Les fonctions avec fibonacci

bloc - 1 période

factorisation - fibonacci - suite

Compétences

Mathématique

- relever des régularités dans des suites de nombre
- Interpréter un tableau de nombres, un graphique, un diagramme
- Écrire des nombres sous une forme adaptée en vue de les comparer, de les organiser ou de les utiliser.
- Vérifier le résultat d'une opération

Éducation par la technologie

- Imaginer des dispositifs expérimentaux simples et prendre des initiatives
- Élaborer un concept, un principe, une loi
- Dans le cadre d'une énigme, agencer les indices en vue de formuler au moins une question, une supposition ou une hypothèse
- Réfléchir aux pratiques mises en œuvre, évaluer la démarche suivie
- Confirmer ou infirmer un raisonnement par des arguments vérifiés

Objectifs

L'élève sera capable de :

- Manipuler un système tangible (micro:bit)
- Utiliser une interface de programmation par blocs
- Comprendre le concept de fonction et ses utilités
- Comprendre la factorisation
- Trouver et reconnaître, dans l'interface de programmation, les blocs liés aux fonctions
- Manipuler des fonctions dans un exercice donné
- Comprendre le principe d'une suite (Fibonacci)
- Résoudre un problème donné et proposer une solution
- Tester une solution
- Comprendre pourquoi une solution ne fonctionne pas et la corriger

Description de l'activité

Il arrive souvent qu'un programme doive faire la même chose mais à des moments différents. Plus le programme est grand et compliqué, plus cela arrive. Pour éviter de devoir recopier exactement la même chose plusieurs fois, on va utiliser des fonctions. Une fonction est en fait un bout de programme auquel on va donner un nom et, dès qu'on aura besoin de ce bout de programme, il suffira d'appeler cette fonction grâce à son nom.
Pour illustrer l'utilité des fonctions, les élèves vont créer un compteur de Fibonacci, c'est-à-dire un compteur qui va donner le Xème nombre de la suite de Fibonacci.

Matériel et ressources

Matériel nécessaire :	Ressources nécessaires :
• (par groupe de 2) – 1 micro:bit avec cable USB – 1 ordinateur • 1 ordinateur avec projecteur pour le professeur	• fiche explicative Fonctions

Déroulement

Étape 1 : Explications théoriques (10'-15')

Qui	Fait quoi	Matériel
Le professeur	Explique ce qu'est une fonction et l'intérêt d'en avoir	
Le professeur	présente la suite de fibonacci	

Étape 2 : Énoncé (5-10')

Qui	Fait quoi	Matériel
Les élèves	s'asseyent par 2 devant un ordinateur	ordinateurs
Le professeur	distribue les consignes	Fiches élève
Le professeur	Énonce les fonctionnalités du compteur de fibonacci	

Étape 3 : Implémentation (25')

Les moments de réflexion sont prévus tous ensemble au tableau, mais, en fonction du temps et du niveau de la classe, vous pouvez laisser aux élèves un temps de réflexion avant chaque mise en commun.

1. Quand on démarre, on initialise toutes les variables

Qui	Fait quoi	Matériel
Tous	réfléchissent à quelles sont les variables utiles (nb, nombre1 et nombre2)	au tableau
Tous	réfléchissent à combien doivent être initialisées les variables (nb=1, nombre1=0, nombre2=1)	au tableau
Les élèves	implémentent cette étape	ordinateur

2. Incrémenter et décrémenter le nombre (nb) avec B et A

Qui	Fait quoi	Matériel
Les élèves	implémentent l'incrémentation de nb lorsqu'on appuie sur B	ordinateur
Les élèves	implémentent la décrémentation de nb lorsqu'on appuie sur A	ordinateur

3. Créer la fonction "suivant"

Qui	Fait quoi	Matériel
Tous	réfléchissent à comment calculer le Xème nombre de fibonacci: 1. Les 2 cas de base (nb=1 et nb=2) 2. Les cas suivants (nombreX = nombreX-2 + nombreX-1)	au tableau
Les élèves	implémentent la fonction "suivant" qui calcule le nombre suivant de fibonacci et stocke les 2 derniers dans nombre1 et nombre2	ordinateur
Les élèves	implémentent le calcul du Xème nombre de fibonacci lorsqu'on secoue	ordinateur

4. Réinitialiser quand on appuie sur A+B

Si le temps manque, cette étape peut être ignorée (car il est possible de réinitialiser avec le bouton reset au dos du micro:bit)

Qui	Fait quoi	Matériel
Les élèves	implémentent la réinitialisation lorsqu'on appuie sur A+B	ordinateur
Tous	réfléchissent à une manière d'éviter de dupliquer du code	au tableau
Les élèves	implémentent une fonction "initialisation" et l'utilisent à bon escient	ordinateur

Test (5')

Le test peut être fait uniquement sur le simulateur de l'application web si le temps manque.

