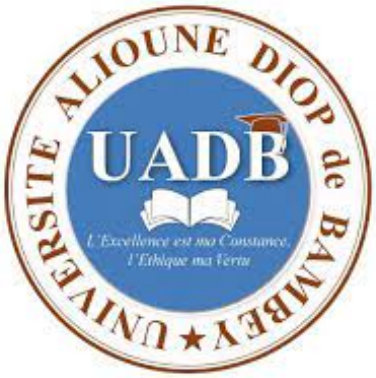

Calculatrice Scientifique

Examen de Python

Année universitaire 2023 - 2024

M. Abdoulaye Déthié Sarr

L'objectif de ce projet est de créer une **calculatrice scientifique** en utilisant les concepts de base du langage de programmation python. La calculatrice devrait offrir un large éventail de fonctionnalités :

- **arithmétiques** : addition/soustraction, modulo, division, muliplication ;
- **mathématiques** : puissance, logarithme, exponentiel, factoriel, racine carrée.

Le programme affichera le menu suivant :

1. Addition/Soustraction
2. Modulo
3. Multiplication
4. Division
5. Puissance
6. Factoriel
7. Logarithme Népérien
8. Exponentiel
9. Racine Carrée
0. Quitter

Ledit programme doit permettre à un utilisateur de choisir parmi plusieurs options en entrant un numéro entier correspondant à l'option souhaitée. Il exécute ensuite un sous-programme relatif à l'option sélectionnée. Après chaque exécution, le menu doit être réaffiché pour permettre à l'utilisateur de faire un autre choix jusqu'à ce qu'il décide de quitter le programme.

Il est important de vérifier la saisie de l'utilisateur pour chaque option choisie, afin de s'assurer que l'entrée est valide. Dans le cas échéant, le programme doit afficher un message d'erreur puis, à nouveau, demander à l'utilisateur de saisir une option valide.

Le processus de conception des sous-programmes (options) doit respecter les consignes suivantes :

1 Addition/Soustraction

La soustraction est une conséquence de l'addition : $2-1=2+(-1)$
Ainsi, cette option doit permettre à l'utilisateur d'additionner deux ou plusieurs nombres **réels** saisis.

- Saisir le nombre de termes (au moins 2)
- A l'aide d'une boucle, saisir les termes en précisant leur signe puis les additionner
- Afficher le résultat

2 Modulo

Cette option doit permettre à l'utilisateur de calculer le reste d'une division d'un nombre **entier**, saisi, par un autre.

- Saisir le dividende et le diviseur
- Si le diviseur est nul alors un message d'erreur doit être affiché

$$\textbf{dividende} = \textbf{quotient} \times \textbf{diviseur} + \textbf{reste}$$

- Calculer le reste de la division, si définie, en se basant sur la relation ci-dessus
- Afficher le résultat

NB : Il est interdit d'utiliser l'opérateur **modulo (%)** dans cette option.

3 Multiplication

Cette option doit permettre à l'utilisateur de multiplier deux ou plusieurs nombres **réels** saisis.

- Saisir le nombre de facteurs (au moins 2)
- A l'aide d'une boucle, saisir les facteurs en précisant leur signe puis les multiplier
- Afficher le résultat

4 Division

Cette option doit permettre à l'utilisateur de diviser un nombre **réel**, saisi, par un autre.

- Saisir le dividende et le diviseur
- Si le diviseur est nul alors un message d'erreur doit être affiché
- Calculer le résultat de la division, si définie, puis l'afficher

5 Puissance

Cette option doit permettre à l'utilisateur de calculer la puissance d'un nombre **réel** saisi.

