La méthode de Héron

2 heures

Python

- ☐ Utiliser une boucle
- ☐ Connaitre la différence d'utilisation des deux types de boucle

1 Situation de recherche

L'algorithme de Héron permet de déterminer des valeurs approchées de racines carrées d'entiers.
Pour déterminer une valeur approchée de $\sqrt{a}$, où a est un entier naturel, aussi écrit $a \in \mathbb{N}$, on doit calculer les valeurs successives de u_1, u_2, u_3, u_4, ... avec $u_0 = a$ et $u_1 = \frac{1}{2}\left(u_0 + \frac{a}{u_0}\right)$, $u_2 = \frac{1}{2}\left(u_1 + \frac{a}{u_1}\right)$, $u_3 = \frac{1}{2}\left(u_2 + \frac{a}{u_2}\right)$, ...

1. Cas pour $a = 2$
   (**a**) Dans une même colonne d'une feuille de calcul, saisir la valeur u_0 puis la formule permettant de calculer u_1. Copier cette formule vers le bas afin de calculer u_2, u_3, ...
   (**b**) A partir de quel terme de la suite u_0, u_1, u_2, u_3, ... obtient-on une valeur approchée de $\sqrt{2}$ à 10^{-7} près ? Utiliser l'éditeur de formule pour déterminer $\sqrt{2}$.
2. Autres cas
   (**a**) Compléter plusieurs autres colonnes de la feuille de calcul pour déterminer des valeurs approchées de $\sqrt{3}$, $\sqrt{5}$, $\sqrt{6}$, $\sqrt{7}$.

Dans cette activité on a créé pour chaque valeur de a une suite de nombres qui apparaissent dans les cellules du tableur. Cette suite de nombre peut s'écrire :

$$\begin{cases} u_0 = a \\ u_{n+1} = \frac{1}{2}\left(u_n + \frac{a}{u_n}\right) \end{cases}$$

? **Situation à programmer**

a et n sont deux entiers naturels donnés par l'utilisateur.

1. Construire un algorithme qui permet de déterminer à partir de quel terme de la suite, l'approximation donnée par la méthode de Héron du nombre $\sqrt{a}$ est inférieur à 10^{-n}.
2. Programmer cet algorithme sous Python. Pour utiliser les formules mathématiques sous Python, on doit importer la bibliothèque mathématique avec `from math import *`
   La fonction Racine Carré s'écrit en python sqrt. Ainsi, pour écrire $\sqrt{a}$, $a \geq 0$, on tape sqrt(a).
3. On peut améliorer l'algorithme pour qu'il donne le nombre de valeurs avant que la condition soit remplie.

Un tel algorithme est appelé **algorithme de seuil**.

Définition 1. ***Itération. Boucle***

Une itération est un procédé qui répète une même action. Une boucle est une itération.

Remarque

Il existe deux types de boucles :

- Lorsque le nombre d'itérations est *a priori* connue
- Lorsque le nombre d'itérations est inconnue

La boucle Pour

Syntaxe

L'écriture algorithmique de $n+1$ itérations

```
POUR i ALLANT_DE 0 A n
    DEBUT_POUR
    Action
    FIN_POUR
```

La programmation en Python de n itérations. i varie de 0 à n inclus.

```
for i in range(n+1) :
        action
```

Exemple

Calculer la somme des 5 premiers nombres entiers naturels non nuls.

```
somme = 0
for i in range(1,6) :
        somme = somme + i
print(somme)
```

Tester le code

```
somme = 0
for i in range(1,6) :
        somme = somme + i
print(somme)
```

Que se passe-t-il si `print(somme)` est dans la boucle?

Remarque

L'instruction `for i in range(1,6)` calcule de 1 jusqu'à 5 inclus.

2 Application directe

Créer un programme qui demande à l'utilisateur un nombre entier naturel n et qui calcule le produit des entiers naturels non nuls inférieurs où égal à n. On appelle se nombre factorielle n et on l'écrit $n!$.

3 Application directe

On souhaite déterminer les images des entiers compris entre 1 à 5 par la fonction f définir par $f(x) = x^2$.

1. Écrire un algorithme.
2. Coder cet algorithme en Python.

4 Application directe

Variables :	i et n sont des entiers naturels
	u est un réel
Entrée :	Saisir n
Initialisation :	Affecter à u la valeur $0,6931$
Traitement :	Pour i variant de 1 à n
	|Affecter à u la valeur $\frac{1}{i} - u$
	Fin de Pour
Sortie :	Afficher u

A l'aide de cet algorithme, on a obtenu le tableau de valeurs suivant.

n	0	1	2	3	4	5	10	100
u_n	0,693 1	0,306 9	0,193 1	0,140 2	0,109 8	0,090 2	0,047 5	0,005 0

Programmer avec Python cet algorithme et retrouver ces résultats.

La boucle Tant que

Syntaxe

L'écriture algorithmique de la boucle "TANT QUE"

```
TANT_QUE condition FAIRE
    DEBUT_TANT_QUE
    action1
    FIN_TANT_QUE
```

La programmation en Python

```python
while condition :
    action
```

Exemple

Déterminer l'entier naturel n_0 à partir duquel la somme des premiers entiers naturels inférieur ou égal à n_0 est strictement supérieure à 15.

Etapes	somme	i	condition
0	0	1	Vraie
1	1	2	Vraie
2	3	3	Vraie
4	6	4	Vraie
5	10	5	Vraie
6	15	6	Faux

Tester le code

```python
somme = 0
i=1
while somme < 15 :
  somme = somme + i
  i=i+1
print(i)
```

Attention

Il faut être vigilant dans la position des lignes de code dans la boucle.

5 Application directe

Variables	n est un entier, u et M sont deux réels
Initialisation	u prend la valeur M n prend la valeur 0 Saisir la valeur de M
Traitement	Tant que $u > 50$ $u = 0,9 * u$ $n = n + 1$ Fin tant que
Sortie	Afficher n

1. Complète le tableau ci-dessous à l'aide de l'algorithme.

Étapes	u	n	Condition
0	80	0	Vraie
1			
2			
3			
4			
5			

2. Programmer avec Python cet algorithme pour $M = 80$ et retrouve les valeurs.
3. Expliquer le rôle de ce programme.