Qui	Fait quoi	Matériel
Le professeur	distribue un micro:bit et un cable par groupe	micro:bit
Les élèves	testent leur programme sur les micro:bit	ordinateur et micro:bit

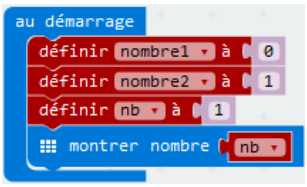
---

Solutions & Problèmes

1. Quand on démarre, on initialise toutes les variables

Quelles sont les variables utiles ? Il faut un nombre qui servira de "compteur" (on va l'appeler "nb"). Il faudra également des nombres pour stocker la suite de fibonacci. 2 suffiront puisque pour calculer un nombre de la suite, il suffit de connaître les 2 précédents. Nous les appelleront "nombre1" et "nombre2".

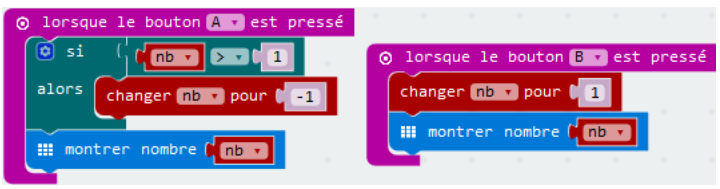
A combien doivent être initialisées les variables ? Les variables nombre1 et nombre2 étant respectivement le premier et le second nombre de la suite, on va initialiser nombre1 à 0 et nombre2 à 1. Notre compteur nb quant à lui commencera au minimum, soit 1.



Pour l'implémentation, il faut initialiser les variables au démarrage. Après avoir initialisé les variables, il est intéressant d'afficher la valeur de nb pour que l'utilisateur puisse savoir combien il vaut.
Si les élèves émettent la possibilité de faire une fonction pour l'initialisation (pour faire de l'abstraction), c'est une bonne idée et vous pouvez le faire. Ca anticipe la seconde partie de l'étape 4.

2. Incrémenter et décrémenter nb avec A et B

Pour l'incrémentation, il suffit de rajouter 1 à nb quand on appuie sur B. Pour la décrémentation, il faut ajouter -1 à nb lorsqu'on appuie sur A et UNIQUEMENT si nb n'est pas déjà à 1 (car ne peut pas descendre plus bas).

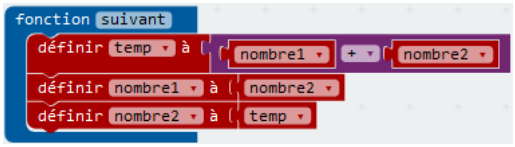


3. Créer la fonction "suivant"

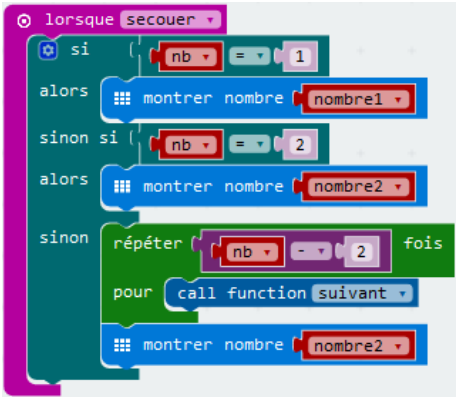
Pour la plupart des problèmes demandant de calculer une suite, la solution se fait en deux étapes: calculer le ou les cas de base (c'est-à-dire les cas qui ne demandent aucun calcul) puis calculer systématiquement les cas suivants.

- Cas de base : si on veut le premier ou le deuxième nombre, il n'y a rien à calculer. Il suffit de prendre nombre1 ou nombre2.
- Cas suivants : pour connaître n'importe quel nombre de la suite de fibonacci (sauf les 2 premiers), il faut additionner les deux précédents. Donc, pour calculer un nombre X, nombreX = nombreX-2 + nombreX-1. En d'autres termes, on a toujours besoin des 2 derniers nombres. Ça tombe bien, on a déjà 2 variables pour ça : nombre1 et nombre2.

