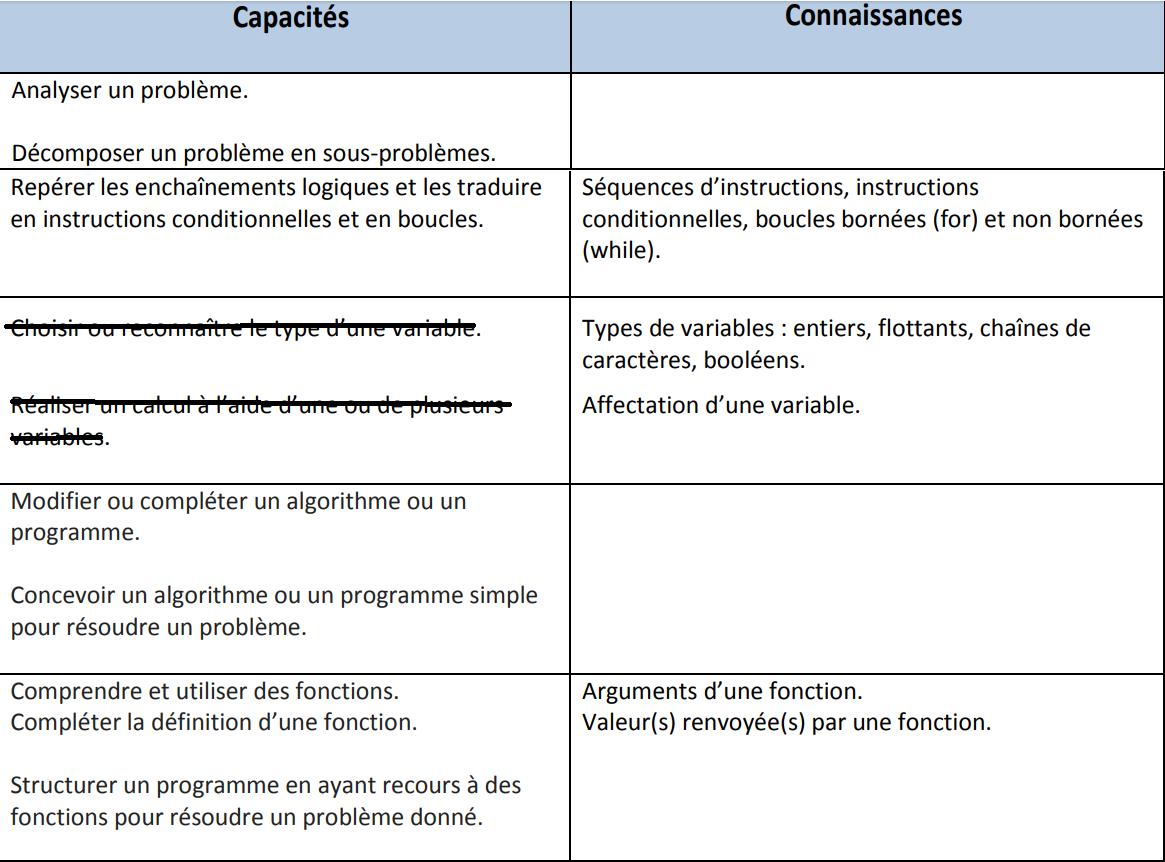
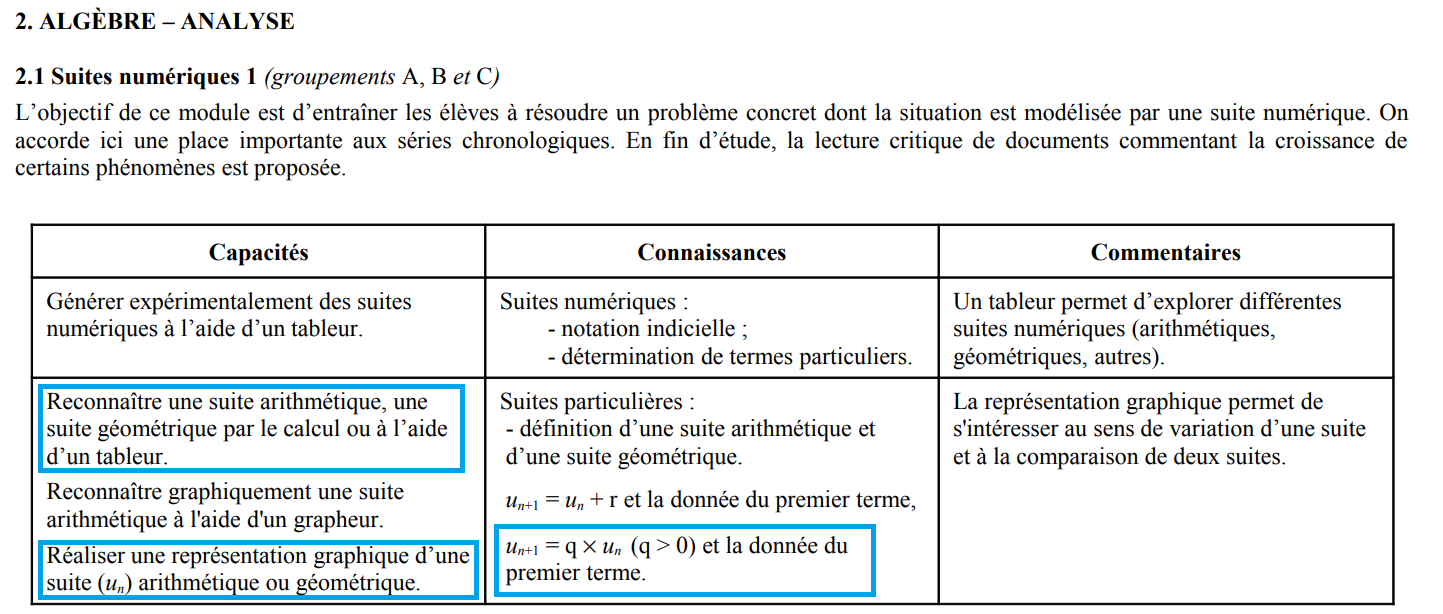
Un problème entre algorithmique et suites géométriques autour du triangle de Sierpinski

