Thème : géométrie

171 pas

TI-83 Premium CE

python™

Résumé : il s'agit d'un travail collaboratif dans lequel les élèves cherchent à trouver des emplacements possibles de coffres au trésor. Il est préférable d'avoir fait la fiche précédente sur la réciproque du théorème de Pythagore.

Mots-clés : bibliothèque `math` ; théorème de Pythagore

Compétences visées

Chercher : « Observer, s'engager dans une démarche, expérimenter en utilisant éventuellement des outils logiciels ».

Modéliser : « Utiliser, comprendre, élaborer une simulation numérique ».

Représenter : « passer d'un mode de représentation à un autre »

Calculer : « mettre en œuvre des algorithmes simples ».

Situation déclenchante

Mathurin le pirate a caché des trésors et il a laissé des instructions bien étranges pour les retrouver.

Dans un grand désert bien plat, il a caché des trésors selon une méthode bien précise :

« Depuis le point de départ en noir sur la carte ci-contre, il faut faire exactement 171 pas vers le nord, puis faire un nombre entier de pas vers l'est au moins égal à 171 pas. S'il est possible de revenir au point de départ par un nombre entier de pas, alors il faut creuser ! »

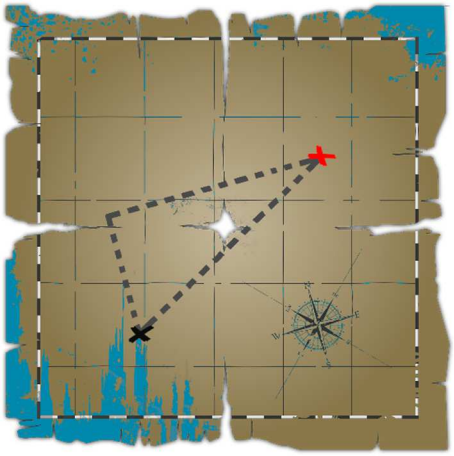


Image libre de droits d'après Pixabay

Problématique

Quels sont tous les emplacements des trésors ?

171 pas

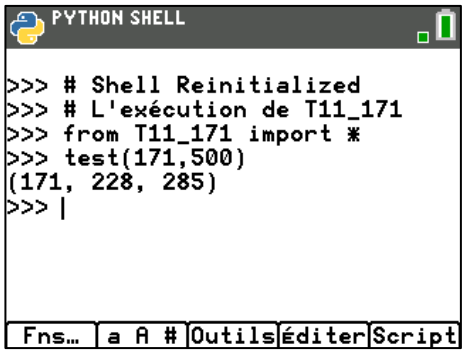
Scénario pédagogique

- **Avec la classe :** proposer le problème précédent et s'assurer que tous les groupes ont compris l'utilisation du théorème de Pythagore pour cette activité. Cela pourra être facilitée par l'utilisation préalable de la fiche précédente sur le théorème de Pythagore.
- **En groupe de 2 à 4 élèves :** les élèves doivent faire une ébauche d'algorithme permettant la résolution du problème proposé, avant de passer à l'implantation dans un langage ou un autre, après validation du professeur. Cette partie peut nécessiter un temps assez long pour que les élèves s'approprient le programme, notamment par la maîtrise de fonctions dédiées : boucle `while` par exemple.
- **Mise en commun :** les élèves expliquent leurs démarches.
- **Preuve :** les élèves peuvent montrer qu'il existe un nombre fini de solutions.
- **Pour les élèves les plus en avance :** il est possible de leur proposer un ou plusieurs prolongements possibles, décrit en fin de fiche.

Voici les visuels à l'issue des programmes :

en Scratch

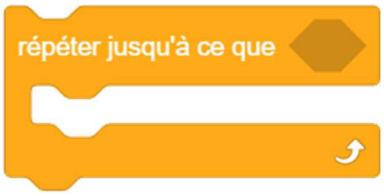
avec la TI-83 Premium CE Edition Python

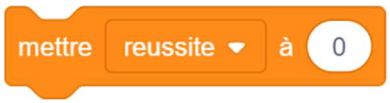
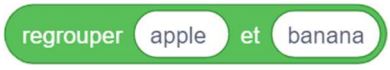


171 pas

Avec Scratch

Les briques de codes principales en Scratch pour ce programme	Explications Cette instruction permet de :	Traduction en langage Python sur la TI-83 Premium CE Edition Python
	Prendre la racine carrée du nombre en paramètre.	`sqrt()` C'est l'abréviation de ***square root***, signifiant « racine carrée » en anglais.

	Répéter les instructions contenues dans la boucle jusqu'à ce que la condition se réalise.
	
	Il n'y a pas de boucle « répéter tant que » dans Scratch, et pas de boucle « répéter jusqu'à ce que » dans le langage Python.
	
	Attention donc à l'adaptation en langage Python.
		`while condition inverse :`
		`◆◆instructions`
		
		L'indentation avec ◆◆ est nécessaire pour définir les instructions à exécuter d'une structure dans le code Python.

