Informatique en CPGE (2018-2019)
TP 3 : algorithmes de base

Il est recommandé d'écrire un algorithme sur papier avant d'écrire un programme. Lors de l'écriture d'un programme il faut le tester régulièrement, après chaque ligne ou chaque bloc d'instructions, afin de corriger les erreurs éventuelles sans attendre la fin. L'utilisation de la fonction print() placée en différents endroits du programme permet de vérifier que les valeurs des variables sont bien celles attendues.

1 Boucles itératives conditionnelles

Exercice 1 : les suites

1. Rang à partir duquel u_n est supérieur à un réel A

   On considère l'algorithme suivant où S a la valeur 2000 et N la valeur 0 :

```
Tant que S < 2500
    Remplacer S par S*1,025
    Augmenter N de 1
Fin du Tant Que
```

   (a) Ecrire la suite des cinq premières valeurs de N et de S.

   (b) Déterminer les valeurs finales de N et de S.

   (c) Ecrire le programme et l'exécuter afin de vérifier les réponses aux questions 1 et 2.

2. Suites arithmético-géométriques

   On considère la suite (u_n) définie par $u_1 = \dfrac{3}{4}$ et $u_{n+1} = \dfrac{1}{2}u_n + \dfrac{1}{4}$.

   On démontre que la suite (u_n) est décroissante et minorée, donc convergente et de limite $\dfrac{1}{2}$.

   Expliquer ce que fait l'algorithme suivant puis écrire le programme et le tester. On suppose que u a pour valeur initiale 0,75 et n la valeur 1

```
Tant que u-0,5 > 0,0001
    Remplacer u par 0,5*u+0.25
    Augmenter n de 1
Fin du Tant Que
```

Exercice 2

1. Tester le programme suivant et expliquer son fonctionnement :

```
# racine cubique (cas entier positif)

n=int(input("entrer un nombre naturel"))
p=n%2

while p**3<n:
    p=p+2
if p**3!=n:
    print(n,"n'est pas cube parfait")
else:
    print('la racine cubique de',n,'est',p)
```

2. Ecrire un programme, sur le même modèle que le précédent, qui trouve et affiche la racine cubique d'un **entier relatif** s'il est un cube parfait et sinon affiche le message "n n'est pas un cube parfait".

Exercice 3

Etudier et commenter ce programme (on dit que la saisie est "filtrée") :

```
n=int(input('Entrer un  nombre entier strictement positif : '))
while not(n>0):
    n=int(input('Attention, entrer un entier strictement positif : '))
x=n//2
while x>1:
    if n%x==0:
        print(n,'a pour facteur',x)
        break # interruption
    x-=1
if x==1:
    print(n,'est premier')
```

2 Boucles itératives

Exercice 4

Etudier et commenter ce programme :

```
for n in range(1,256):
    nb,compteur=n,0
    while n>0:
        n//=2
        compteur+=1
    print(nb,"s'écit en binaire avec",compteur,"chiffres")
```

Exercice 5 : moyenne et variance

1. Ecrire un programme qui demande à l'utilisateur d'entrer des nombres, (le programme doit demander au préalable combien de nombres l'utilisateur souhaite entrer), puis calcule la moyenne et la variance de ces nombres. Le programme affichera la moyenne et la variance arrondies à 10^{-3} près ; pour cela, on utilisera l'instruction `round(x,3)` qui arrondit la valeur d'une variable x à 10^{-3} près.
2. Modifier le programme précédent pour afficher une moyenne arrondie au point entier.
3. Modifier le programme précédent pour afficher une moyenne arrondie au demi-point.

Exercice 6

Ecrire un programme qui permet d'afficher un tableau de valeurs sur l'intervalle $[1;3]$ en utilisant un pas de 0,1, pour la fonction f définie par $f(x) = \sin x + \dfrac{1}{2}\ln x$.

Exercice 7

On demande à un joueur de deviner en sept essais au maximum un nombre tiré au hasard entre 1 et 100. On lui indique à chaque fois si le nombre qu'il propose est supérieur ou inférieur au nombre cherché. Sans stratégie, il peut être long pour parvenir à touver ce nombre.

Ecrire un programme qui permet de jouer à ce jeu ; pour choisir un nombre entier au hasard entre 1 et 100, on utilise la fonction `randint()` du le module `random` :

```
>>> from random import randint
>>> a=randint(1,100)
```

Complément : écrire un programme qui joue lui-même au jeu en utilisant une stratégie basée sur la dichotomie et étudier le nombre moyen de coups joués pour gagner.

3 Nombres complexes

Exercice 8 : équation du second degré

1. On définit un polynôme du second degré P par $P(x) = ax^2 + bx + c$ où a, b et c sont des nombres réels. Ecrire un programme qui demande à l'utilisateur d'entrer les valeurs des nombres a, b et c, puis résout dans $\mathbb{R}$ l'équation $P(x) = 0$ et affiche en sortie la (ou les) solution(s).
2. Reprendre le programme et le modifier afin de traiter la résolution dans $\mathbb{C}$.
3. Reprendre le programme et le modifier afin de traiter la résolution dans $\mathbb{C}$ dans le cas où a, b et c sont des nombres complexes.