Boucles while (2) — TP7

Exercice 1 — Pour s'échauffer (20 min)

Q1 Soit (u_n) la suite définie par $u_1 = 1$ et pour tout $n \in \mathbb{N}^*$, $u_{n+1} = \dfrac{3-u_n}{2}$. Écrire deux fonctions prenant en argument $n \in \mathbb{N}^*$ et renvoyant u_n. La première utilisera une boucle `for`, la deuxième, une boucle `while`.

Q2 Pour tout $n \geq 2$ on note $S_n = \displaystyle\sum_{k=2}^{n} \sqrt{k}$. Écrire deux fonctions prenant en argument $n \geq 2$ et renvoyant S_n. La première utilisera une boucle `for`, la deuxième, une boucle `while`.

Q3 Soient (u_n) et (v_n) les suites définies par $u_0 = 3$, $v_0 = 2$ et pour tout $n \in \mathbb{N}$,

$$u_{n+1} = -2u_n + v_n, \text{ et } v_{n+1} = u_n + 2v_n.$$

Écrire une fonction prenant en argument un réel M et renvoyant le premier couple (u_n, v_n) tel que $|u_n - v_n| > M$. Quel type de boucle doit-on utiliser ici et pourquoi ?

Exercice 2 — Approximation de $\ln(2)$ (15 min)

Dans cet exercice, on souhaite écrire une fonction Python prenant en argument une valeur ε et renvoyant une approximation de $\ln(2)$ qui soit d'autant meilleure que ε est petite.

Pour cela, on va utiliser la quantité $S_n = \displaystyle\sum_{k=0}^{n} \frac{(-1)^k}{k+1}$. On peut démontrer (mais on ne demande pas de le faire) que : $\displaystyle\lim_{n\to+\infty} S_n = \ln(2)$.

On souhaite donc écrire une fonction Python renvoyant une valeur de S_N qui soit proche de $\ln(2)$. Pour savoir si S_N est proche de $\ln(2)$, on choisit de fixer $\varepsilon > 0$ petit, et de dire sur S_N est proche de $\ln(2)$ lorsque $|S_{N+1} - S_N| \leq \varepsilon$.

Q1 *(Sans ordinateur)* Simplifier $S_{N+1} - S_N$ et reformuler la condition $|S_{N+1} - S_N| \leq \varepsilon$ de manière plus simple.

Q2 Écrire une fonction Python prenant en argument ε et renvoyant l'approximation de $\ln(2)$ correspondante calculée par la méthode décrite ci-dessus.

> **Remarque**
> Pour obtenir la valeur de $\ln(2)$ en Python, on utilisera la commande `np.log(2)` après avoir importé la bibliothèque `numpy` via la commande `import numpy as np`.

Exercice 3 Approximation de π (15 min)

La formule de Brent-Salamin (des noms de deux mathématiciens des années 1970) permet d'obtenir rapidement une bonne approximation de π (elle fut utilisée en 1999 pour obtenir plus de 206 millions de décimales de π !).

Cette méthode consiste à définir $a_0 = 1$, $b_0 = \frac{1}{\sqrt{2}}$, $t_0 = \frac{1}{4}$ et $p_0 = 1$ puis pour tout $n \geq 0$,

$$t_{n+1} = t_n - p_n \left(\frac{a_n - b_n}{2}\right)^2 \ , \ a_{n+1} = \frac{a_n + b_n}{2} \ , \quad b_{n+1} = \sqrt{a_n b_n} \ , \qquad \text{et} \qquad p_{n+1} = 2p_n.$$

Un théorème affirme alors que, lorsque a_n et b_n sont "proches", la valeur

$$\frac{(a_n + b_n)^2}{4t_n}$$

est une "bonne" approximation de π.
Écrire une fonction `approx` prenant en argument un réel `epsilon` et renvoyant l'approximation de π obtenue par cette méthode lorsque l'on s'arrête dès que $|a_n - b_n| \leq$ `epsilon`.
Testez votre fonction, puis écrivez une fonction `approx_bis` pour savoir également combien d'itérations ont été nécessaires pour obtenir le résultat.

Exercice 4 Conjecture de Syracuse (temps restant)

Soit $f : \mathbb{N}^* \longrightarrow \mathbb{N}^*$ la fonction définie par

$$f(k) = \begin{cases} \frac{k}{2} & \text{si } k \text{ est pair} \\ 3k+1 & \text{si } k \text{ est impair} \end{cases}$$

On appelle *«suite de Syracuse d'un entier N»* la suite (u_n) définie par $u_0 = N$ et $u_{n+1} = f(u_n)$ pour tout $n \geq 0$.

Q1 Calculer à la main la suite de Syracuse de 3. Que se passe-t-il lorsque la suite atteint la valeur 1 ?

> **Remarque**
> La conjecture de Syracuse affirme que toutes les suites de Syracuse des entiers positifs atteignent la valeur 1 au bout d'un certain temps. Cette conjecture a été vérifiée pour tous les entiers naturels N inférieurs à 2^{62}, mais on ignore encore si elle est vraie.

Q2 Écrire une fonction `f` prenant en argument un entier `k` ≥ 1 et renvoyant $f($`k`$)$.

Q3 Écrire une fonction `syracuse` prenant en argument deux entiers `N` et `n` et renvoyant la valeur `syracuse(N,n)` $= u_{\mathrm{n}}$ où (u_n) est la suite de Syracuse de `N`. On vérifiera que `syracuse(15,9)` renvoie la valeur 40.

Q4 Écrire une fonction `TempsVol` prenant en argument un entier `N` et renvoyant la plus petite valeur de n telle que le n-ème terme de la suite de Syracuse de l'entier N vaut 1 (on suppose que ce terme existe, c'est-à-dire que la conjecture est vérifiée). On vérifiera que `TempsVol(15)` renvoie la valeur 17.

Q5 Que se passe-t-il si on choisit $N < 0$? On pourra consulter la vidéo du youtuber "El Jj" sur le sujet : `https://www.youtube.com/watch?v=BP2G28694z8`.