**I – Intégration dans le nouveau référentiel :**

Cette séance est prévue pour consolider la notion de suite géométrique en parallèle avec le travail d’algorithmes, en classe de première. Elle s’insère donc dans ces 2 parties du programme :



(Cet extrait de programme est celui de seconde, ceux de première n’étant pas encore paru, mais gageons qu’ils seront similaires d’un point de vue algorithmique, et que son travail ne sera pas cantonné qu’aux seules classes de secondes).



Bien qu’assez guidée, l’activité n’est pas si évidente. Elle mélange deux concepts (suites et algorithmes), et le cheminement n’est pas des plus simples. La situation n’est pas ancrée dans le monde professionnel, et l’enjeu de l’étude est autant dans le « décomposer un problème en sous-problèmes » que dans la reconnaissance de suites géométriques. Pour toutes ces raisons, l’activité n’est pas conçue en tant qu’activité introductrice. Elle me parait plutôt être un bon problème de fin de séquence, afin de rendre sa difficulté accessible.

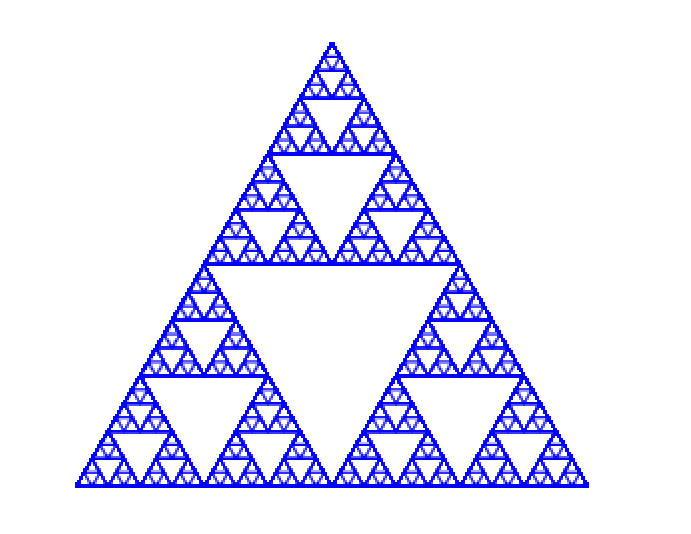
**II – Prérequis :**

Algorithmes (Scratch et Python) :