- Saisir la base (nombre réel à multiplier par lui-même) et l'exposant (nombre de fois que la base est multipliée par elle-même)
- Calculer le résultat de la puissance, si définie, puis l'afficher en tenant compte des proprié-

tés suivantes :

$$a \in \mathbb{R}\ ,\ n \in \mathbb{N}\ ,\ a^n = \begin{cases} a \times a \times \cdots \times a \text{ (n fois)} & \text{si } n \geq 1 \\ 1 & \text{si } a \neq 0 \text{ et } n = 0 \\ impossible & \text{si } a = 0 \text{ et } n = 0 \end{cases}$$

- Si l'exposant est strictement négatif alors considérer la propriété suivante :

$$a \in \mathbb{R}\ ,\ n \in \mathbb{N}^*\ ,\ a^{-n} = \frac{1}{a^n} = \left(\frac{1}{a}\right)^n = \frac{1}{a} \times \frac{1}{a} \times \cdots \times \frac{1}{a} \text{ (n fois) si } a \neq 0$$

NB : Il est interdit d'utiliser l'opérateur **puissance (**)** dans cette option.

6 Factoriel

Cette option doit permettre à l'utilisateur de calculer le factoriel d'un nombre **entier** saisi.

- Saisir un nombre entier
- Si l'entier saisi est négatif alors un message d'erreur doit être renvoyé et l'utilisateur doit recommencer
- Calculer le résultat du factoriel, si définie, puis l'afficher.

$$\forall n \in \mathbb{N}\ ,\ n! = \begin{cases} n \times (n-1) \times (n-2) \times \cdots \times 1 & \text{si } n \neq 0 \\ 1 & \text{sinon} \end{cases}$$

NB : Il est interdit d'utiliser la fonction **factorial** du module **math** dans cette option.

7 Logarithme Népérien

Cette option doit permettre à l'utilisateur de calculer le logarithme népérien (ln) d'un nombre **réel** saisi.

Le logarithme népérien (aussi appelé logarithme naturel) de n'importe quel nombre réel **positif** peut être calculé à l'aide de diverses techniques. Une des méthodes les plus courantes est l'utilisation de la série de Taylor pour $ln(x)$ autour de 1. La série de Taylor pour $ln(x)$, infinie c'est à dire que les termes continuent sans fin, est donnée par :

$$ln(x) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{(x-1)^k}{k} = (x-1) - \frac{(x-1)^2}{2} + \frac{(x-1)^3}{3} - \frac{(x-1)^4}{4} + \cdots$$

C'est une série alternante, qui converge vers $ln(x)$ pour $0 < x \leq 2$ (pour $x \in \,]0,2\,]$) .

- Saisir un nombre réel x
- Si $x \leq 0$ alors un message d'erreur doit être renvoyé et l'utilisateur doit recommencer
- Si $0 < x \leq 2$ alors utiliser la formule de taylor définie précédement pour calculer et afficher $ln(x)$
- Si $x > 2$, vous pouvez utiliser le fait que $ln(ab) = ln(a) + ln(b)$ pour réduire le calcul du logarithme d'un grand nombre à celui de petits nombres. Par exemple, vous pouvez répéter la division de x par 2 jusqu'à obtenir un nombre entre 1 et 2, calculer le logarithme de ce nombre, puis ajouter le nombre de divisions par 2 multiplié par $ln(2)$.

Par exemple pour calculer $ln(10)$, sachant que 10 n'appartient pas à l'intervalle $]0,2]$, nous suivons les étapes suivantes :

a. Division répétée par 2 :

Nous divisons 10 par 2 jusqu'à obtenir un nombre qui se situe dans l'intervalle $]0,2]$. Ceci est fait pour nous permettre d'utiliser la série de Taylor.

$$10 \div 2 = 5$$

$$5 \div 2 = 2.5$$

$$2.5 \div 2 = 1.25$$

Nous avons donc divisé par 2 **trois fois** pour obtenir le nombre 1.25 qui est dans l'intervalle $]0,2]$.

b. Utilisation de la série de Taylor :

Nous utilisons ensuite la série de Taylor pour calculer $ln(1.25)$ et $ln(2)$.

