Devoir maison du 11 février 2019
TSI 1

Ce devoir est à faire en binôme en rédigeant une copie par personne. Les codes Python doivent être regroupés dans un script déposé sur https://edu-nuage.ac-versailles.fr/s/GiqqNo2MoQVMCa7

Exercice 1 — Approximation de π par la 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.

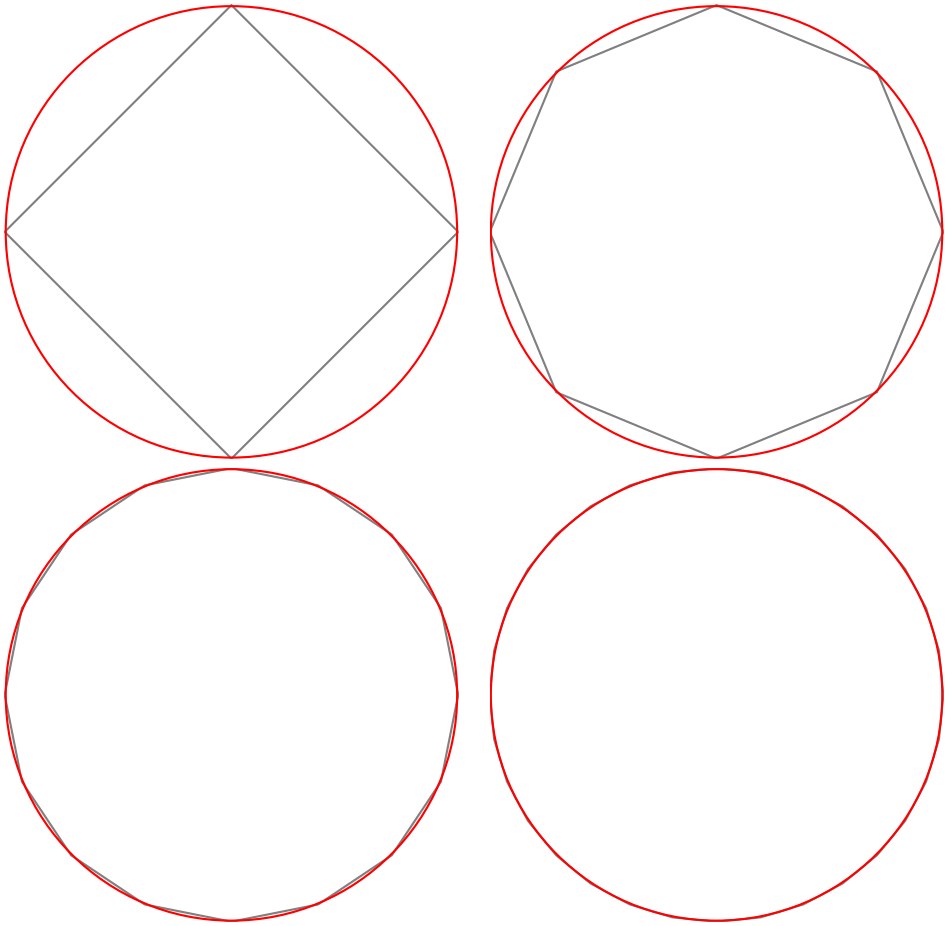
Puis on redivise chacun des côtés de l'octogone pour obtenir un 16-gone inscrit et ainsi de suite...
A l'étape n, on note C_n la longueur d'un côté du n^{e}polygone construit et P_n le périmètre du n^{e}polygone construit.

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^o 1: 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 XIV^{e} siècle



Exercice 2

Soit n un entier naturel non nul et $a_1 \leq a_2 \leq \cdots \leq a_n$, n nombres réels. Calculer

$$\inf_{x \in \mathbb{R}} \sum_{1 \leq i \leq n} |x - a_i|$$

Exercice 3 — 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° 2: 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° 3: 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° 4: 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° 5: 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° 6: 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° 7:* 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° 8:* 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 $\frac{p}{q} = \frac{p'}{q'}$ avec p' et q' premiers entre eux.

  On pourra faire le test suivant :

  - `simplification(12,6)` doit renvoyer 2, 1 car $\frac{12}{6} = \frac{2}{1}$;
  - `simplification(12,100)` doit renvoyer 3, 25 car $\frac{12}{100} = \frac{3}{25}$.