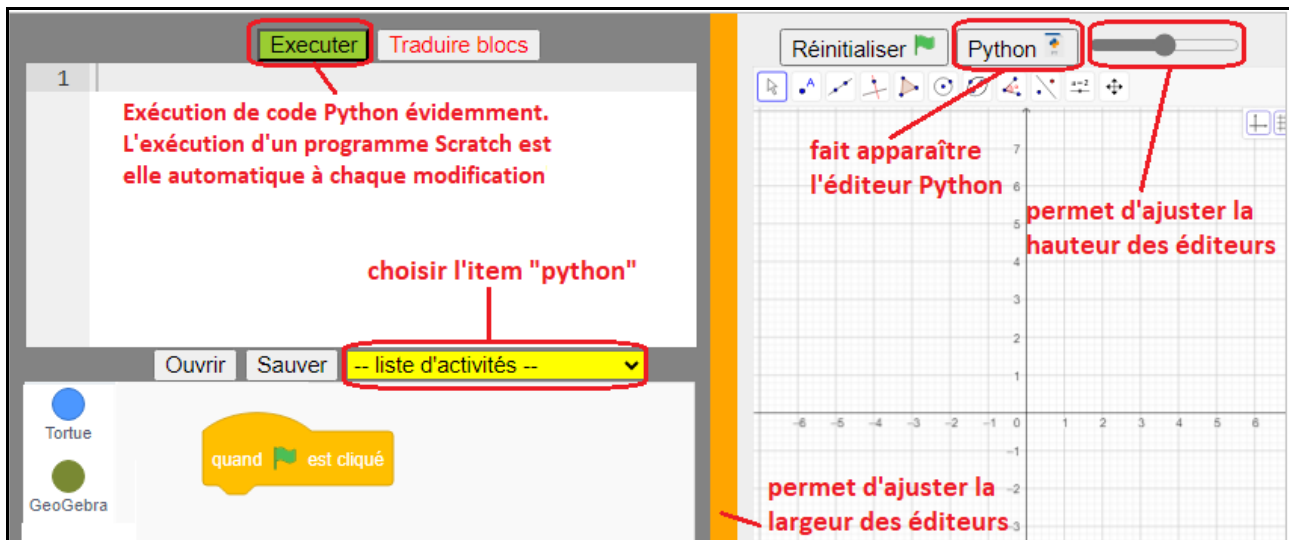


ScratchGGB, un logiciel pour enseigner Python au lycée avec GeoGebra

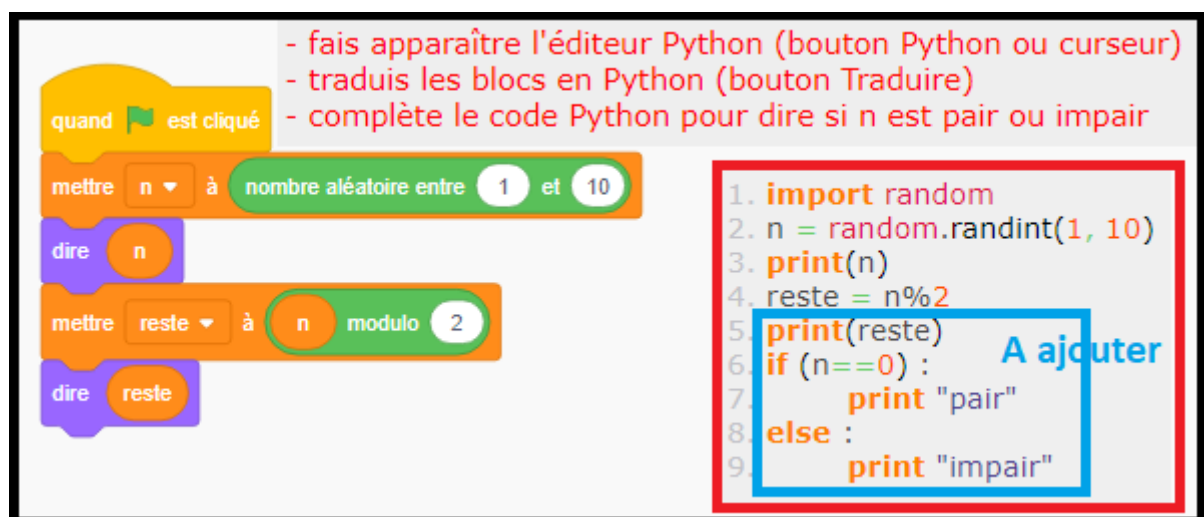
Patrick Raffinat (<https://sites.google.com/view/raffinatp/scratchggb/lycee>)

