PSL **CPES 1re année**
Informatique **2023-2024**

TP3 : Exercices sur les instructions répétitives

Exercice de Debugage sur équation du second degré

Télécharger le fichier `Debug_Equation2ndDegre.py` disponible sous moodle. Ce fichier contient un programme Python permettant de résoudre l'exercice sur la résolution d'une équation du second degré. Malheureusement le programme ne fonctionne pas ! Vous allez devoir le corriger et le faire fonctionner.

1. A quoi servent les lignes commençant par # ? Vous aident-elles à comprendre ce que fait ce programme ?
2. Exécuter le programme ci-dessus tel quel. Quel est le message d'erreur ? Que faut-il corriger dans le programme pour résoudre le problème ?
3. L'instruction `print("la valeur de delta est",d)` est-elle nécessaire dans le programme ? Si non, pourquoi l'ajouter ?
4. Exécuter le programme en donnant 2 pour a, -8 pour b et 6 pour c. Quel est le message d'erreur ? Que faut-il corriger dans le programme pour lever ce message ?
5. Exécuter le programme en donnant 2 pour a, -8 pour b et 6 pour c. Le programme s'exécute-t-il ? Comment s'assurer que la solution proposée est la bonne ? Si ce n'est pas la bonne que faut-il corriger ? Où dans le programme ? Faire les corrections correspondantes.
6. Exécuter le programme en donnant 2 pour a, 4 pour b et 4 pour c. Quel est le message d'erreur ? Que faut-il corriger dans le programme pour résoudre le problème ?
7. Le cas où l'équation admet une unique solution réelle a-t-il été testé ? Proposer un jeu de valeur permettant de l'exécuter.
8. Exécuter le programme en donnant 0 pour a, 2 pour b et 6 pour c. Quel est le message d'erreur ? Que faut-il corriger dans le programme pour résoudre le problème ?

Exercice 1

Télécharger le programme `demain.py`, disponible sous moodle, qui corrige l'exercice 4 du TP2. Ce programme :

— demande à l'utilisateur l'année courante,
— demande à l'utilisateur le mois courant,
— demande à l'utilisateur le jour courant,
— et affiche "Demain, nous serons le " suivi du numéro du jour.

Modifier ce programme pour qu'il vérifie la validité des données saisies au clavier par l'utilisateur :

— le mois doit être compris entre 1 et 12,

— le jour doit être compris entre 1 et 31 si le mois est impair et inférieur ou égal à 7 ou si le mois est pair et supérieur ou égale à 8
— le jour doit être compris entre 1 et 28 pour le mois de février, sauf les années bissextiles pendant lesquelles il varie de 1 à 29 jours,
— le jour doit être compris entre 1 et 30 dans les autres cas.

Demandez à l'utilisateur de saisir de nouvelles valeurs en cas d'erreur jusqu'à ce que les valeurs soient correctes.

Exercice 2

Étant donnée $f(x) = x^3 - 3x^2 + 1$, une fonction continue et strictement croissante sur l'intervalle $[2; 3]$ (avec $f(2) = -3$ et $f(3) = 1$), on cherche à déterminer la valeur approchée de x telle que $f(x) = 0$ par une méthode dichotomique avec une précision ϵ. La méthode dichotomique de résolution s'énonce comme suit :

— on calcule m le milieu de $[a; b]$, si $f(m) = 0$ alors on a trouvé la solution ;
— si $f(m)$ et $f(b)$ sont de même signe, c'est que la solution se trouve dans $[a; m]$: on affecte à b la valeur de m afin de pouvoir continuer le processus ;
— dans le cas contraire, la solution se trouve dans $[m; b]$: on affecte à a la valeur de m afin de pouvoir continuer le processus ;
— on recommence le processus jusqu'à obtenir un encadrement de m suffisamment petit, c'est-à-dire $|b - a| \leq \epsilon$.

Programmer en Python cet algorithme.

Exercice 3

Écrire un programme en Python qui saisit un réel $x \geq 0$ et un entier $n \geq 0$, et affiche la valeur de x^n (sans utiliser l'opérateur puissance). Tester votre programme pour les valeurs suivantes : $x = 4$ $n = 3, x = 5$ $n = 0, x = 0$ $n = 0$.

Exercice 4

Soit pour tout $n \in \mathbb{N}^*$, $S_n = \sum_{k=1}^{n} \frac{1}{k}$, écrire un programme déterminant la plus petite valeur de n pour laquelle $S_n > A$, A étant un réel entré par l'utilisateur. N'essayez pas votre programme avec des valeurs de A supérieures à 20.

Exercice 5

Soit la suite définie par $u_0 = 0$, et pour tout $n \in \mathbb{N}$, $u_{n+1} = \sqrt{3u_n + 4}$.

