Fractales avec Geogebra

Au cours de cette séance, vous allez apprendre à représenter de façon approchée des fractales avec le logiciel Geogebra. Les figures obtenues pourront êre insérées comme illustrations dans les diapositives du diaporama rendu comme document projet. Les protocoles de construction devront être joints comme Annexes du projet.

1 Triangle de Sierpinski

Le mathématicien polonais Sierpinski a proposé en 1905, une figure géométrique obtenue par itérations successives d'un même procédé algorithmique.

- on part d'un triangle plein (pour le triangle de Sierpinski canonique, on part d'un triangle équilatéral)
- on relie les milieux des trois côtés ce qui partage le triangle initial en quatre triangles
- on évide le triangle central (en le coloriant d'une autre couleur)
- on réitère le procédé avec les trois petits triangles pleins obtenus

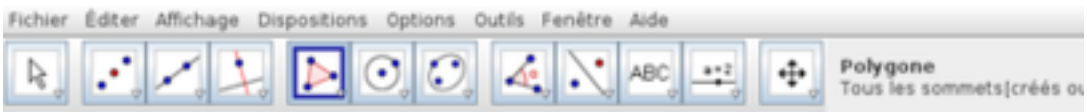
En itérant ce procédé *ad infinitum*, on obtient le triangle de Sierpinski. Pour le représenter de façon approchée, on fixe un nombre fini d'itérations.

itération 0

itération 1

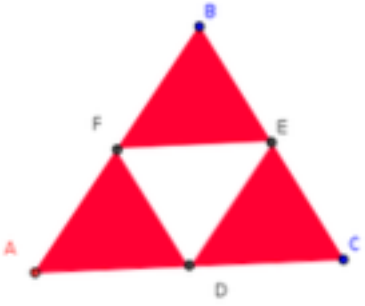
itération 2

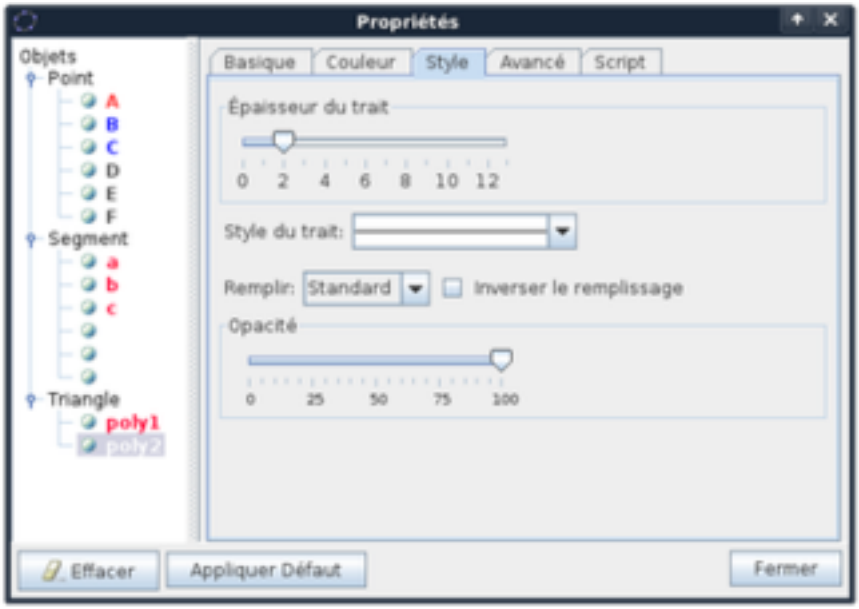
1. Ci-dessus on a représenté une approximation du triangle de Sierpinski après deux itérations. Compléter la dernière figure, pour obtenir une représentation approchée après trois itérations.
2. Lancer le logiciel Geogebra et ouvrir un nouveau fichier qu'on enregistrera dans son espace personnel sous le nom `sierpinski.ggb`. On va créer un outil sierpinski qui va nous permettre de créer un triangle de Sierpinski après une itération en cliquant sur 3 points.
   a. Créer d'abord trois points A,B et C avec l'outil Point de la barre d'outils puis le triangle ABC (nommé `poly1`) avec l'outil Polygone.



