TP 1 informatique

BCPST 2 2021-2022

Révisions

V.Vong

Points abordés

— Opérateurs logiques, comparaisons.
— Structures conditionnelles.
— Boucles for et while.
— Fonctions.

Exercice 1. Soit $a \in \mathbb{N}$. On définit la suite $(a_n)_{n\in\mathbb{N}}$ par :

$$a_0 = a, \forall n \in \mathbb{N}, a_{n+1} = \begin{cases} \sqrt{a_n} & \text{si} \quad \exists p \in \mathbb{N}, p^2 = a_n \\ a_n + 3 & \text{sinon} \end{cases}$$

1. Écrire une fonction `suite(a,n)` qui retourne la valeur de a_n.
2. Écrire une fonction `suiteL(a,n)` qui retourne la liste $[a_0, a_1, \cdots, a_n]$. Attention, on veillera à ce que chaque terme de la suite ne soit calculé qu'une seule fois.
3. Écrire une fonction `boucle(a,N)` qui renvoie `True` s'il existe $(p, q) \in \{0, \cdots, N\}^2, p \neq q$ tel que $a_p = a_q$ et `False` sinon.
4. Écrire une fonction `premierCarreP(a)` qui détermine le premier indice n tel que a_n est le carré d'un entier.

Exercice 2. On considère le pseudo-code suivant :

```
fonctionmystere(x) :
    P=1
    n=1
    tant que P<x faire
         n=n+1
         P=P*n
    retourner n
```

1. Quelle est la valeur de `fonctionmystere(43)` ?
2. À quoi correspondent les valeurs P et n ?
3. Écrire en Python une fonction correspondant au pseudo-code précédent.

Exercice 3.

1. Écrire une fonction `matrice_aleatoire(n)` qui retourne une matrice de taille $n \times n$ où les coefficients de la matrice sont des entiers compris entre 0 et 6 choisis au hasard.
2. Écrire une fonction `occurrence(M)` qui prend en argument une matrice carrée et qui retourne une liste L où $L[i]$ est le nombre de i dans la matrice M. On restreindra au cas où i est compris entre 0 et 6.
3. Écrire une fonction `somme(M)` qui prend en argument une matrice M et qui retourne la somme de tous les coefficients de la matrice.

Exercice 4. On considère le programme suivant :

```
a=1
def toto() :
    global a
    a=49
toto()
```

1. Déterminer la valeur de a à l'issue du programme.
2. Écrire un programme qui modifie la variable $L = [1, 2, 3]$ en inversant 1 et 2.

Remarque : en général, on évite d'utiliser les variables globales. Mais celles-ci peuvent être pratique dans certains cas.

Exercice 5. Le jeu du FizzBuzz consiste à énumérer les entiers de 1 à n en prononçant "fizz" si le nombre est un multiple de 3, "buzz" si le nombre est un multiple de 5. Dans le cas où le nombre est un multiple de 3 et 5 on prononce alors "fizzbuzz". Par exemple, pour les entiers de 1 à 3, on obtient : un, deux, fizz. Pour l'entier 5 on prononce "buzz" et pour l'entier 15, on prononce "fizzbuzz".

Écrire un fonction `fizzbuzz(n)` qui affiche tous les entiers de 1 à n en respectant les règles du fizzbuzz.

Exercice 6. On considère la suite définie par :

$$u_0 = 1, \forall n \in \mathbb{N}, u_{n+1} = \sum_{k=0}^{n} u_k u_{n-k}.$$

Écrire une fonction `suite(n)` qui prend en argument un entier n et qui renvoit la liste $[u_0, u_1, u_2, \ldots, u_{n-1}]$.

Exercice 7. Étant donné un sous-ensemble fini A de $\mathbb{N}$, on définit $mex(A)$ par le plus petit entier positif qui n'est pas un élément de A. Écrire une fonction `mex(A)` qui prend en argument une liste d'entiers A et qui renvoit $mex(A)$.

Exercice 8. Écrire une fonction `BienParenthesee(mot)` qui prend en arguement une chaîne de caractères composée uniquement de parenthèses ouvrantes `"("` et fermantes `")"` et qui retourne `True` si la chaîne est bien parenthésée et `False` sinon. On dit que `mot` est bien parehntésé s'il a exactement le même nombre de parenthèses ouvrantes et fermantes et si tout préfixe de `mot` a au moins autant de parenthèses ouvrantes que fermantes. Par exemple, "((()))" est bien parenthésé mais "(()))(" et "((())" ne le sont pas.

Exercice 9.

1. Écrire une fonction `SuiteCroissante(suite,n)` qui prend en arguments un entier naturel n et une fonction `suite` (qui correspond à une suite de réels) et qui retourne `True` si `suite` est croissante jusqu'au rang n et `False` sinon.
2. Justifier qu'il est difficile d'écrire un programme pour montrer qu'une suite réelle arbitraire est croissante.

Exercice 10. Écrire une fonction `Jackpot()` qui génère des entiers aléatoires entre 0 et 1000 jusqu'à tomber sur 777. On comptera le nombre d'entiers ainsi générés.