IPT DS2

Lundi 8 Janvier – Durée 1h

1. Retour sur les bases (7 pts)

1) Soit T=[1,2,3,4]
   a) Que donne T[0]==2 ?
   b) Que donne T[0]=2 ?
   c) Que donne print(T[2]) ?
   d) Que donne print(T[1:]) ?
   e) Que donne type(T) ?

2) Exprimer les nombres suivants en base 10.
   a) $(11010)_2$
   b) $(D0)_{16}$

3) Ecrire une fonction qui prend en entrée une liste, et remplace son dernier élément par l'entier naturel 0.

4) Triplets pythagoriciens.
   a) Soit n un entier fixé. Écrire un code permettant d'afficher à l'écran tous les triplets pythagoriciens (triplets d'entiers (a ; b ; c) tels que $a^2 + b^2 = c^2$), tels que $1 \leq a \leq b \leq c \leq n$.
   b) Préciser la complexité temporelle de votre algorithme par rapport à n.

2. Algorithmique (13 pts)

5) Factorielle

Rappel : la factorielle d'un nombre entier naturel n est définie par :

$$n! = \prod_{i=1}^{n} i = 1 \times 2 \times 3 \times \ldots \times (n-1) \times n$$

Écrire une fonction `factorielle` qui prend pour argument un entier naturel n, et qui retourne la valeur de sa factorielle. Préciser la complexité temporelle.
Dans le cas où n n'est pas un entier naturel, la fonction devra retourner le booléen `False`.

6) Suite récurrente linéaire 1

Soit la suite U_{n+1} telle que : $U_{n+1} = \frac{1}{3}U_n - 2$ avec $U_0 = 4$

   a) Écrire la fonction `suite(n)` qui prend en argument un entier n et qui retourne la liste des valeurs de la suite de 0 à U_n inclus.
   b) On peut montrer que la suite converge, on va donc essayer de calculer les termes de la suite jusqu'à ce qu'on considère qu'elle n'évolue plus. Proposer un autre code,

impliquant un critère d'arrêt : on calcule les valeurs de la suite jusqu'à ce que la différence U_{n+1}-U_n, en valeur absolue, soit inférieure ou égale à 10^{-10}. Alors, le code s'arrête, affiche ''la suite n'évolue plus'' et renvoie la valeur U_n ainsi que le nombre d'itérations.

7) Approximation des solutions d'une équation.

On recherche une approximation d'une solution de l'équation $f(x) = 0$ sur l'intervalle [a,b] avec f une fonction connue. Pour simplifier la situation, on suppose que f est continue et strictement croissante, avec $f(a) < 0$ et $f(b) > 0$.

a) Justifier qu'il existe une unique solution sur [a,b] pour l'équation $f(x) = 0$. On la note c.

b) On rappelle la méthode de la dichotomie pour approcher c, que l'on décrit en termes d'algorithme : on considère deux variables, $x1$ et $x2$, initialisées en a et b. On calcule $f(\frac{x1+x2}{2})$, tant que cette valeur est supérieure en valeur absolue à un nombre fixé eps:

- Si $f(\frac{x1+x2}{2})$ <0, alors la variable $x1$ est remplacée par $\frac{x1+x2}{2}$
- Sinon, c'est $x2$ qui est remplacée par $\frac{x1+x2}{2}$.

Lorsque $\left|f(\frac{x1+x2}{2})\right| < eps$ (ce qui arrive au bout d'un moment d'après le cours de maths), l'algorithme s'arrête et on renvoie la dernière valeur de $\frac{x1+x2}{2}$.

Ecrire la fonction `dichotomie(f,a,b,eps)` qui prend en argument une fonction f et qui renvoie une solution de l'équation $f(x) = 0$ sur [a,b] à epsilon près.

c) On ne suppose plus que f est strictement croissante, mais seulement que $f(a)$ et $f(b)$ sont de signes opposés, ce qui peut se traduire par $f(a)f(b) < 0$. Proposer un nouvel algorithme de dichotomie qui tienne compte de cela, et une nouvelle fonction `dichotomie2(f,a,b,eps)` adaptée. On essayera de faire un minimum de modifications par rapport à la fonction précédente plutôt que de tout réécrire .

8) Variant

On considère le programme suivant :

```
while p > 0 :
      if c == 0 :
            p = p-2
            c = 1
      else :
            p+ = 1
            c = 0
```

a) Partant de p=5 et c=0, donner les états successifs (c'est à dire les valeurs des variables globales) obtenus pendant l'exécution.

b) Partant de valeurs p,c entières et strictement positives, montrer que 2p + 3c est un variant de boucle strictement décroissant.

c) Que dire de la terminaison de ce programme ?