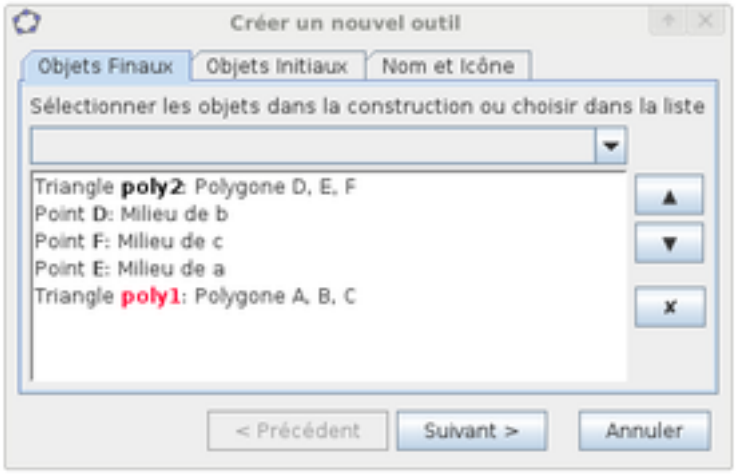
   b. Créer les milieux des trois côtés du triangle avec l'outil milieu puis le triangle (nommé `poly2`) reliant ces trois points avec l'outil Polygone.

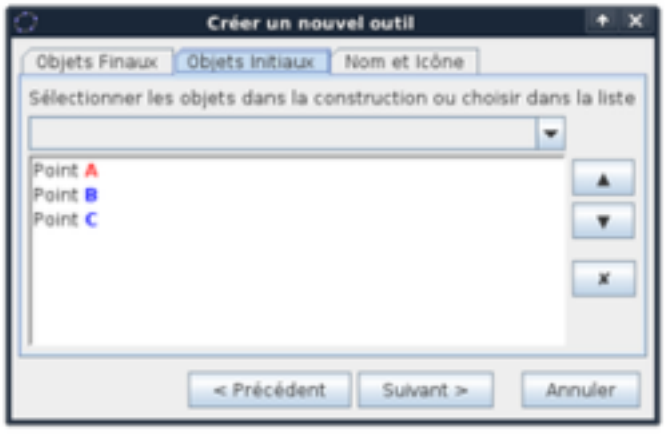
   c. Paramétrer en rouge la couleur de fonds de `poly1` et en blanc celle de `poly2` en réglant l'opacité sur 100.





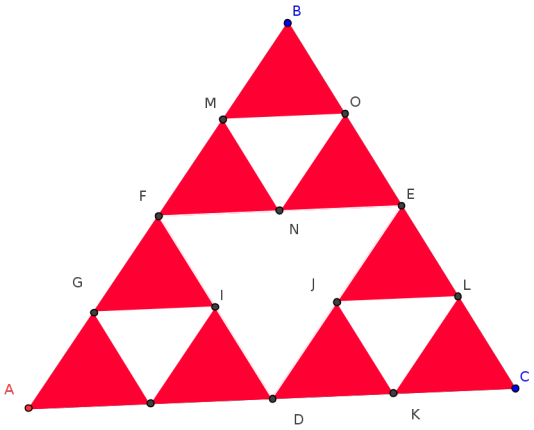
d. On peut maintenant créer notre outil `sierpinski`. Cliquer sur l'item `Créer un nouvel outil` du menu `Outils`, puis sélectionner les objets finaux (en plaçant au sommet de la pile les derniers objets à construire), les objets initiaux (les trois points A, B et C) et donner un nom à l'outil.





e. En sélectionnant notre nouvel outil dans la barre d'outils et en cliquant sur les points A,F et D on peut alors itérer le procédé pour le triangle AFD. En répétant l'opération pour les triangles FBE et DEC on obtient une approximation du triangle de Sierpinski obtenu à partir de ABC après deux itérations.

