Exercice 1 Approximation de π à l'aide d'une intégrale

Soit f une fonction dérivable sur l'intervalle $[a, b]$; on suppose qu'il existe un réel $M_1 \geqslant 0$ tel que pour tout $x \in [a, b]$ on a $\mid f'(x) \mid \leqslant M_1$.

Pour tout $n \in \mathbb{N}^\star$, on pose

$$R_n = \sum_{k=1}^{n} \frac{b-a}{n} f\left(a + k\frac{b-a}{n}\right).$$

On démontre que

$$\mid R_n - \int_a^b f(x)dx \mid \leqslant \frac{M_1 \mid b-a \mid^2}{2n}.$$

Soit f la fonction qui à tout réel x associe $f(x) = \dfrac{1}{x^2+1}$.

Ecrire un programme qui demande un réel $\varepsilon > 0$, puis calcule et affiche une valeur approchée à ε près de π.

Exercice 2 Polynôme de Lagrange

Soit $n > 0$ une suite strictement croissante $(x_0, x_2, ..., x_n)$ de $n+1$ nombres réels d'un intervalle I. Pour $k \in [\![0, n]\!]$, on définit la fonction polynôme L_k (polynôme de Lagrange) par

$$L_k(x) = \prod_{\substack{0 \leqslant j \leqslant n \\ j \neq k}} \frac{x - x_j}{x_k - x_j}$$

a) Ecrire une fonction python qui prend en entrée $X = (x_0, x_1, ..., x_n)$, un entier $k \in [\![0, n]\!]$ et un réel x, puis calcule et affiche le nombre $L_k(x)$.

b) Supposons que la bibliothèque

```
import matplotlib.pyplot as plt
```

est importée, Ecrire un programme qui tracer sur l'intervalle $[-3, 3]$ chacune des fonctions L_k correspond à la suite $X = (-2, -1, 0, 1, 2)$.

Exercice 3Algorithme de Newton

On veut résoudre l'équation $f(x) = 0$ par la méthode dite « l'algorithme de Newton ».

— Soit I un intervalle contenant une unique solution x_0 de l'équation $f(x) = 0$ pour f dérivable et de dérivée non nulle sur I et $a \in I$ (donc $f'(a) \neq 0$).

— On définit par récurrence la suite $(a_n)_{n>0}$ par : $a_0 = a$ et, pour tout entier naturel n, $a_{n+1} = a_n - \dfrac{f(a_n)}{f'(a_n)} = g(a_n)$ pourvu que $g(I) \subset I$.

On Montre que, pour tout entier n, a_{n+1} est l'abscisse du point d'intersection de l'axe des abscisses avec la tangente à la courbe de f au point d'abscisse a_n.

On admet que, dans des conditions adéquates, la suite $(a_n)_n > 0$ converge et que sa limite est solution de l'équation $f(x) = 0$. De plus, on montre que a_n est une valeur approchée de cette solution à ε près lorsque $\mid a_{n+1} - a_n \mid \leqslant \varepsilon$.

Ecrire un programme python, qui calcule et affiche une valeur approchée à ε près de la solution de l'équation $x + \ln(x) = 0$ dans l'intervalle $]0; +\infty[$, et le nombre d'itérations réalisées. $a; \varepsilon$ deux réels données par l'utilisateur.

Exercice 4 *Algorithme de HORNER*

Soit $P(x) = \sum_{i=0}^{n} a_i x^i$ est une fonction polynôme et x_0 un nombre réel donné.
Pour déterminer la valeur de $P(x_0)$ on utilise l'algorithme de Horner :
— on multiplie le premier coefficient par x_0 puis on ajoute au produit le second coefficient.
— On multiplie alors le nombre obtenu par x_0 puis on ajoute le troisième coefficient,
— etc ...on obtient alors

$$P(x_0) = ((...((a_n x_0 + a_{n-1})x_0 + a_{n-2})x_0 + ...)x_0 + a_1)x_0 + a_0$$

Ecrire une fonction python **eval_poly** prenant comme arguments, une fonction polynôme P, un réel x_0 et renvoie la valeur de $P(x_0)$ en utilisant la méthode de HORNER.