* Boucles « Tant que »
* Fonctions à paramètres
* Stylo / Turtle

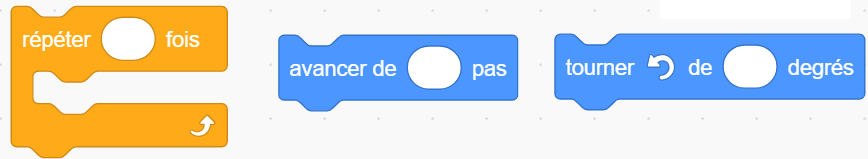
Suites : Reconnaître une suite géométrique, Générer une suite géométrique sur tableur

**III – Activité :**



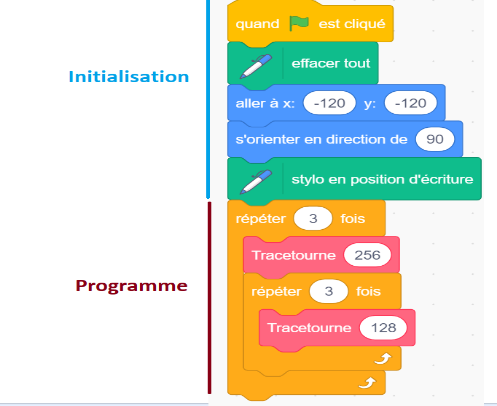
Combien y a-t-il de triangles dessinés ?

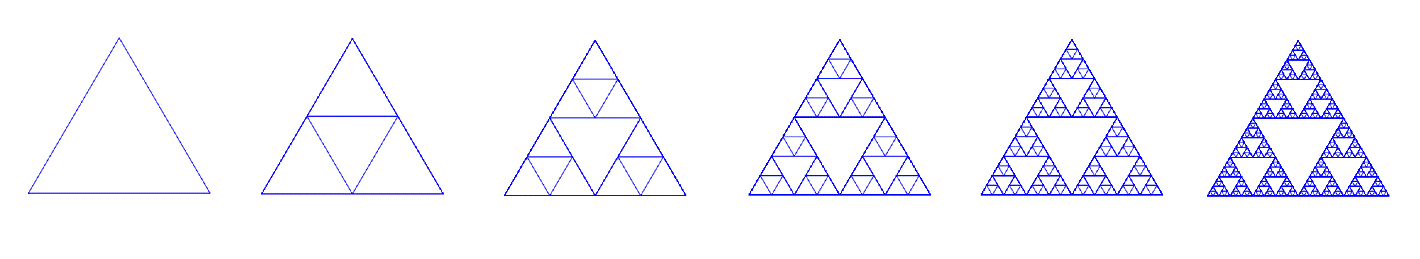
**Question 1 :** A partir des trois instructions suivantes, construire un triangle équilatéral de 256 de côté.



**Question 2 :** Créer une fonction « Tracetourne(n) » permettant d’avancer d’un nombre n de pas, puis de tourner de 120 degrés vers la droite. Construire le même triangle qu’à la question 1, mais en utilisant cette fonction.

**Question 3 :**

Voici comment tracer le second dessin. Analyse l’imbrication des boucles et trace les dessins suivants :



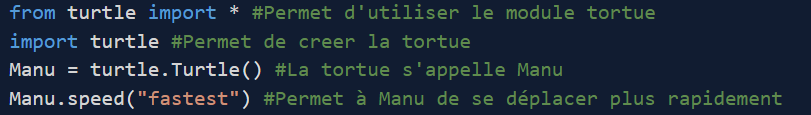
**Question 4 :** A chaque étape, combien de nouveaux triangles sont ajoutés ? Quel type de suite peut-on reconnaitre ?

