Devoir maison du 8 février 2021
TSI 1

Ce devoir est à faire en binôme en rédigeant une copie par personne. Les codes Python doivent être regroupés dans un script portant vos noms de famille et déposé sur :

https://herissonceleste.hopto.org/nextcloud/index.php/s/HcqipjgnjCoNpEE

Exercice 1 — Une première approximation

Dans tout cet exercice, a désigne un réel strictement positif et on considère la fonction f_a qui à x associe

$$f_a(x) = \frac{x + \frac{a}{x}}{2}$$

1. Quel est l'ensemble de définition de f_a ? f_a est-elle continue, dérivable sur cet ensemble ? Préciser également la parité de f_a.
2. En détaillant les étapes, dresser le tableau de variations de f_a sur $\mathbb{R}_*^+$.
   On précisera et on justifiera les limites de f_a en 0^+ et $+\infty$.
3. Résoudre l'équation $f_a(x) = x$ sur l'ensemble $\mathbb{R}_*^+$.

   On considère maintenant la suite (u_n) définie par :

$$\begin{cases} u_0 > 0 \\ u_{n+1} = \dfrac{u_n + \frac{a}{u_n}}{2} = f_a(u_n) \end{cases}$$

   On admettra sans difficulté que u_n est positif pour tout n.
4. Montrer que, pour tout entier n,

$$u_{n+1} - \sqrt{a} = \frac{\left(u_n - \sqrt{a}\right)^2}{2u_n}$$

5. (a) Déduire de l'égalité précédente ou du tableau de variation de f_a, que pour tout $n \geq 1$, $u_n \geq \sqrt{a}$.
   (b) En étudiant le signe de $u_{n+1} - u_n$, prouver que (u_n) est décroissante à partir du rang 1.
6. (a) Prouver que la suite (u_n) converge, puis déterminer sa limite.
7. Pour simplifier, on suppose ici que $u_0 > \sqrt{a}$; de sorte que l'on a pour tout n, $u_n > \sqrt{a}$.
   (a) À partir de l'égalité obtenue la question 4, prouver que :

$$u_{n+1} - \sqrt{a} \leq \frac{\left(u_n - \sqrt{a}\right)^2}{2\sqrt{a}}$$

   (b) En déduire que, pour tout $n \geq 1$:

$$u_n - \sqrt{a} \leq \frac{\left(u_0 - \sqrt{a}\right)^{2^n}}{\left(2\sqrt{a}\right)^{2^n - 1}}$$

Algorithme n° 1: Écrire en Python une fonction `heron` qui prend en entrée un nombre $a > 0$ et un entier n et qui retourne la valeur de u_n.

On pourra vérifier la rapidité de l'algorithme avec $a = 2$ et $n = 4$.

Exercice 2 — Méthode d'Al Kashi

Cet exercice revient sur une méthode développée par Al Kashi [1] pour approcher le nombre π.

Dans toute la suite on considère un cercle Γ de rayon 1 et de centre O.

Partie A : résultat préliminaire

Soient A et B deux points de Γ, distincts et non diamétralement opposés.
On pose I le milieu de $[AB]$ et C le point à l'intersection entre la demi-droite $[OI)$ et Γ.
a et b désignent respectivement les distances AB et AC.

1. Calculer OI en fonction de a. *On pourra utiliser le triangle OIA dont on justifiera la nature.*
2. En déduire IC en fonction de a, puis en déduire AC en fonction de a. *On pourra utiliser le triangle IAC dont on justifiera la nature.*

Partie B : le vif du sujet

L'idée d'Al Kashi est d'approximer le périmètre de Γ par une suite de périmètres de polygones réguliers inscrits possédant de plus en plus de côtés.
On initialise le procédé de la manière suivante :

(Étape 1) On construit un carré inscrit dans Γ.
La première approximation de π est donnée par le périmètre du carré divisé par 2.

(Étape 2) On remplace le carré par un octogone régulier inscrit dont un sommet sur deux correspond à l'un des sommets du carré de départ.
La seconde approximation de π est donnée par le périmètre de l'octogone divisé par 2.