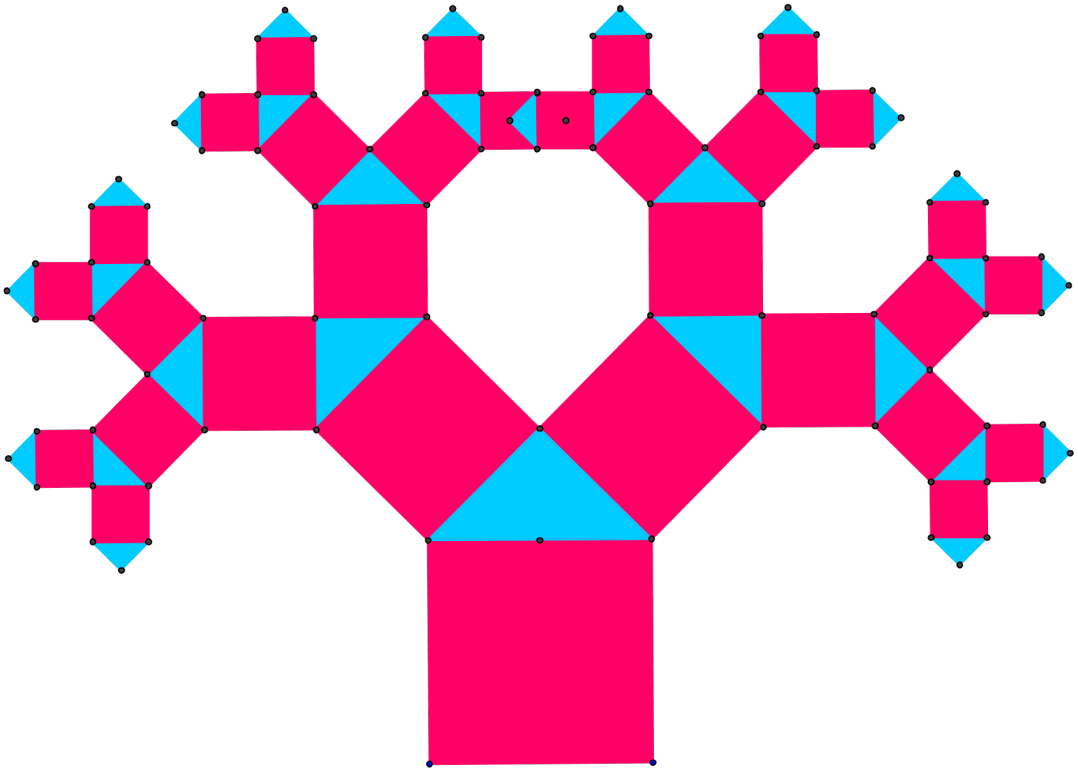
Construire ainsi une approximation du triangle de Sierpinski après 4 itérations.



2 Arbre de Pythagore

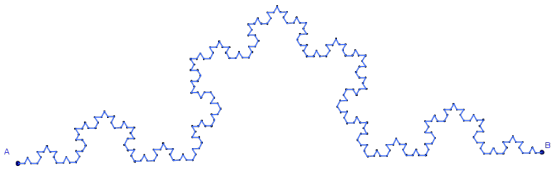
1. Faire une recherche documentaire sur l'arbre de Pythagore.
2. En vous inspirant du travail effectué pour le triangle de Sierpinski, créer avec Geogebra un arbre de Pythagore après au moins 5 itérations comme sur la figure ci-dessous.

   On utilisera les outils `Polygone régulier` , `Milieu`, `Perpendiculaire`, `Cercle (centre-point)`, et `Polygone`.



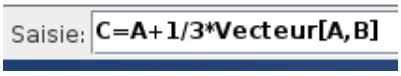
3 Flocon de Von Koch

On va représenter une approximation de la courbe de Von Koch d'extrémités deux points A et B, obtenue après 4 itérations.



1. On commence par créer un outil `koch` qui à partir de deux points A et B construit la liste des extrémités des segments de la courbe obtenue après une itération de l'algorithme de Von Koch.

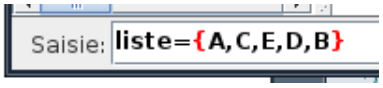
   a. On créé d'abord le point C tel que $\overrightarrow{AC} = \frac{1}{3}\overrightarrow{AB}$ avec cette commande dans la barre de saisie :



   b. Puis on crée de même le point D tel que $\overrightarrow{BD} = \frac{1}{3}\overrightarrow{BA}$.