**Question 5 :** Combien y a-t-il de triangles dans le dessin initial ? (S’aider du tableur)

**Questions supplémentaires :**

**Question 6 :** Combien de boucle faut-il imbriquer pour afficher 1 million de triangles ?

**Question 7 :** Implémenter le programme en Python On donnera l’initialisation suivante :



**Question 8 :** Utilisation d’Excel

En colonne B, calculer le nombre de triangles colorés rajoutés à l'étape n.

En colonne C, calculer le périmètre d'un triangle rajouté à l'étape n.

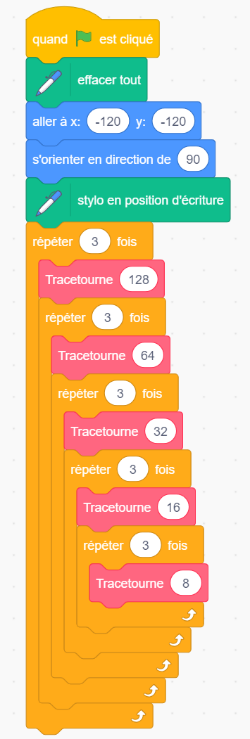
En colonne D, calculer le périmètre de la surface colorée rajoutée à l'étape n.

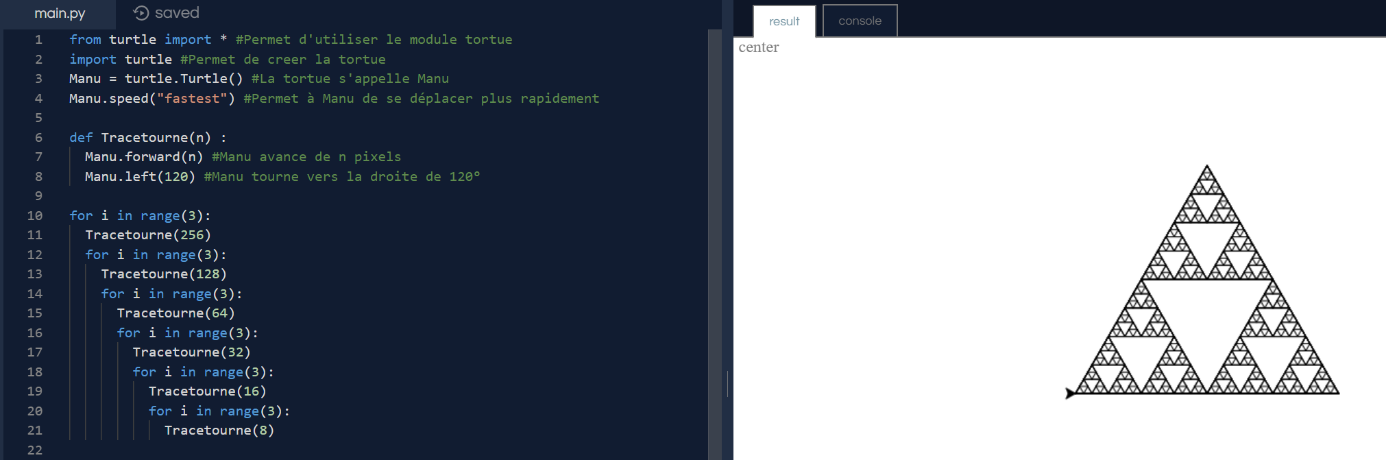
En colonne E, calculer le périmètre de la surface colorée à l'étape n.