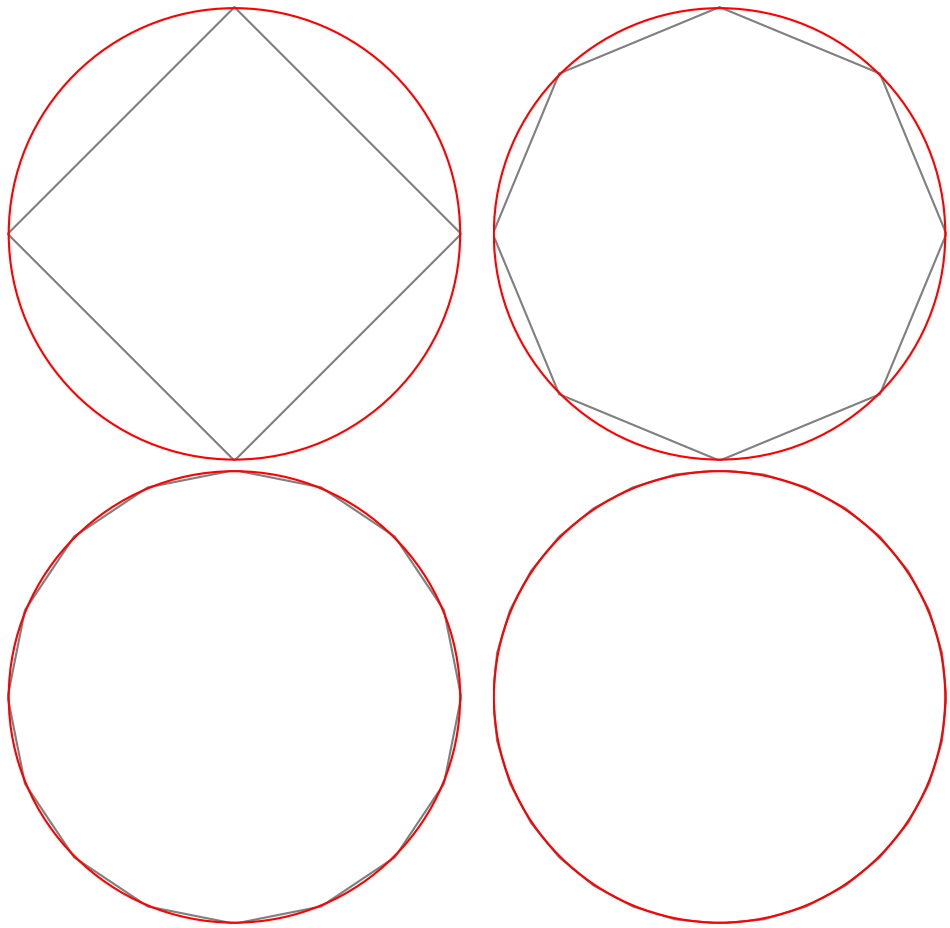
Puis on redivise chacun des côtés de l'octogone pour obtenir un 16-gone inscrit et ainsi de suite...
A l'étape n, on note C_n la longueur d'un côté du n-ième polygone construit et P_n le périmètre du n-ième polygone construit.

1. Calculer l'approximation de π à la première étape.
2. Donner le lien entre C_n et C_{n+1}. *On utilisera le résultat de la partie A.*
3. Combien de côtés possède le polygone à l'étape n ? En déduire une formule donnant P_n en fonction de C_n.
4. Prouver que la suite P_n converge. *On pourra examiner le sens de variation de cette suite.*

Algorithme n° 2: Faire un script Python qui affiche les 20 premières valeurs de C_n et P_n.

Les dessins plus bas illustrent le début du procédé. En rouge on a tracé le cercle de rayon 1 que l'on cherche à approximer. On constate qu'au bout de quatre étapes, le cercle et le polygone peuvent difficilement être distingués.

1. Mathématicien perse du XIVe siècle



Exercice 3 (facultatif) — Fonctions python pour l'arithmétique

Les deux parties ne sont pas indépendantes. On suppose que tous les nombres de cet exercice sont des entiers naturels.

> ⚠ Dans le code à produire, il est interdit d'utiliser les fonctions d'arrondi, la division, la fonction racine ainsi que la fonction modulo (notée %). Tout doit être réalisé avec des boucles, des tests, et des opérations de soustraction, addition et multiplication ! On pourra également utiliser des fonctions associées aux listes telles que `count` ou `remove` dont on trouvera une description sur internet.