A) Introduction

En 2023, j'ai intégré **Python** dans **ScratchGGB** (voir <http://revue.sesamath.net/spip.php?article1606>). L'éditeur Python n'apparaît que si on clique sur le bouton « Python » ou sur le curseur à sa droite : c'est un choix pédagogique permettant de ne pas complexifier inutilement l'ergonomie du logiciel pour un usage au collège.



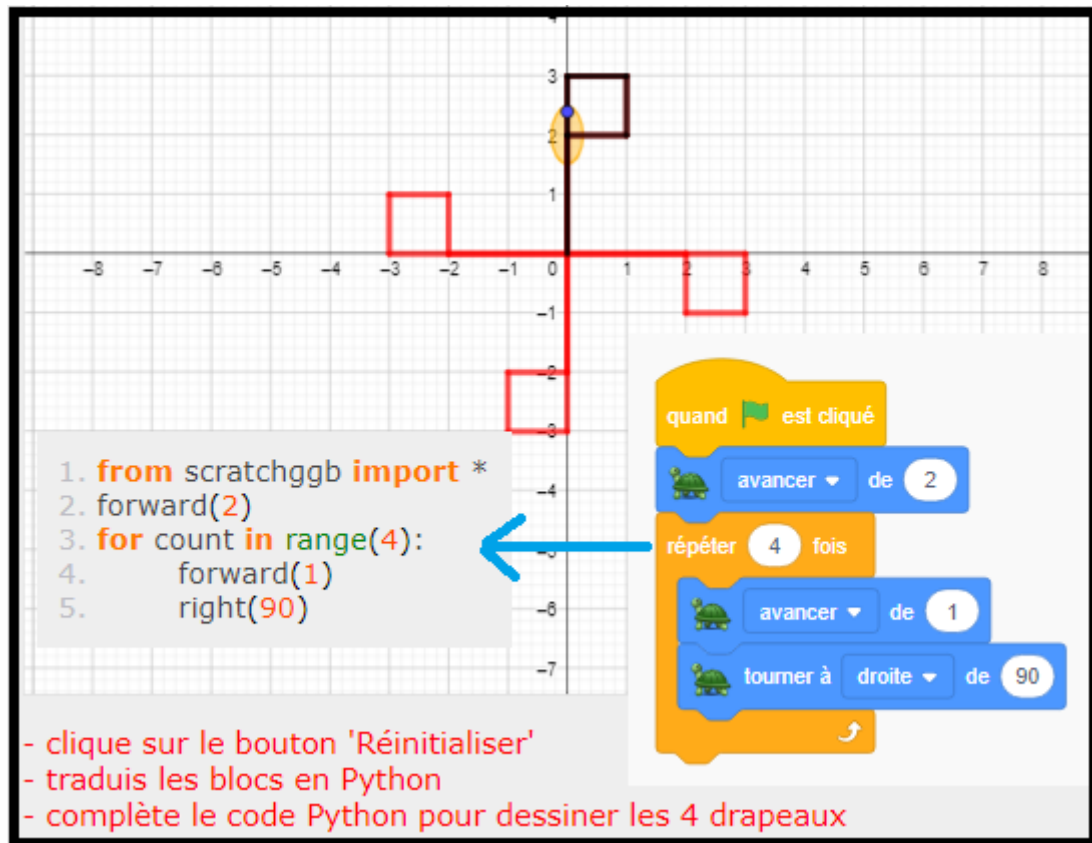
Voici un premier exemple (« test de parité »), pas fondamental évidemment, qui a pour but de faire découvrir le fonctionnement du logiciel aux élèves :



Il est accessible depuis la banque d'exercices de ScratchGGB. La traduction du programme par blocs (bouton **Traduire**) permet d'obtenir la syntaxe d'instructions Python pas forcément connues : « random.randint » et, à un degré moindre, « % ». Les 5 lignes de code ainsi créées sont ensuite à compléter par une instruction conditionnelle pour effectuer le test de parité demandé.