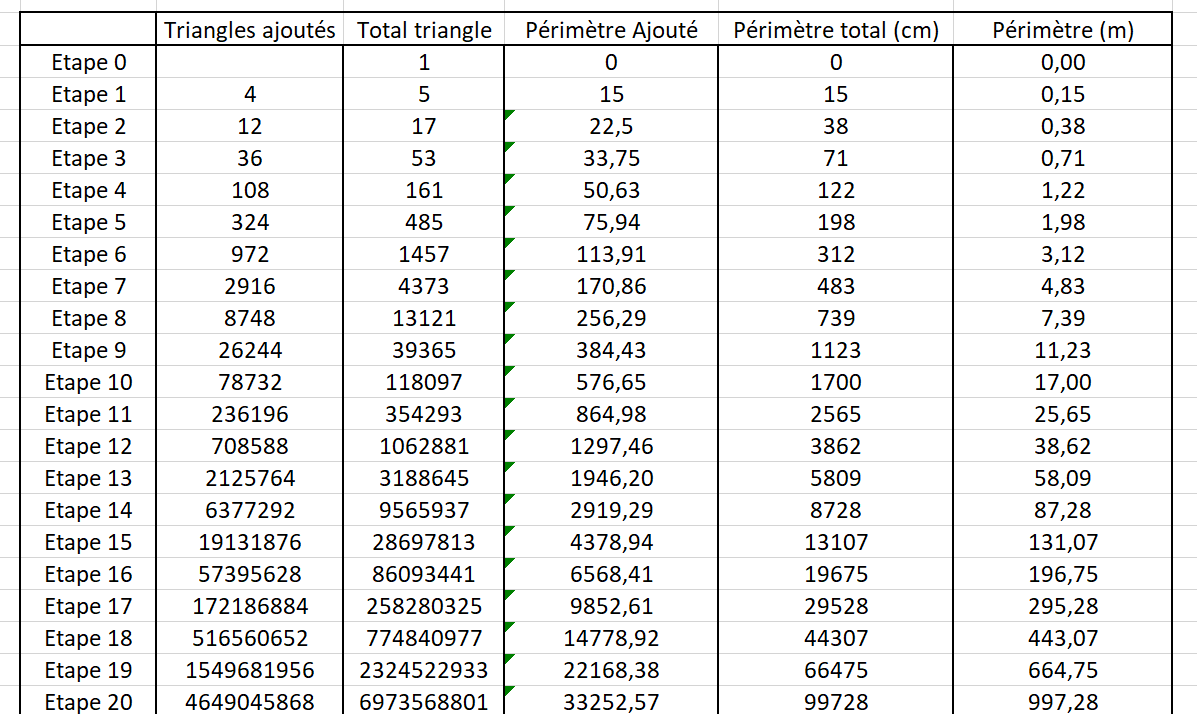
A partir de combien d’étape Manu devra parcourir plus de 100 km pour tracer la figure ?

|  |
| --- |
| ***Indics à distribuer en cas de blocage :***  Aide n°1 : Fonctions de base Turtle |
| Aide n°2 : Fonction Tracetourne |
| Aide n°3 : Code pour tracer le dessin n°2 |

***Corrigé succinct :***



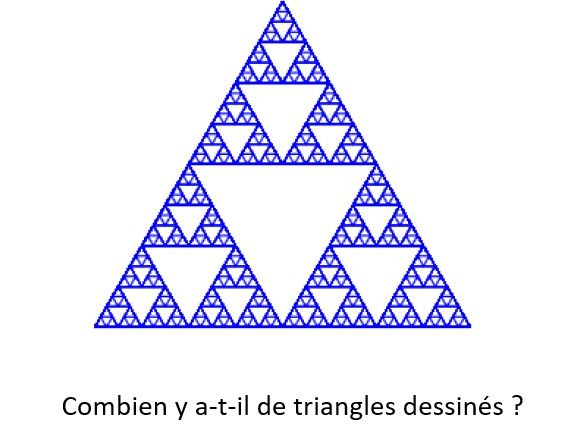




On peut voir apparaitre 2 suites géométriques : La suite Tn et la suite Pn. La suite Tn représente le nombre de triangles ajoutés à chaque étape. C’est une suite de premier terme T1 = 4, et de raison 3. La deuxième représente le périmètre ajouté. Elle à pour premier terme P1 = 15 et pour raison 1,5.

**IV – Scénario :**

Les élèves allument les ordinateurs et se connectent. On dispose donc d’environ 5 mn le temps que les ordinateurs fassent leur long travail de connexion au bureau. On profite donc de ce temps mort pour projeter la diapositive suivante.



