ISTA Aïn Mlila
MP

Examen Informatique S2

11 mai 2024

Questions de cours

1. (2½ points) Python représente les données à travers des types élémentaires. Citez les 5 vus en cours, en donnant un exemple pour chacun.

   Solution:

   - `int` ex. -14
   - `float` ex. 1.4
   - `complex` ex. 1-1j
   - `bool` ex. False
   - `str` ex. 'T' ou "A"

2. (4½ points) Les messages d'erreurs nous donnent des indications précieuses pour corriger rapidement les programmes. Python différencie plusieurs types d'erreurs :

   - `ZeroDivisionError`
   - `NameError`
   - `TypeError`
   - `IndentationError`
   - `SyntaxError`
   - `ValueError`

   Expliquez brièvement chaque type et donnez un exemple pour chacun.

   Solution:

   - `>>> 1/0 -> ZeroDivisionError`
   - `>>> a + 1 -> NameError`
   - `>>> 1 + "2" -> TypeError`
   - `>>>  1 + 2 -> IndentationError`
   - `>>> (1+2)) -> SyntaxError`
   - `>>> int("1.2") -> ValueError`

3. (1 point) Quelle est la fonction qui permet de connaître le nombre d'éléments d'une séquence (chaîne de caractères, liste, tuple, range...) ?

   Solution: La fonction `len()`

Exercice 1

1. (3 points) On définit les variables suivantes :

```
n = 10
pair = (7%2 != 0)
texte = "nombres premiers"
premiers = [2, 3, 5, 7, "deux", "trois", "cinq", "sept"]
```

   (a) Donnez le type Python de chacune des variables

   Solution:

   1. `n : int`
   2. `pair : bool`
   3. `texte : str`
   4. `premiers : list`

(b) Donnez la valeur des expressions suivantes :

1. `n == 2`
2. `pair`
3. `'les ' + texte + " sont :"`
4. `texte[1:3]`
5. `texte[:4]`
6. `premiers[2]*2`
7. `premiers[5]*2`
8. `premiers + ['onze']`

Solution:

1. `False`
2. `True`
3. `'les nombres premiers sont :'`
4. `'omb'`
5. `'nomb'`
6. `10`
7. `'troistrois'`
8. `[2,3,5,7,'deux','trois','cinq','sept','or`

2. (3 points) Soit la definition de la fonction suivante :

```
def foo (n) :
    res = 1
    while n > 1 :
        res = res * n
        n = n - 1
    return res
```

Calculez foo(2) ;foo(3), ensuite trouvez ce que fait la fonction foo(x)

Solution: foo(2) vaut 2 ; foo(3) vaut 6 ; la fonction foo(n) calcule le factoriel de n

Exercice 2

On considère les définitions suivantes

```
def f1(n):
   res = []
   for i in range(n):
      if i%2 != 0:
         res.append(i**2)
   return res
```

```
def f2(n):
   res = []
   for i in range(n):
      j = i**2
```

```
        if j%2 != 0:
            res.append(j)
    return res
```

1. (1½ points) Que fait la fonction f1(n) ; donnez un exemple.

> **Solution:** f1(5) donne [1,9]
> la fonction f1(n) donne la liste des carrés des nombre impairs de 0 jusqu'a n-1

2. (1½ points) Que fait la fonction f2(n) ; donnez un exemple.

> **Solution:** f2(5) donne [1,9]
> la fonction f2(n) donne la liste des nombres (de 0 à n-1) carrés impaire.

Exercice 3

1. (3 points) Définissez fonction `racinesEquation(a,b,c)` qui prends les coefficients `a, b, c` et qui renvoie la liste des racines réelles ou complexes d'un trinôme du second degré à coefficients réels, c'est à dire une expression de la forme $ax^2 + bx + c$, où a, b et c sont trois réels avec a non nul.

   Rappels :
   - Le discriminant de cette équation $delta = b^2 - 4ac$.
   - La liste des solutions est soit deux racines réels ou une racine réel double ou deux racines complexes.
   - Pour les racines réels : $\frac{-b \pm \sqrt{delta}}{2a}$
   - Pour les solutions complexes :
     si $x_1 = r_1 + i_1 j$ et $x_2 = r_2 + i_2 j$ sont les racines de l'équation alors :

$$\begin{aligned} r_1 \leftarrow r_2 &\leftarrow \frac{-b}{2a} \\ i_1 &\leftarrow \frac{\sqrt{-delta}}{2a} \\ i_2 &\leftarrow -i_1 \end{aligned}$$

   Indications :
   - Pour pouvoir utiliser la fonction racine carrée `sqrt` du module `math`, ne pas oublier d'ajouter l'instruction `from math import sqrt`.
   - Pour affecter des complexes on doit respecter la forme suivante : `(real + imag*1j)`

> **Solution:**
>
> ```
> from math import sqrt
> def racinesEquation(a,b,c):
>     res = []
> ```

```
    delta = b**2 - 4*a*c
    if delta <0:
        r1 = r2 = -b/2*a
        i1 = sqrt(-delta)/(2*a)
        i2=-i1
        x1,x2 = r1+i1*1j,r2+i2*1j
        res.append(x1)
        res.append(x2)
    elif delta==0:
        x=-b/(2*a)
        res.append(x)
    else :
        x1 = -b+sqrt(delta)/(2*a)
        x2 = -b-sqrt(delta)/(2*a)
        res.append(x1)
        res.append(x2)
    return res
```