$$ln(1.25) \approx \sum_{k=1}^{\infty} (-1)^{k+1} \frac{(1.25-1)^k}{k} \approx (1.25-1) - \frac{(1.25-1)^2}{2} + \frac{(1.25-1)^3}{3} - \frac{(1.25-1)^4}{4} + \cdots$$

$$ln(2) \approx \sum_{k=1}^{\infty} (-1)^{k+1} \frac{(2-1)^k}{k} \approx (2-1) - \frac{(2-1)^2}{2} + \frac{(2-1)^3}{3} - \frac{(2-1)^4}{4} + \cdots$$

Dans la pratique, vous ne pouvez pas calculer une série infinie, donc vous arrêtez généralement après un certain nombre de termes. Plus vous prenez de termes, plus l'approximation sera précise, mais le calcul sera également plus lent (**choisir un nombre de termes égal à 10**).

c. Addition des résultats :

Enfin, nous additionnons le résultat obtenu avec 3 fois $ln(2)$, car nous avons divisé par 2 **trois fois** lors de la première étape.

$$ln(10) = ln(1.25) + 3ln(2)$$

NB : Il est interdit d'utiliser, dans cette option, une fonction **native** ou définie dans quelconque **module**, qui calcule le logarithme népérien d'un nombre réel strictement positif donné.

8 Exponentiel

Cette option doit permettre à l'utilisateur de calculer l'exponentiel d'un nombre **réel** saisi.

La fonction exponentielle, notée $exp(x)$ ou e^x, peut également être exprimée par une série de Taylor. La série de Taylor pour e^x autour de 0 est la suivante :

$$exp\,(x) = \sum_{k=0}^{\infty} \frac{x^k}{k!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots$$

- Saisir un nombre réel x
- Calculer puis afficher l'exponentiel de x en utilisant la formule de Taylor ci-dessus (**choisir un nombre de termes égal à 10**).

La série de Taylor pour $exp(x)$, aussi connue sous le nom de série de MacLaurin, converge pour toutes les valeurs réelles de x. C'est-à-dire, peu importe la valeur que vous choisissez pour x, la série de Taylor pour $exp(x)$ convergera toujours vers $exp(x)$.

Ceci est une différence importante par rapport à la série de Taylor pour $ln(x)$, qui ne converge que pour les valeurs de x dans l'intervalle $]0, 2]$.

Par exemple pour calculer $exp(8)$ à l'aide de la série de Taylor, nous devons substituer $x = 8$ dans la série.

Comme ladite série est infinie, nous ne pouvons pas calculer tous les termes. Au lieu de cela, nous devons nous arrêter après un certain nombre de termes. Plus nous incluons de termes, plus notre approximation sera précise.

Si nous substituons $x = 8$ et arrêtons après 10 termes, nous obtenons :

$$exp\,(8) = \sum_{k=0}^{\infty} \frac{8^k}{k!} \approx 1 + 8 + \frac{8^2}{2!} + \frac{8^3}{3!} + \frac{8^4}{4!} + \frac{8^5}{5!} + \frac{8^6}{6!} + \frac{8^7}{7!} + \frac{8^8}{8!} + \frac{8^9}{9!}$$

NB : Il est interdit d'utiliser, dans cette option, une fonction **native** ou définie dans quelconque

module, qui calcule l'exponentiel d'un nombre réel donné.

9 Racine Carrée

Cette option doit permettre à l'utilisateur de calculer la racine carrée d'un nombre **réel** positif ou nul saisi.