Les élèves disposent de 5 minutes de recherche, temps durant lequel ils doivent tenter de répondre à la question donnée. Le but principal de l’image est de les engager cognitivement, et on ne s’attends en aucun cas à ce qu’ils trouvent la réponse. Plusieurs techniques de comptage peuvent émerger telles que :

* La Bruteforce : consiste en compter tous les triangles sur le dessin. La technique est rendue très complexe, les élèves n’ayant pas le dessin sur feuille (car il est simplement projeté).
* Le tri en semblables : technique de Bruteforce organisée, elle consiste en compter les triangles des plus grands aux plus petits. J’ai ainsi un grand triangle, puis 4 plus petits, puis 4\*4 plus petits, puis 4\*4\*4 plus petits et ainsi de suite jusqu’à arriver aux plus petits triangles. C’est celle qui s’avèrerait la plus efficace, car elle permet de dégager facilement la suite.
* L’isolation : Consiste en isoler le dessin en 3 triangles distincts, puis à réitérer l’opération jusqu’à isoler le triangle initial. On compte ainsi 3 fois moins de triangle. Cela dit, il faut penser à compter les triangles qui s’ajoutent lorsque l’on rejoint les 3 triangles isolés. Un temps de recherche assez long et de nombreux risques d’erreurs découlent de cette technique.

Après ce temps de recherche, on distribue l’activité papier aux élèves, composée du dessin, de la question principale et des 5 premières questions. Les questions supplémentaires sont au verso de la feuille. Les ordinateurs sont prêts, les élèves peuvent travailler en autonomie, et on peut passer dans les rangs, afin de débloquer les différents élèves. Scratch a pour but d’apprivoiser le problème. Il permet d’avancer rapidement. Il est dans cet exercice, l’équivalent du pseudocode, que l’on peut tester en direct.

Le but de l’activité est que tout les élèves parviennent à résoudre la question 5 (qui est la question initiale). Sachant qu’ils ne prendront pas tous le même temps, des différenciations sont prévues.

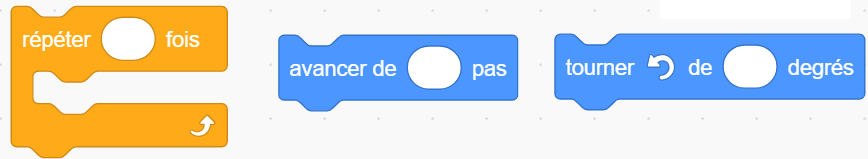
Ainsi, dans sa version la plus simple, l’élève réalise sur Scratch le triangle de Sierpinski, puis reconnait la suite géométrique, calcule son rang (grâce aux tracés sur Scratch par une méthode de type essai-erreurs), puis calcule le nombre de triangles (sur Excel, ou à la calculatrice). Il s’agit là d’une difficulté minimale, qui a pour objectif d’être validée par l’intégralité de la classe. Dans sa version intermédiaire, l’élève traduit le programme Scratch en Python, et répond à la question n°6. Cette difficulté intermédiaire à pour objectif d’être validée par un maximum d’élèves de la classe. Dans sa version la plus complexe, l’élève travaillera aussi sur une suite géométrique supplémentaire générée par le chemin parcouru par notre tortue. On supposera qu’elle ne retrace jamais deux fois le même trait (ce qui n’est pas vrai) afin de simplifier les calculs.

Un ultime niveau de difficulté peut être pensé, pour les utilisateurs de python les plus aguerris, mettant en jeu la programmation récursive du triangle de Sierpinski. Ce type permet d’aboutir à une fonction de type **Sierpinski (longueur)** qui trace la nième itération du fameux triangle de côté longueur.

Bien que les élèves soient censés travailler sur Python, le passage sur Scratch dans ce problème particulier me semble pertinent. En effet, la question n°3 est une question mettant en place un raisonnement inductif, et un logiciel de programmation visuelle me semble faciliter cet obstacle.