1. Écrire un programme demandant à l'utilisateur un entier n et affichant tous les termes de la suite jusqu'à u_n.
2. Modifier votre programme pour qu'il affiche le plus petit entier n pour lequel $u_n > 4 - \epsilon$, où $\epsilon > 0$. Que trouve-t-on pour $\epsilon = 10^{-8} = 1.e - 8$?

Exercice 6

Écrire un programme python, qui saisit un entier (supposé strictement positif), affiche la somme de tous ses diviseurs stricts et précise si ce nombre est premier. Par exemple, si l'on saisit 8, il affiche 7 (car somme de ses diviseurs stricts qui sont 1, 2 et 4).

Exercice 7

Un entier est dit parfait s'il est égal à la somme de ses diviseurs stricts (6 par exemple est parfait). Écrire un programme python qui saisit un entier n (supposé strictement positif) et affiche tous les nombres parfaits inférieurs ou égaux à n.

Exercice 8

Le Plus Petit Commun Multiple (PPCM) de deux entiers positifs a et b est le plus petit entier p tel que p soit à la fois multiple de a et de b.

Afin de calculer le PPCM de a et b, on applique l'algorithme suivant qui consiste à calculer les multiples successifs de a et b, notés m_a et m_b, jusqu'à arriver à l'obtention du PPCM.

(i) initialiser m_a et m_b ;

(ii) tant que le PPCM n'est pas trouvé, on augmente la plus petite des deux valeurs m_a et m_b ;

(iii) afficher le PPCM.

1. Comment initialiser m_a et m_b ?
2. Comment identifier que le PPCM est trouvé ?
3. De combien doit-on augmenter à chaque itération m_a ? et m_b ?
4. Écrire en Python le programme complet : il doit lire deux nombres saisis par l'utilisateur (on les supposera entiers et positifs), calculer leur PPCM et l'afficher.

Exercice 9

A l'aide de 2 boucles `TANT_QUE` emboîtées, écrire un programme Python qui affiche :

```
1   2   3   4   5
1   3   5   7   9
1   4   7   10  13
1   5   9   13  17
```

Exercice 10

Écrire un programme python qui saisit un entier strictement positif n, puis n entiers, et qui vérifie si la suite des n entiers saisis est triée de façon croissante.
Par exemple, si la valeur saisie pour n est 7 et les 7 valeurs saisies sont : -1, 3, 7, 8, 11, 234, 300, le programme affichera "les 7 valeurs sont triées de façon croissante", alors que le programme affichera "les 7 valeurs ne sont pas triées" si les entiers saisis sont : 1, 2, 7, -2, 4, 5, 200.

Exercice 11

Écrire un programme python qui saisit un entier strictement positif n, puis n entiers strictement positifs, et qui vérifie si parmi les n entiers saisis, la somme des entiers pairs est égale à la somme des entiers impairs. Le programme doit refuser les valeurs négatives ou nulles.
Par exemple, si la valeur saisie pour n est 5 et les 5 valeurs saisies sont 2, 3, 2, 3, 2, le programme affichera "la somme des nombres pairs est égale à la somme des nombre impairs", alors que le programme affichera "la somme des nombres pairs n'est pas égale à la somme des nombres impairs" si les valeurs saisies sont 2, 3, 4, 3, 1. Si la valeur saisie pour n (ou une des n valeurs) est par exemple -2, le programme demandera à nouveau d'insérer une valeur, jusqu'à ce que celle-ci soit strictement positive.

Exercice 12

On joue à lancer 3 dés : un dé rouge à 6 faces (numérotées de 1 à 6), un dé vert à 8 faces (numérotées de 1 à 8) et un dé bleu à 10 faces (numérotées de 1 à 10). On veut déterminer combien il y a de façons d'obtenir un nombre n donné en additionnant les trois faces des dés. Par exemple, il y a une seule façon d'obtenir 3 qui est de faire 1 avec le dé vert, 1 avec le dé rouge et 1 avec le dé bleu. Il y a 3 façons d'obtenir 4 :

— 1 avec le dé vert, 1 avec le dé rouge et 2 avec le dé bleu, ou bien,
— 1 avec le dé vert, 2 avec le dé rouge et 1 avec le dé bleu, ou bien,
— 2 avec le dé vert, 1 avec le dé rouge et 1 avec le dé bleu.

Écrire le programme **Python** qui implémente l'algorithme suivant :

1. demander à l'utilisateur de saisir une valeur entière ;
2. à l'aide de boucles imbriquées, compter le nombre de possibilités permettant d'obtenir ce nombre comme somme des dés rouge, vert et bleu ;
3. afficher le résultat.

Exemple d'exécution :

affichage Entrez un nombre entier :

saisie 5

affichage Il y a 6 façon(s) de faire 5 avec ces trois dés.