Pour commencer, il faut créer la fonction "suivant" dont voici les spécifications : Calcule le nombre qui suit nombre1 et nombre2 dans la suite de fibonacci
Précondition : nombre1 et nombre2 sont (dans cet ordre) les nombres qui précèdent celui cherché dans la suite de fibonacci
Postconditions : nombre1(après) vaut nombre2(avant) et nombre2(après) vaut le nombre suivant dans la suite de fibonacci (soit nombre1(avant)+nombre2(avant))



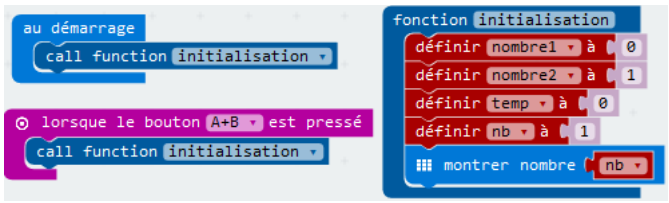
Ensuite, il faut afficher le *nb*ième nombre de fibonacci lorsqu'on secoue le micro:bit. Pour cela :



- on traite les cas de base :
  - si nb=1 alors on affiche nombre1;
  - sinon, si nb=2 alors on affiche nombre2;
- sinon, on calcule les cas suivants :
  - sinon, on calcule le nombre suivant autant de fois qu'il le faut. si nb = 3, il faut le calculer 1 seule fois. Si nb=4, 2 fois. Si nb=5, 3 fois. Vous l'aurez compris, il faut calculer le suivant nb-2 fois.

4. Réinitialiser lorsqu'on appuie sur A+B

Pour réinitialiser le programme, il faut "remettre à 0" les variables, c'est-à-dire les initialiser de nouveau (voir étape 1). Il faut également remettre "temp" à 0 pour éviter tout problème (effets de bord éventuels). Comme je viens de le dire, on va faire presque la même chose qu'à l'étape 1. On va donc dupliquer du code. Pour éviter cela, on peut faire une fonction "initialisation" qui sera appelée au démarrage et quand on appuie sur A+B.



Pourquoi une solution pourrait ne pas fonctionner ?

Attention à bien réinitialiser (A+B ou reset) avant chaque nouvelle utilisation pour bien remettre nombre1 à 0 et nombre2 à 1.

Évaluation

Manipuler un système tangible (micro:bit) [./1]

	non	oui
Appuyer sur les boutons et secouer le micro:bit	0	1

Utiliser une interface de programmation par blocs [./0.5]

	non	oui
Ajouter et emboîter des blocs	0	0.5

Comprendre le concept de fonction et ses utilités [./4]

	non	oui
Savoir ce qu'est une fonction : un morceau de code réutilisable	0	1
Citer l'**abstraction** comme utilité	0	1
Citer **le découpage en sous-problèmes** comme utilité	0	1
Citer la **factorisation** comme utilité	0	1

Comprendre la factorisation [./1]

	non	oui
Savoir ce qu'est la factorisation : éviter d'avoir plusieurs fois le même code à des endroits différents	0	1

Trouver et reconnaître, dans l'interface de programmation, les blocs liés aux fonctions [./2.5]

	non	oui
Trouver le menu contenant les blocs liés aux fonctions	0	0.5
Reconnaître le bloc d'appel de fonction	0	1
Reconnaître le bloc de déclaration de fonction	0	1

Manipuler des fonctions dans un exercice donné [./3]

	non	oui
Créer une fonction	0	1
Implémenter une fonction	0	1
Appeler une fonction	0	1

Comprendre le principe d'une suite (Fibonacci) [./2]

	non	oui
Expliquer que un élément de la suite est la somme des 2 éléments précédents	0	1
Expliquer que la suite commence par 0 puis 1	0	1

Résoudre un problème donné et proposer une solution [./4]

	non	oui
Initialiser une variable et la modifier (incrémenter et décrémenter)	0	0.5
Réinitialiser lorsqu'on appuie sur A+B	0	0.5
Implémenter la fonction "suivant"	0	1
Appeler la fonction "suivant" au bon moment	0	1
Programmer un compteur fonctionnel	0	1

Tester une solution [./1]

	non	oui
Tester la solution (sur le micro:bit ou l'émulateur)	0	1

Comprendre pourquoi une solution ne fonctionne pas et la corriger [./1]

	non	oui
Ne pas avoir d'erreur ou les corriger s'il y en a	0	1