Exercice 5 EDHEC 2016

Pour chaque entier naturel n, on définit la fonction f_n par : $\forall x \in [n;+\infty[,\ f_n(x) = \int_n^x e^{\sqrt{t}}dt$.
On montre que
(a) f_n est de classe $\mathcal{C}^1$ sur $[n;+\infty[$,
(b) pour chaque entier naturel n, il existe un unique réel, noté u_n, élément de $[n;+\infty[$, tel que $f_n(u_n) = 1$.
(c) La suite (u_n) admet les propriétés suivantes :
— $\lim\limits_{n \longrightarrow +\infty} u_n = +\infty$.
— $\forall n \in \mathbb{N}, e^{-\sqrt{u_n}} \leqslant u_n - n \leqslant e^{-\sqrt{n}}$
1) Utiliser les propriétés de f_n et (u_n) pour écrire un programme python, qui permet d'afficher un entier n pour lequel $u_n - n$ est inférieur ou égal à 10^{-4}.
2) Le script affiche l'une des trois valeurs $n = 55$, $n = 70$ et $n = 85$. Préciser laquelle en prenant $2,3$ comme valeur approchée de $ln 10$.

Exercice 6 EML 2015

On considère l'application $f : \mathbb{R} \longrightarrow \mathbb{R}, x \longmapsto f(x) = x^3 e^x$, et la série convergente $\sum\limits_{n \geqslant 1} \dfrac{1}{f(n)}$.

On note $S = \sum\limits_{n=1}^{+\infty} \dfrac{1}{f(n)}$ on a

$$\forall n \in \mathbb{N}^* \mid S - \sum_{k=1}^{n} \frac{1}{f(k)} \mid \leqslant \frac{1}{(e-1)\, e^n}$$

Ecrire fonction en Python qui calcule une valeur approchée de S à 10^{-4} près.

Exercice 7 EML 2016

On s'intéresse dans cet exercice, pour tout x de $\mathbb{R}$, à la série $\sum\limits_{n \in \mathbb{N}*} \dfrac{(-1)^{n+1}}{n^x}$

1. ON ADMET QUE :

pour tout x de $\mathbb{R}^-$, la série $\sum\limits_{n \in \mathbb{N}} \dfrac{(-1)^{n+1}}{n^x}$ diverge.

Soit $x \in \mathbb{R}^{+*}$. On note, pour tout n de $\mathbb{N}^*$, $u_n = \sum\limits_{k=1}^{n} \dfrac{(-1)^{k+1}}{k^x}$.

a. les suites $(u_{2p})_{p \in \mathbb{N}^*}$ et $(u_{2p-1})_{p \in \mathbb{N}^*}$sont adjacentes et elles convergent vers une même limite notée$S(x)$.
b. $\forall \varepsilon > 0, \exists n_0 \in \mathbb{N}^* / \forall n \geqslant n_0, |u_n - S(x)| \leqslant \varepsilon$.
c. la série $\sum\limits_{n \in \mathbb{N}^*} \dfrac{(-1)^{n+1}}{n^x}$ converge et que l'on a : $S(x) = \sum\limits_{k=1}^{+\infty} \dfrac{(-1)^{k+1}}{k^x}$.
d. $\forall p \in \mathbb{N}^*,\ \ u_{2p} \leqslant S(x) \leqslant u_{2p+1} \leqslant u_{2p-1}$

e. $\forall n \in \mathbb{N}^*, |u_n - S(x)| \leqslant \dfrac{1}{(n+1)^x}$.

2. En déduire de ce qui précède une **fonction en Python** qui, étant donnés deux réels $x > 0$ et$\varepsilon > 0$, renvoie une valeur approchée de $S(x)$ à ε près.

Exercice 8 Ecricome 2019 ECS

On considère la suite $(I_n)_n > 0$ définie par :

$$\forall n \in N, \quad I_n = \int_0^{\frac{\pi}{2}} (cos(t))^n dt$$

1. Montrer que I_n est bien défini pour tout$n \in \mathbb{N}$.
Calculer I_0, I_1 et I_2.
2. Etudier la monotonie de la suite (In)n>0 : En déduire que la suite $(I_n)_{n>0}$ converge.
3. À l'aide d'une intégration par parties, montrer que $\forall n \in \mathbb{N},\ I_{n+2} = (n+1)(I_n - I_{n+2})$
4. En déduire que :

$$\forall n \in \mathbb{N},\ I_{2n} = \frac{(2n)!}{(2^n n!)^2}\frac{\pi}{2} \text{ et } I_{2n+1} = \frac{(2^n n!)^2}{(2n+1)!}$$