Partie A : fonctions préliminaires

Le crible d'Ératosthène est une méthode qui permet d'obtenir tous les nombres premiers inférieur à un certain entier noté n. Voici son principe :

- on fabrique une liste l qui contient tous les entiers entre 1 et n
- pour chaque entier k de l'intervalle $\left[2;\ \sqrt{n}\right]$, on élimine de l l'ensemble des $k \times i$, avec i qui parcourt tous les entiers de l'intervalle $\left[2;\ \dfrac{n}{k}\right]$

- *Algorithme n° 3:* Créer une fonction appelée `crible` qui prend pour argument n et qui renvoie la liste des nombres premiers inférieurs ou égaux à n.
- *Algorithme n° 4:* Combien y-a-t-il de nombres premiers inférieurs ou égaux à 15 000 ? *On pourra répondre au moyen d'une ligne de commande dans le script.*
- *Algorithme n° 5:* Fabriquer une fonction appelée `div` qui prend comme argument deux nombres n et d et qui renvoie le quotient et le reste de la division euclidienne de n par d.

  On pourra faire les deux tests suivants :
  - `div(12,5)` doit renvoyer 2, 2 car $12 = 5 \times 2 + 2$;
  - `div(5,11)` doit renvoyer 0, 5 car $5 = 11 \times 0 + 5$.

Partie B : décomposition en facteurs premiers et applications

La décomposition en facteurs premiers d'un nombre entier positif x est la recherche d'une liste de nombres premiers $(p_i)_{1\leq i\leq n}$ supérieurs ou égaux à 2 et associés à des coefficients entiers positifs $(\alpha_i)_{1\leq i\leq n}$ tels que

$$x = \prod_{1\leq i\leq n} p_i^{\alpha_i} = p_1^{\alpha_1} \times p_2^{\alpha_2} \times \cdots \times p_n^{\alpha_n}$$

Par exemple, la décomposition en facteurs premiers de 360 est

$$360 = 2^3 \times 3^2 \times 5^1$$

- *Algorithme n° 6:* Fabriquer une fonction appelée `decomposition` qui prend pour argument un entier et qui renvoie une liste correspondant aux facteurs premiers, présents autant de fois que dans la décomposition.

  On pourra faire les deux tests suivants :
  - `decomposition(360)` doit renvoyer [2,2,2,3,3,5] ;
  - `decomposition(23)` doit renvoyer [23] car 23 est un nombre premier.
- *Algorithme n° 7:* Fabriquer une fonction `facteurs_pgcd` qui prend en entrée deux nombres n_1 et n_2 et qui renvoie une liste obtenue par l'intersection entre les listes des facteurs premiers de ces deux nombres.

  On pourra faire le test suivant :
  - `facteurs_pgcd(12,5)` doit renvoyer une liste vide ;
  - `facteurs_pgcd(54,90)` doit renvoyer [2,3,3] car $2^1 \times 3^2$ est le plus grand commun diviseur de ces deux nombres.
- *Algorithme n° 8:* Fabriquer une fonction `facteurs_ppcm` qui prend en entrée deux nombres n_1 et n_2 et qui renvoie une liste obtenue par la réunion entre les listes des facteurs premiers de ces deux nombres.

  On pourra faire le test suivant :
  - `facteurs_ppcm(12,5)` doit renvoyer [2,2,3,5] car $2^2 \times 3^1 \times 5^1$ est le plus petit multiple commun de ces deux nombres ;
  - `facteurs_ppcm(75,45)` doit renvoyer [3,3,5,5] car $3^2 \times 5^2$ est le plus petit multiple commun de ces deux nombres.
- *Algorithme n° 9:* Fabriquer une fonction `simplification` qui prend en entrée deux nombres non nuls p et q et qui renvoie deux nombres p' et q' tels que $\dfrac{p}{q} = \dfrac{p'}{q'}$ avec p' et q' premiers entre eux.

  On pourra faire le test suivant :
  - `simplification(12,6)` doit renvoyer 2, 1 car $\dfrac{12}{6} = \dfrac{2}{1}$;
  - `simplification(12,100)` doit renvoyer 3, 25 car $\dfrac{12}{100} = \dfrac{3}{25}$.