B) Exemple : tortues et drapeaux

Beaucoup de logiciels permettent de programmer avec une tortue. Le petit plus de ScratchGGB, grâce à Geogebra, est que l'on peut facilement vérifier le résultat à l'exécution : il sera correct si la trace de la tortue se superpose à la figure à reproduire (ici en rouge), et incorrect dans le cas contraire.



Les **commandes tortue** (forward, backward...) sont dans un **module nommé** « **scratchggb** » que j'ai créé. J'ai peut-être un peu trop aidé les élèves en leur donnant au départ un programme par blocs dessinant un drapeau, mais il leur reste quand même un travail algorithmique non négligeable à effectuer :

- soit **en imbriquant deux boucles**, sans oublier de faire revenir la tortue à sa position de départ après le dessin d'un drapeau
- soit **en introduisant un sous-programme dessinant un drapeau**, puis en l'utilisant dans une boucle

Le module « scratchggb » contient également des commandes géométriques (Point, Circle, Segment, Midpoint...) et des commandes tableur (setCell, getCell).

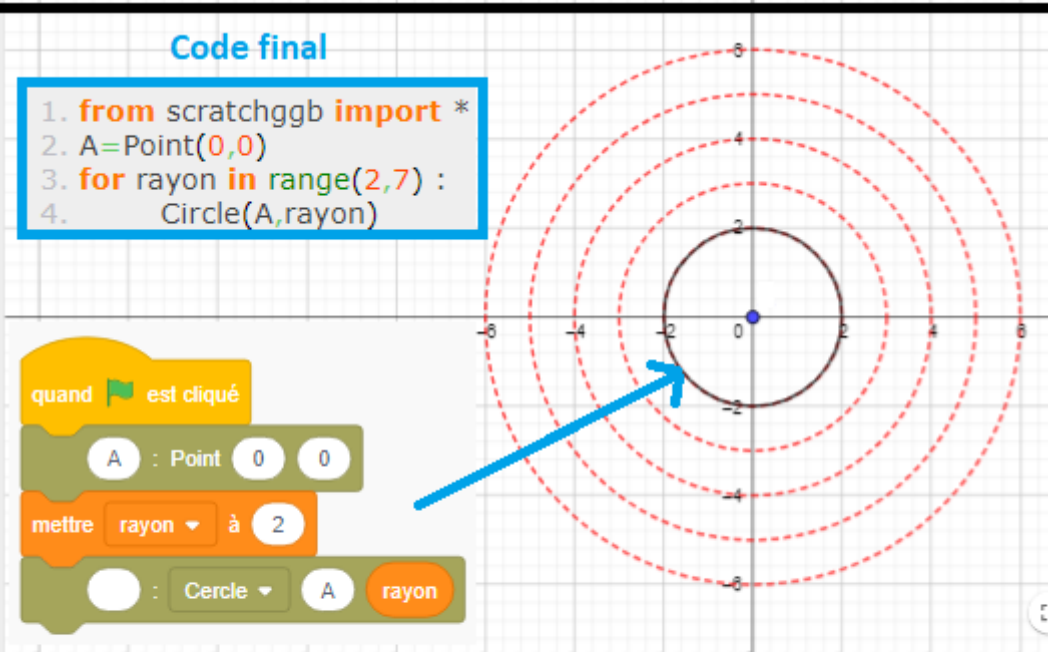
C) Exemple : cercles concentriques

Le programme par blocs dessine le centre, ainsi que le plus petit des 5 cercles à reproduire. Après en avoir obtenu la traduction en Python, il est facile d'en déduire le code final.