La méthode de Newton-Raphson est une technique efficace pour trouver les racines d'une fonction. Pour calculer la racine carrée d'un nombre réel K à l'aide de cette méthode, nous pouvons considérer le problème comme la recherche d'une racine de la fonction suivante :

$$f(x) = x^2 - \mathrm{K}$$

L'algorithme de Newton-Raphson utilise une estimation initiale x pour la racine et itère sur la formule suivante jusqu'à ce que la convergence soit atteinte à une précision souhaitée :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Pour notre fonction $f(x) = x^2 - \mathrm{K}$, la dérivée est $f'(x) = 2x$. En substituant dans l'équation ci-dessus, nous obtenons :

$$x_{n+1} = x_n - \frac{x_n^2 - \mathrm{K}}{2x_n} = \frac{1}{2}\left(x_n + \frac{\mathrm{K}}{x_n}\right)$$

- Saisir un nombre réel K
- Si $\mathrm{K} < 0$ alors un message d'erreur doit être afficher et l'utilisateur doit recommencer
- Si $\mathrm{K} \geq 0$, calculer puis afficher la racine carrée de K en utilisant l'algorithme itératif de Newton-Raphson.

Prenons un exemple spécifique : calculons la racine carrée de $\mathrm{K} = 2$. Supposons que notre estimation initiale soit $x_0 = 1$. Ensuite, nous itérons comme suit :

$$x_1 = \frac{1}{2}\left(x_0 + \frac{2}{x_0}\right) = \frac{1}{2}\left(1 + \frac{2}{1}\right) = 1.5$$

$$x_2 = \frac{1}{2}\left(x_1 + \frac{2}{x_1}\right) = \frac{1}{2}\left(1.5 + \frac{2}{1.5}\right) \approx 1.4167$$

$$x_3 = \frac{1}{2}\left(x_2 + \frac{2}{x_2}\right) = \frac{1}{2}\left(1.4167 + \frac{2}{1.4167}\right) \approx 1.4142$$

Et ainsi de suite, jusqu'à ce que nous atteignions le niveau de précision souhaité. Dans ce cas, nous constatons que la méthode converge rapidement vers la valeur correcte de la racine carrée de 2, qui est environ 1.4142.

La précision dans la méthode de Newton-Raphson est généralement déterminée par un critère

d'arrêt basé sur la différence entre deux estimations consécutives. Si cette différence est inférieure à une certaine tolérance, alors nous pouvons considérer que nous avons atteint le niveau de précision souhaité et nous arrêter.

Mathématiquement, cela peut être exprimé comme suit. Soit ϵ la tolérance (par exemple, $\epsilon = 0.00001$ pour une précision de 5 chiffres après la virgule). Alors nous continuons l'itération tant que :

$$|x_{n+1} - x_n| \geq \epsilon$$

Autrement dit, tant que la différence absolue entre deux estimations consécutives est supérieure ou égale à ϵ nous continuons à itérer. Lorsque cette différence tombe en dessous de ϵ , nous arrêtons l'itération et nous considérons x_{n+1} comme notre approximation finale.

Pour la racine carrée, spécifiquement, un choix courant pour l'estimation initiale est la moitié du nombre dont vous voulez trouver la racine carrée. Par exemple, si vous voulez trouver la racine carrée de $K \neq 0$, vous pourriez choisir $x_0 = \frac{K}{2}$ comme estimation initiale. C'est généralement un choix raisonnable qui permet une convergence relativement rapide.
Et, si $K = 0$, vous pourriez choisir $x_0 = \frac{1}{2} = 0.5$ comme estimation initiale.

NB : Il est interdit d'utiliser, dans cette option, une fonction **native** ou définie dans quelconque **module**, qui calcule la racine carrée d'un nombre réel positif ou nul donné.

10 Quitter

Si l'utilisateur choisit cette option alors le programme doit afficher « **Au revoir !** » puis s'arrêter.

NB : L'esthétique du programme sera tenu en compte à la correction.

1. Faire un controle de saisie pour chaque entrée de l'utilisateur.
2. Travailler par groupe d'au maximum 2 étudiants.
3. Préciser dans le fichier que vous devez rendre le prénom et le nom des différents membres.
4. Envoyer le projet avant la date limite à l'adresse mail suivant : abdoulayedethie.sarr@uadb.edu.sn en mettant comme objet : **Examen Python M1 CRD 2023-2024**

Dateline : Dimanche 10 mars 2024 à 23h59.

GOOD LUCK ! ! !