Thème : calculs numériques

Multiples

TI-83 Premium CE

python™

Résumé : dans cette activité, les élèves doivent programmer afin d'obtenir les multiples d'un nombre dans une liste.

Mots-clés : multiples ; fractions ; liste ; boucles itératives

Compétences visées

Chercher : « observer, s'engager dans une démarche, expérimenter en utilisant éventuellement des outils logiciels » ; le but ici est de démarrer le travail de manière autonome et de poursuivre par l'outil informatique.

Calculer : « effectuer un calcul automatisable à la main ou à l'aide d'un instrument (calculatrice, logiciel) » ; la recherche systématique d'un dénominateur commun passe par des calculs automatisés.

Situation déclenchante

Les élèves ont des difficultés sur le calcul numérique, en particulier sur les fractions. La somme ou différence de deux fractions de dénominateurs différents impliquent de connaître les multiples de ceux-ci afin de pouvoir faire le calcul demandé. Certains élèves n'ont pas acquis l'optimisation de recherche du plus petit multiple commun (ppcm ou ppmc).

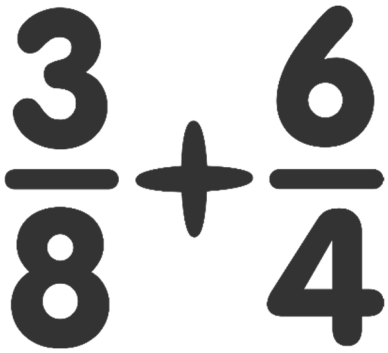


Image libre de droits d'après Pixabay

Problématique

Comment obtenir, par programmation, le ppcm ou ppmc de deux nombres entiers donnés ?

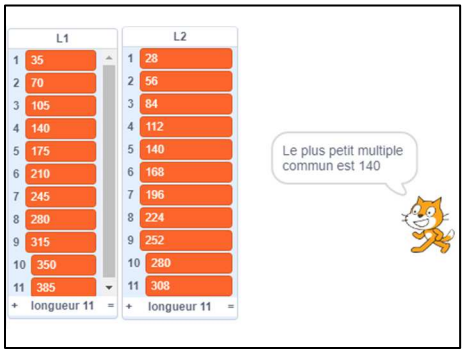
Multiples

Scénario pédagogique

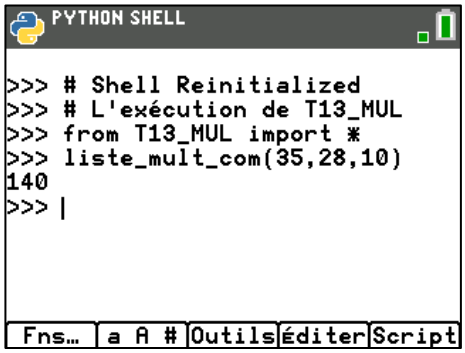
- **Avec la classe :** à l'occasion d'un exercice ou d'une correction de devoir sur le calcul fractionnaire, mettre au défi les élèves de programmer une aide pour la somme ou différence de fractions.
- **En groupe de 2 à 4 élèves :** les élèves doivent faire une ébauche d'algorithme permettant la résolution 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 script, notamment par la maîtrise de fonctions dédiées : les listes par exemple, et notamment le parcours des éléments d'une liste dans une boucle `for` ou une structure conditionnelle `if`.
- **Mise en commun :** les élèves expliquent leurs démarches.
- **Preuve :** les élèves peuvent montrer qu'il existe toujours au moins un dénominateur commun.
- **Pour les élèves les plus en avance :** il est possible de leur proposer un ou plusieurs prolongements possibles, décrit en fin de fiche.
- **Difficultés rencontrées :**
    - la gestion des listes.

Voici les visuels à l'issue des programmes :

en Scratch

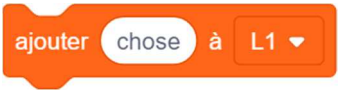


avec la TI-83 Premium CE Edition Python

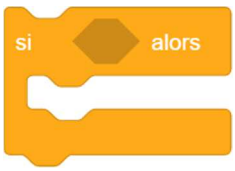


Multiples

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
	Ajouter un élément à une liste préalablement créée.	`L1.append(élément)` *Append* signifie « ajouter » en anglais. « `élément` » est donc ajouté à la suite des autres éléments dans la liste `L1`.

	Créer une boucle répéter.	`for i in range n:`
		`••instructions`
		
		Par défaut, les valeurs de `i` commencent à 0 et augmentent avec un pas de 1.
		L'indentation avec •• est nécessaire pour définir les instructions à exécuter dans un bloc du code Python.

