Savoir parler à la machine

MPSI/MP Option informatique - TP n°3 - Olivier Reynet

Objectifs d'apprentissage :

- ☐ Développer et d'exécuter un code simple dans un IDE (Pyzo par exemple)
- ☐ Savoir invoquer un shell Python afin d'y tester simplement des instructions
- ☐ Connaître le concept de langage interprété et observer un bytecode généré (module Disassemble)
- ☐ Connaître et comprendre les concepts qui détermine la syntaxe de Python (indentation significative, blocs, scripts, variables globales, module, fonction, main)

Examen

Cours: MPSI/MP Option informatique

Date: 24-04-2022

Consignes :

1. Écrire de manière lisible et intelligible.
2. Préparer une réponse au brouillon avant de la reporter sur la feuille.
3. Répondre dans les espaces procurés à cet effet.
4. S'il ne reste plus de place, écrire au dos de la feuille.

Élève **Classe :** ____________________

Nom : ____________________

Prénom : ____________________

```python
def rec_dyadic_decomp(a, d=0, aux=0, p=0, r=None):
t = "String"
# commentaire
my_list = []
my_dict = {}
my_dict[key] = "Test"
for k in my_dict:
print(my_dict[k])
else:
if aux >= 1:
aux = aux - 1
return rec_dyadic_decomp(a, d, aux, p, r)
```

Algorithme 1 Illustration du pseudo-langage algorithmique

Fonction DYADIC_DECOMP(a) ▷ $a = 0, d_1 d_2 d_3 \dots d_n$
  $r \leftarrow$ une nouvelle liste
  $\alpha \leftarrow a$
  $p \leftarrow -1$
  $d \leftarrow 0$
  tant que $d < a$ **faire**
    $\alpha \leftarrow 2 \times \alpha$
    $b \leftarrow \lfloor \alpha \rfloor$
    ajouter b à r
    $p \leftarrow p - 1$
    $d \leftarrow d + b \times 2^p$
    si $\alpha \geq 1$ **alors**
      $\alpha \leftarrow \alpha - 1$
  retourner r ▷ la liste résultat

```python
def is_leap(y):
    return (y%4==0 and not y%100==0) or y%400==0
```

Test du inline python `return 3`

1. **Structures conditionnelles**
   (a) Que fait l'algorithme 2 ?

Algorithme 2 Illustration du pseudo-langage algorithmique

Fonction DYADIC_DECOMP(a) ▷ $a = 0, d_1 d_2 d_3 \dots d_n$
  $r \leftarrow$ une nouvelle liste
  $\alpha \leftarrow a$
  $p \leftarrow -1$
  $d \leftarrow 0$
  tant que $d < a$ **faire**
    $\alpha \leftarrow 2 \times \alpha$
    $b \leftarrow \lfloor \alpha \rfloor$
    ajouter b à r
    $p \leftarrow p - 1$
    $d \leftarrow d + b \times 2^p$
    si $\alpha \geq 1$ **alors**
      $\alpha \leftarrow \alpha - 1$
  retourner r ▷ la liste résultat

Solution: 1011110001 soit $101111,0001 = 2^5 + 2^3 + 2^2 + 2^1 + 2^0 + 2^{-4}$.

   (b) Que fait le programme suivant ?

```python
def rec_dyadic_decomp(a, d=0, aux=0, p=0, r=None):
  t = "String"
  # commentaire
  my_list = []
  my_dict = {}
  my_dict[key] = "Test"
  for k in my_dict:
        print(my_dict[k])
  else:
     if aux >= 1:
        aux = aux - 1
  return rec_dyadic_decomp(a, d, aux, p, r)
```

Solution: 1011110001 soit $101111,0001 = 2^5 + 2^3 + 2^2 + 2^1 + 2^0 + 2^{-4}$.

(c) Que fait ce programme ?

```
def is_leap(y):
    return (y%4==0 and not y%100==0) or y%400==0
```

Solution: 1011110001 soit $101111,0001 = 2^5 + 2^3 + 2^2 + 2^1 + 2^0 + 2^{-4}$.

(d) Test du inline python `return` 3.

Solution: 1011110001 soit $101111,0001 = 2^5 + 2^3 + 2^2 + 2^1 + 2^0 + 2^{-4}$.

(e) Donner la représentation du nombre $47,0625$ au format fixed<6,4>.

Solution: 1011110001 soit $101111,0001 = 2^5 + 2^3 + 2^2 + 2^1 + 2^0 + 2^{-4}$.

(f) (1 point) Donner la représentation du nombre $47,0625$ au format fixed<6,4>.

Solution: 1011110001 soit $101111,0001 = 2^5 + 2^3 + 2^2 + 2^1 + 2^0 + 2^{-4}$.

Que fait le programme suivant ?

```
def estbis(a):
  if (a%4==0 and a%100!=0) or (a%400==0):
    return True
  else:
    return False
```

Exercice 2

Essayer de prédire la valeur des booléens suivants, puis vérifier cette prédiction.