**V – Organisation mathématique**

***Question 1 :*** *A partir des trois instructions suivantes, construire un triangle équilatéral de 256 de côté.*



T – Tracer un triangle à l’aide des TICE

τ : On répète trois fois : Tracer un segment de longueur l, Tourner de 60 °

θ : Un triangle équilatéral à 3 angles de 60° et 3 côtés de même longueur

La technique est « imposée » par les blocs. Même si l’on peut tourner dans les deux sens, l’utilisation de la boucle est obligatoire. L’utilisation de cette dernière est fondamentale si l’on veut pouvoir traiter la suite du problème d’un point de vue algorithmique. En effet, la construction la plus simple du triangle de Sierpinsky impose plusieurs boucles imbriquées. S’il n’y a pas de boucle dès le départ, cette construction n’est plus possible.

***Question 2 :*** *Créer une fonction « Tracetourne(n) » permettant d’avancer d’un nombre n de pas, puis de tourner de 120 degrés vers la droite. Construire le même triangle qu’à la question 1, mais en utilisant cette fonction.*

Cette question est facultative dans l’absolu, mais permet de simplifier considérablement le programme par la suite. Les boucles seront plus visuelles, et la totalité du programme paraitra beaucoup moins chargé, ce qui évitera je l’espère de désengager les élèves d’un programme qui parait trop compliqué.

T – Utiliser et concevoir des fonctions à arguments

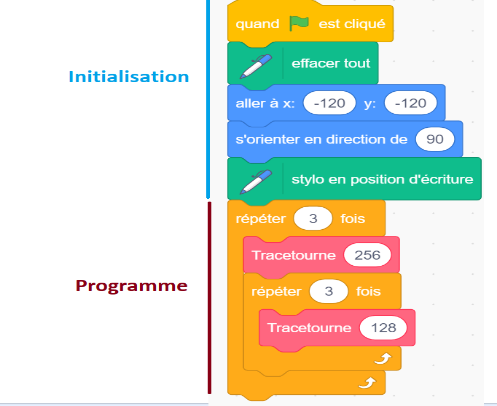
τ : def fonction (arg1)

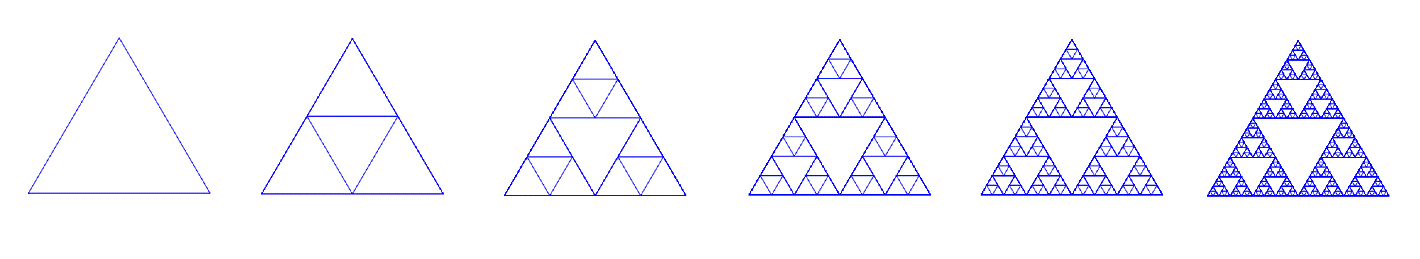
Instruction 1

Instruction 2

θ : Langage Python

**Question 3 :**

*Voici comment tracer le second dessin. Analyse l’imbrication des boucles et trace les dessins suivants :*



C’est la question cruciale de l’exercice. On pourra pourquoi pas prévoir une aide supplémentaire consistante à donner à l’élève bloqué l’algorithme permettant de tracer la 3eme figure.

T – Décomposer un problème en sous problèmes. Modifier ou compléter un algorithme donné.

τ  – Repérer l’équivalence entre nombre d’imbrication de boucles et

θ **–** Language Python

***Question 4 :*** *A chaque étape, combien de nouveaux triangles sont ajoutés ? Quel type de suite peut-on reconnaitre ?*

T – Reconnaitre une suite géométrique

τ  : Un+1 / Un = r

θ : Définition suite géométrique

***Question 5 :*** *Combien y a-t-il de triangle dans le dessin initial ?*

Les élèves n’ont pas en classe de première la formule de la somme, ni ne peuvent calculer une suite géométrique à un rang donné. Excel va donc s’imposer.