	Créer une condition dans laquelle il est testé si la partie entière (« plancher ») de la valeur de la variable c est égale à elle-même. Le mot « plancher » est une traduction de ***floor*** en anglais. C'est une condition souvent utilisée.	`int(c)==c` `int` est l'abréviation de ***integer***, signifiant « entier » en anglais. Ici, il s'agit de la fonction partie entière. A noter que pour une condition dans Python, il faut mettre un double égal, le simple égal signifiant une affectation.
	S'assurer de savoir s'il y a au moins une solution, de sorte qu'un message puisse être transmis.	La fonction `return` arrête le programme, par conséquent, il n'est pas utile d'avoir une variable supplémentaire.
	Regrouper du texte et des variables pour les afficher dans un message texte.	L'utilisation de la fonction `return` limite les interactions avec l'utilisateur. Voir la partie « Pour mieux lire le code Python ».

Une programmation possible est disponible sur le site de Scratch : scratch.mit.edu/studios/27615196/

171 pas

Avec Python

Le code complet est à construire par les élèves.

Ce code est composé d'une seule fonction :

- La fonction `test` de paramètres `min` et `max` qui sont respectivement le nombre de départ et le nombre maximal auquel finir.

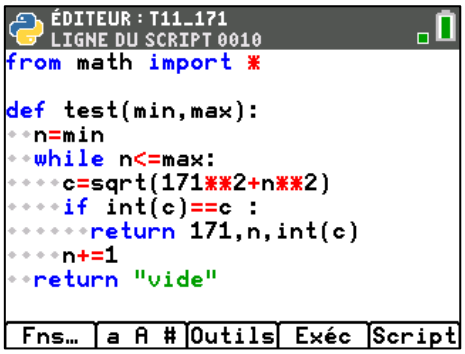
  Etant donné que le nombre de répétitions n'est pas connu, il faut utiliser une boucle `while`, qui nécessite une condition. Ici, la variable `n` doit rester inférieure ou égale au maximum pour que les instructions de la boucle se poursuivent.

  Il aurait été tout à fait possible d'utiliser une boucle `For`.

  Un test est réalisé pour chaque triplet potentiel pour savoir si la valeur de l'hypoténuse est bien un nombre entier.

  A noter l'utilisation de `int(c)` dans le `return` : voir la partie « Pour mieux lire le code Python ».

  Si aucun triplet n'a été trouvé dans l'étendue, alors le script renvoie le message `"vide"`.



Il faudra répéter plusieurs fois ce programme pour s'assurer d'obtenir toutes les valeurs d'emplacements des coffres à trésors, ce qui nécessite une coordination au sein des groupes et l'implémentation des programmes dans chaque calculatrice.

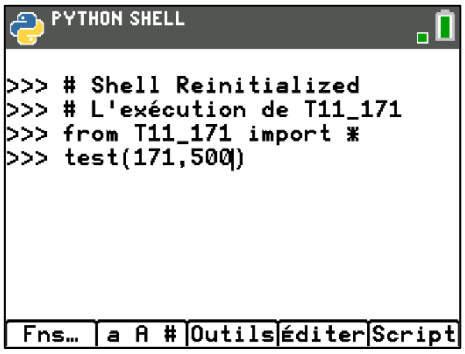
Une programmation possible est disponible sur le site TI : education.ti.com/fr/scratch-python

171 pas

Mode opératoire

Une fois le script exécuté, il faut appuyer sur la touche `[var]` : la fonction définie dans le script apparait.

Par les flèches directionnelles, il faut sélectionner la fonction `test`, valider par `Ok`, puis ajouter la valeur de départ du deuxième côté du triangle rectangle et la valeur finale à laquelle finir, en séparant les paramètres par une virgule.



Prolongements possibles

Voici des pistes pour les élèves les plus rapides ou qui ont envie de prolonger le travail :

- programmer un test pour s'assurer que l'utilisateur a bien mis un nombre supérieur à 171 ;
- mettre les solutions trouvées dans une liste pour pouvoir être utilisée ensuite ;
- tester avec un nombre plus important de pas au départ comme 1 771.

Pour mieux lire le code Python

Il faut faire attention avec les boucles « répéter tant que » et « répéter jusqu'à ce que ». En effet, sans une condition d'arrêt, ces boucles peuvent créer une erreur en s'exécutant de façon infinie.

Il faudra appuyer sur la touche `[on]` pour arrêter le programme.

Il faut importer ici la bibliothèque `math` afin d'avoir la fonction racine carrée : `sqrt()`.

Concernant les interactions avec l'utilisateur, il convient d'indiquer que Scratch est un langage de programmation pour faire progresser la pensée algorithmique, très visuel et davantage poussé pour l'interaction avec un utilisateur. Le langage Python se veut plus général et permet de faire des mathématiques plus poussées, avec un réinvestissement des renvois d'une fonction pour une autre. C'est une raison à l'utilisation de la fonction `return` plutôt que la fonction `print`.

Il est cependant toujours possible d'écrire par exemple :

```
return "un triplet est ",171,n,int(c)
```

Le résultat est ci-contre.