1. `2 < 3 or 8 <= 7`
2. `1 > 4 and 5 != 6`
3. `2 == 3 and (8 < 4 or 7 >= 6)`
4. `(2 == 3 and 8 < 4) or (7 >= 6)`

Exercice 3

Dans cet exercice, il est interdit d'utiliser les fonctions `sqrt`, `exp` et `log` !

1. Écrire une fonction `racine_carree(n)` prenant en entrée un entier strictement positif n et qui renvoie l'entier m maximal tel que $m^2 \leqslant n$.
2. Écrire une fonction `racine_pieme(n,p)` prenant en entrée deux entiers strictement positifs n et p et qui renvoie l'entier m maximal tel que $m^p \leqslant n$.
3. Écrire une fonction `nombre_chiffres(n)` donnant le nombre de chiffres d'un entier.

Exercice 4 : Suite de Syracuse

On considère la suite u définie par $u_0 \in \mathbb{N}^\star$ et

$$\forall n \in \mathbb{N}, \quad u_{n+1} = \begin{cases} 3u_n + 1 & \text{si } u_n \text{ est impair} \\ \frac{u_n}{2} & \text{si } u_n \text{ est pair} \end{cases}$$

Compléter à la main :

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13
u_n	3													

On conjecture que quel que soit l'entier u_0 choisi, il existe un rang N tel que $u_N = 1$.

Pour un entier u_0 donné, on nomme :

- *durée du vol*, l'entier défini par : $\min\{N \in \mathbb{N}^\star, u_N = 1\}$
- *altitude maximale du vol*, l'entier défini par : $\max\{u_n, n \in \mathbb{N}\}$
- *durée de vol en altitude*, l'entier défini par : $\max\{n \in \mathbb{N}, \forall k \in [\![0, n]\!], u_k \geqslant u_0\}$

1. Que valent la durée du vol, l'altitude maximale et la durée de vol en altitude pour $u_0 = 3$?
2. Écrire une fonction `suivant(u)` prenant en entrée un entier u représentant un terme quelconque de la suite et renvoyant le terme suivant. Par exemple `suivant(8)` renverra 4 et `suivant(3)` renverra 10.
3. Écrire une fonction `syracuse(u0,n)` prenant en entrée le premier terme u_0 ainsi qu'un entier strictement positif n et renvoyant le *nième* terme de la suite de Syracuse associée (utiliser la fonction précédente).
4. Écrire une fonction `duree_vol(u0)` prenant en entrée le premier terme et renvoyant la durée du vol de la suite de Syracuse associée.
5. Modifier la fonction précédente pour qu'elle renvoie également l'altitude maximale.
6. Écrire une fonction `duree_vol_alt(u0)` renvoyant la durée de vol en altitude.
7. Compléter :

u_0	7	26	27	28	703
Durée du vol					
Altitude maximale					
Durée de vol en altitude					

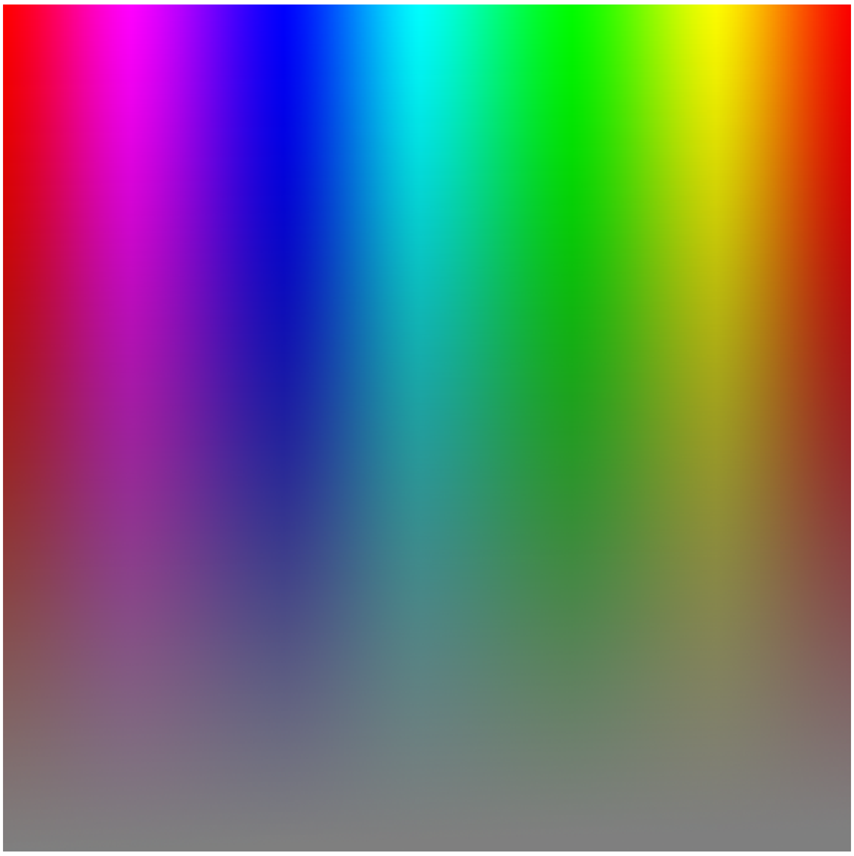

Figure 1