T – Calculer la somme d’une suite de termes géométriques

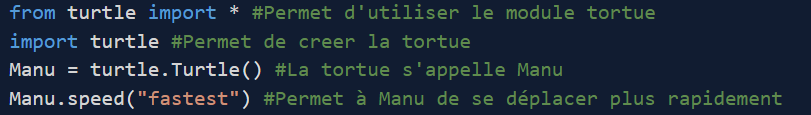
τ  – Formule Somme()

θ : TICE

***Question 6 :*** *Combien de boucle faut-il pour afficher 1 million de triangles ?*

Même Organisation mathématique que la question précédente.

***Question 7 :*** *Implémenter le programme en Python On donnera l’initialisation suivante :*



Traduire le programme généré sur Scratch me parait être un bon exercice. Il permettra de se réapproprier les notions de boucles for et de fonctions à paramètres, tout en revoyant le module Turtle. Il permettra aussi de voir qu’un même raisonnement peut se traduire dans différents langages équivalents. L’intérêt principal de disposer du programme en Python et qu’il permet de s’habituer à un langage de programmation permettant très largement de dépasser, dans un temps futur, les possibilités offertes par Scratch.

***Question 8 :*** *Utilisation d’Excel*

*En colonne B, calculer le nombre de triangles rajoutés à l'étape n.*

*En colonne C, calculer le périmètre rajouté à l'étape n.*

*En colonne D, calculer le périmètre de la figure à l'étape n en cm.*

*En colonne E, calculer le périmètre de la figure à l'étape n en cm*

*A partir de combien d’étape Manu devra parcourir plus de 100 km pour tracer la figure ?*

T – Calculer la somme d’une suite de termes géométriques

τ – Formule Somme()

θ : TICE

T – Convertir des longueurs de cm à m

τ: lkm = lcm/100

θ – Système métrique

**VI – Matériel**

* Ordinateurs connectés à Internet (pour accéder à l’environnement Scratch et Python)
* Tableur

**VII – Conclusion**

Lors de cette activité, les élèves auront pu traiter différents pans du programme de première. Ils auront pu notamment générer des suites géométriques, et les représenter à l’aide d’un tableur, afin de modéliser leur évolution en fonction du rang n. D’un point de vue algorithmique, ils auront décomposé un problème complexe en différents sous programmes qui s’imbriquent les uns les autres, tout en réutilisant les notions de boucles for et de fonctions à argument. L’activité propose plusieurs niveaux différents de complexité permettant plusieurs différenciations possibles en fonction des difficultés de chacun. Ce problème permet enfin de mobiliser 4 des 5 compétences. La décomposition du triangle de Sierpinsky en de nombreux triangles équilatéraux permet de s’approprier le problème. On demande de réaliser une portion de code, puis de raisonner, afin de le modifier pour arriver à un but précis. Le but étant connu (image de départ), l’élève à un modèle à comparer afin de valider qu’il arrive bien au résultat demandé. Il faudra prévoir de travailler les compétences de communication dans d’autres activités.

On peut enfin conclure l’activité par un exemple d’application concrète de ce triangle en lien avec les murs anti-sons, reprenant ce principe de motifs autosimilaires afin d’agrandir la surface d’absorption du son, en lien avec le module « Se protéger du son » fait en classe de première en physique. On peut pourquoi pas imaginer, à l’aide des traditionnelles mallettes permettant de mesurer l’isolation phonique d’un materiau, inclure une forme de type fractale (avec un niveau d’itération raisonnable) en polystyrène, et la comparer avec un morceau plan de même type, comme réalisé dans le dernier sujet des olympiades de physiques.



**VIII - Références :**

La question 3 de l’activité présentée est une activité du livre suivant :

Arnaud Bodin, *Python au lycée : Algorithmes et programmation,* ed Exo 7 p.14-15

Le lien avec les murs de forme fractale est inspiré d’un sujet proposé par le lycée Germaine Tillon – Saint Bel pour les olympiades de physique :

<http://grenoble.udppc.asso.fr/IMG/pdf/gr30_les_fractales_contre_le_bruit.pdf>