5. Compléter la fonction I suivante, qui prend en entrée un entier positif n, afin qu'elle retourne une liste Y qui contient les $2n+2$ premiers termes de la suite $(I_n)_{n>0}$:

```
def I(n):
u=...........
u[0]=.........
u[1]=.........
    for ......
        ......

    y=........
    return y
```

6. Ecrire une fonction en Python qui prend en entrée un entier naturel n et qui renvoie en sortie le terme de rang n de la suite des sommes partielles associée à la série $\sum_{n\geqslant 0} I_n$

Exercice 9 Ecricome 2015 ECS

1. On pose pour tout entier naturel n, $a_n = sin\left(\dfrac{\pi}{3{\times}2^n}\right)$ et $b_n = cos\left(\dfrac{\pi}{3{\times}2^n}\right)$.

L'étude mathématique a permis de montrer que : $\forall n \in \mathbb{N}$, $a_{n+1} = \sqrt{\dfrac{1-b_n}{2}}$ $(*)$ et$b_{n+1} = \sqrt{\dfrac{1+b_n}{2}}$ $(**)$

Puis que pour tout $n \in \mathbb{N}$: $\quad 9 \times 2^n \dfrac{a_n}{2+b_n} < \pi < 2^n \left(2a_n + \dfrac{a_n}{b_n}\right)$,

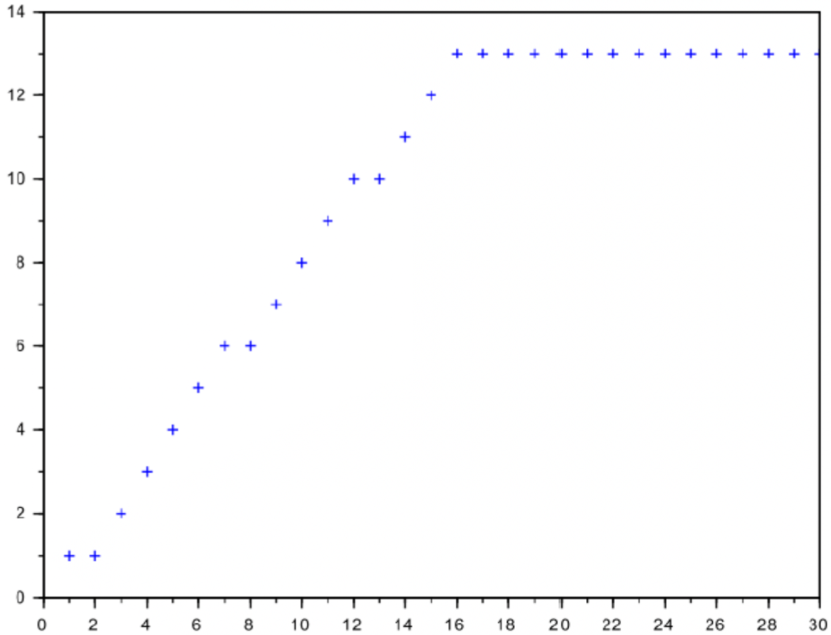
les deux termes de l'encadrement tendant vers π quand n tend vers l'infini.

2. Compléter la fonction python suivante afin qu'elle retourne, à l'aide des relations (*) et (**), une approximation$x$ deπ à e près, ainsi que le nombre k d'itérations qui ont été nécessaires.

```
def h(e):
    k = 0
    a =sqrt(3)/2
    b = 1/2
    while ......................:
        a = .....................
        b = .....................
        k = .....................
   x =.........................
   return x,k
```

3. On souhaite étudier l'évolution du nombre d'itérations nécessaires en fonction de la précision souhaitée. Ecrire une fonction python qui prend comme paramètre d'entrée un entier p et qui retourne une liste de taille p qui contient les nombres d'itérations nécessaires pour les précisions 10^{-k}, pour $k \in \{1, 2, ..., p\}$.

4. On utilise la fonction précédente avec $p = 30$ et on représente graphiquement les valeurs obtenues. On obtient le graphe suivant :



Commenter ce graphe.