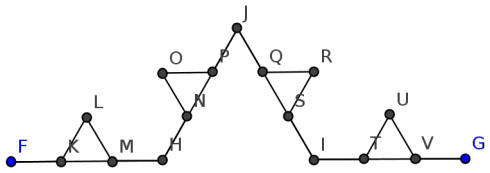
   c. Ensuite on crée le point E, sommet du triangle équilatéral de base [AB] qui est orienté vers le haut. On pourra utiliser l'outil `Polygone régulier`.

   d. Enfin on crée la liste des points A,C,E,D,B qui sera retournée par notre outil `koch`



   e. On crée alors notre nouvel outil `koch` en sélectionnant `liste=A,C,E,D,B` comme objet final et les points A et B comme objets initiaux.

   On peut alors utiliser cet outil pour réaliser des itérations supplémentaires de l'algorithme de Von Koch. On aurait pu inclure dans notre outil le tracé des segments reliant les points successifs de la courbe mais il aurait fallu effacer à chaque nouvelle itération certaines parties tracées lors de l'itération précédentes (les bases des triangles équilatéraux).



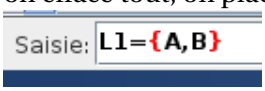
   f. Lors de chaque itération, par combien est multiplié le nombre de segments constituant la courbe approchée ?

   Au départ on a un segment, combien aura-t-ton de segments au bout de trois itérations ?

   Combien de fois faudra-t-il utiliser notre outil `koch` pour réaliser 4 itérations ? et pour 5 itérations ?

   g. Pour réduire un peu le nombre de manipulations on peut utiliser quelques commandes Geogebra comme `Séquence[ <Expression>, <Variable>, <de>, <à> ]` pour réaliser un boucle en retournant une liste de valeurs ,`Elément[ <Liste>, <Position> ]` pour accéder à l'élément d'une liste, `Longueur[ <Liste> ]` pour la longueur d'une liste, `Unir[ <Liste>, <Liste>, ....]` pour faire la réunion non ordonnée de plusieurs listes ou `Segment[ <Point>, <Point> ]` pour tracer un segment.

   - on efface tout, on place deux points A et B puis on crée dans la barre de saisie la liste de points :



- on crée ensuite la liste de points correspondant à l'itération 1 avec la commande composée :

Saisie: `L2=Unir[Séquence[koch[Elément[L1, k], Elément[L1, k + 1]], k, 1,Longueur[L1] - 1]]`

- puis la liste de points correspondant à l'itération 2 avec la commande composée :

Saisie: `L3=Unir[Séquence[koch[Elément[L2, k], Elément[L2, k + 1]], k, 1,Longueur[L2] - 1]]`

- puis la liste de points correspondant à l'itération 3 avec la commande composée :

Saisie: `L4=Unir[Séquence[koch[Elément[L3, k], Elément[L3, k + 1]], k, 1,Longueur[L3] - 1]]`

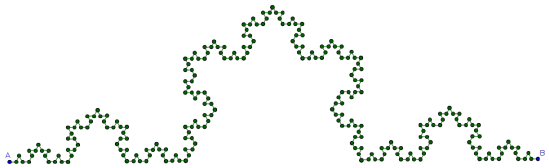
- puis la liste de points correspondant à l'itération 3 avec la commande composée :

Saisie: `L5=Unir[Séquence[koch[Elément[L4, k], Elément[L4, k + 1]], k, 1,Longueur[L4] - 1]]`

- enfin on trace tous les segments de la courbe de Von Koch approchée avec la commande composée :

Saisie: `Séquence[Segment[Elément[L5, k], Elément[L5, k + 1]], k, 1, Longueur[L5] - 1]`

On obtient alors le graphique suivant :



h. Créer alors un outil `koch4` qui retourne cette figure finale lorsqu'on clique sur deux points initiaux A et B puis utiliser cette commande pour dessiner un flocon de Von Koch comme ci-dessous :