	Créer le i-ième multiple du premier élément de la liste L1	`i*n` `n` doit être défini comme le premier élément de la liste.
	Créer une structure de contrôle conditionnelle.	`if condition:` `••instructions`

	Tester si l'élément i de la liste L1 est égal ou non à l'élément j de la liste L2.	`for élément in L1:`
		`••if élément in L2:`
		`••••instructions`
		
		Cette programmation est expliquée dans la partie suivante « Avec Python ».

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

Multiples

Avec Python

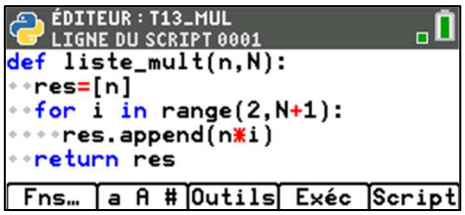
L'un des codes principaux est à construire complétement par les élèves.

Ce code est composé de plusieurs fonctions :

- La fonction `liste_mult` de paramètres `n` et `N` qui sont respectivement le nombre dont il faut trouver les multiples et l'étendue.

  C'est une fonction qui sera utilisée par les scripts principaux pour créer la liste des N multiples du nombre n.

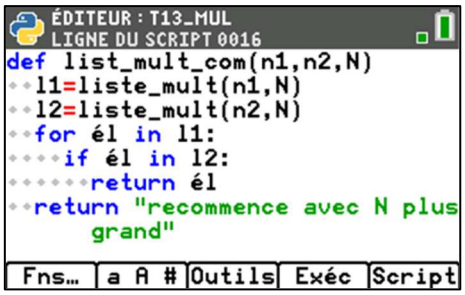
  Il faut donc commencer la boucle `for` à 2 pour la finir à `N+1`.



- La fonction `list_mult_com` de paramètres `n1` ; `n2` et `N` qui sont respectivement les deux nombres pour lesquels il faut déterminer les multiples et l'étendue.

  A noter la programmation des boucles : pour chaque élément de la liste `l1`, celui-ci est comparé à chacun des éléments de la liste `l2`.

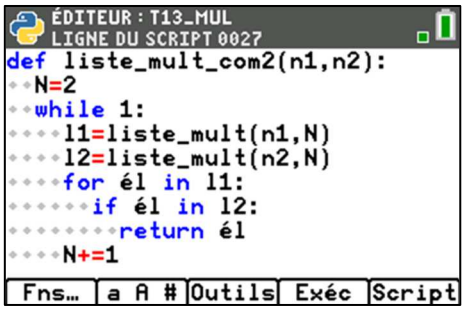
  On trouvera le `é` dans l'onglet `a A #`.



- La fonction `liste_mult_com2` de paramètres les nombres `n1` et `n2`.

  Cette fonction permet de trouver le plus petit multiple commun en recalculant la liste des multiples, en augmentant l'étendue de 1 à chaque itération de la boucle `while`.

  A noter que la boucle `while` se poursuit indéfiniment car la condition est toujours vraie. L'arrêt se produit lorsque le script renvoie un multiple commun, ce qui sera toujours le cas, au moins pour $n1 \times n2$.



- La fonction `ppmc` de paramètres les nombres `n1` et `n2`. Elle est laissée à l'appréciation du lecteur.

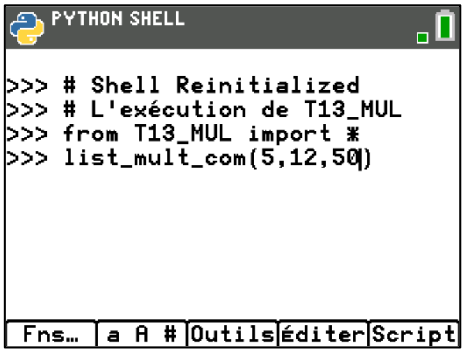
Une programmation possible est disponible sur le site TI : education.ti.com/fr/scratch-python

Multiples

Mode opératoire

Une fois le script exécuté, il faut appuyer sur la touche `[var]` : les fonctions définies dans le script apparaissent.

Par les flèches directionnelles, il faut sélectionner la fonction `list_mult_com`, valider par `Ok`, puis ajouter les deux nombres pour lesquels on souhaite avoir les multiples, et l'étendue, 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 :

- poursuivre la programmation en cherchant quel est le plus petit élément parmi les éléments finaux des listes `l1` et `l2` afin de n'augmenter que la liste qui a ce plus petit élément ;
- écrire un script prenant aussi les numérateurs des fractions afin de renvoyer les fractions avec les représentants de même dénominateurs.