Code final

```

1. from scratchggb import *
2. A=Point(0,0)
3. for rayon in range(2,7) :
4.     Circle(A,rayon)
        
```

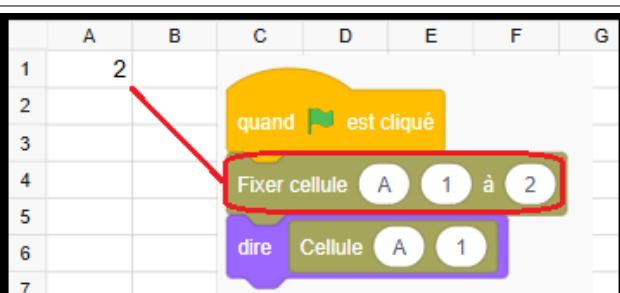


The Scratch script on the left shows a 'when green flag is clicked' event, followed by creating a point 'A' at (0,0), setting the radius to 2, and creating a circle with center 'A' and radius 'rayon'. The Geogebra coordinate plane on the right shows five concentric circles centered at the origin (0,0). The innermost circle is solid black with a radius of 2. The other four circles are dashed red with radii of 3, 4, 5, and 6. A blue arrow points from the 'rayon' block in the script to the innermost circle.

- clique sur le bouton 'Réinitialiser'
- traduis les blocs en Python
- complète le code Python pour dessiner les 5 cercles

D) Exemple : tableau

La traduction des blocs fournis au démarrage de l'activité donne les deux commandes Python à connaître pour communiquer avec le tableur de Geogebra : setCell (qui met une valeur dans une cellule) et getCell (qui récupère la valeur d'une cellule).



The Scratch script shows a 'when green flag is clicked' event, followed by 'fix cell A1 to 2' and 'say Cell A1 for 1 second'. The Geogebra spreadsheet shows column A containing even numbers from 2 to 20. A red arrow points from the 'fix cell A1 to 2' block to cell A1.

- traduis les blocs en Python
- mets en colonne A les nombres pairs de 2 à 20

```

1. from scratchggb import *
2. for k in range(1,11) :
3.     setCell("A",k,2*k)
        
```

Nombres pairs entre 2 et 20

```

from scratchggb import *
n = 4
setCell("A",1,1)
for lig in range(2,n+1) :
    setCell(1,lig,1)
    for col in range(2,lig) :
        val1 = float(getCell(col-1,lig-1))
        val2 = float(getCell(col,lig-1))
        setCell(col,lig,val1+val2)
    setCell(lig,lig,1)
        
```

	A	B	C	D
1	1			
2	1	1		
3	1	2	1	
4	1	3	3	1

Triangle de Pascal

E) Banque d'exercices Scratch : souvent adaptables à Python

La plupart des questions sont orientées **collège** : on y demande effet de compléter un programme par blocs initial (voir ci-dessous), énoncé qu'on remplacera au **lycée** par « traduire en Python le programme Scratch initial (via le traducteur de ScratchGGB), puis le compléter »

The image shows two Scratch scripts. The first script, titled "Maximum de 3 nombres", starts with a "when green flag is clicked" block, followed by three "set" blocks for variables a, b, and c with values 7, 5, and 9 respectively. It then uses an "if-then-else" block to compare a and b, setting a temporary 'maximum' variable to the larger one. Finally, it compares this 'maximum' with c and updates it if c is larger. The second script, titled "Dessin d'un drapeau", starts with a "when green flag is clicked" block, followed by a "define flag" block and an "advance" block for a turtle. Below the scripts is a coordinate grid showing a red flag shape drawn by the turtle.

Maximum de 3 nombres

Dessin d'un drapeau

A noter qu'on peut aussi programmer en Python avec la tortue quand il y a un **curseur Geogebra** :

The image shows a Python script for drawing a flag using the ScratchGGB library. The script defines a function 'drapeau()' that moves the turtle forward 2 units, then enters a loop that moves forward 1 unit, turns right 90 degrees, and moves backward 2 units. The number of times this loop is executed is determined by a slider 'n' in the Geogebra interface, which is currently set to 5. The script also includes a loop to repeat the 'drapeau()' function 'n' times, turning the turtle left by 360/n degrees between each iteration.

```
1 from scratchggb import *
2 def drapeau() :
3     forward(2)
4     for count in range(4):
5         forward(1)
6         right(90)
7     backward(2)
8 n = int(valeur('n'))
9 for count2 in range(int(n)):
10     drapeau()
11     left((360/n))
12
```