Numéro d'anonymat :

Partiel LU2IN003

Jeudi 17 Mars 2022, 1.5 heures
Aucun document autorisé

Exercice 1 – Fonction itérative (6 points)

La suite de Pell est définie par $P_0 = 0$, $P_1 = 1$ et $P_n = 2P_{n-1} + P_{n-2}$ si $n \geq 2$.
On considère la fonction `pell`, dont le paramètre est un entier naturel :

```
def pell(n):
    if (n == 0) or (n == 1):
        return n
    x = 0 ; y = 1 ; z = 0
    i = 1
    print('z␣=', z, ',␣x␣=', x, ',␣y␣=', y, ',␣i␣=', i)
    while (i != n):
        z = x + 2 * y ; x = y ; y = z
        i = i + 1
        print('z␣=', z, ',␣x␣=', x, ',␣y␣=', y, ',␣i␣=', i)
    return y
```

Question 1
Exécuter l'appel de `pell(4)`, en ne donnant que les affichages. Donner la valeur de `pell(4)`.
Solution:

```
z = 0 , x = 0 , y = 1 , i = 1
z = 2 , x = 1 , y = 2 , i = 2
z = 5 , x = 2 , y = 5 , i = 3
z = 12 , x = 5 , y = 12 , i = 4
```

La valeur de `pell(4)` est 12.

Question 2
Montrer que l'algorithme se termine.
Solution:
La boucle est exécutée $n - 1$ fois et chaque itération est constituée d'un nombre fini d'instructions élémentaires donc l'algorithme se termine.

Question 3
Pour $i \in \{1, \ldots, n\}$, soit x_i et y_i les valeurs respectives des variables x et y aux instants suivants :
— $x_1 = 0$ et $y_1 = 1$ sont les valeurs de x et y avant d'entrer dans la boucle ;
— pour $i \in \{2, \ldots, n\}$, x_i et y_i sont les valeurs de x et y à la fin du corps de boucle (juste après l'incrémentation de i).

1. Montrer, par récurrence sur i, que pour $1 \leq i \leq n$, on a : $x_i = P_{i-1}$ et $y_i = P_i$.
2. En déduire que `pell(n)` calcule P_n.

Solution:

1. **Base** $i = 1$: $x_1 = 0 = P_0$ et$y_1 = 1 = P_1$.
   Induction Soit i tel que $1 \leq i < n$. Supposons que $x_i = P_{i-1}$ et $y_i = P_i$.
   Dans l'itération suivante, on a :
   $z = x_i + 2 * y_i = P_{i-1} + 2P_i = P_{i+1}$,
   $x = y_i = P_i$ donc $x_{i+1} = P_i$,
   $y = z$ donc $y_{i+1} = P_{i+1}$.
   La propriété est donc vraie au rang $i + 1$.
   Conclusion On a montré, par récurrence sur i, que pour $1 \leq i \leq n$, on a : $x_i = P_{i-1}$ et $y_i = P_i$.
2. En sortie de boucle $i = n$ donc $y_i = P_i = P_n$. Par conséquent `pell(n)` calcule P_n.

Question 4
Calculer le nombre d'opérations arithmétiques effectuées par `pell(n)`. En déduire un ordre de grandeur de la complexité de `pell(n)`.
Solution:
Pour $n \geq 2$, il y a $n - 1$ tours de boucle et il y a 3 opérations arithmétiques par tour de boucle (2 additions et 1 multiplication). En tout, il y a donc $3(n - 1)$ opérations arithmétiques.
Remarque : si l'on ne compte pas l'incrémentation de i, il y a $2(n - 1)$ opérations arithmétiques.
La complexité de `pell(n)` est en $\Theta(n)$.

Exercice 2 – Etude d'une fonction récursive (9 points)

On considère la fonction suivante. On note `tab` une liste d'entiers de $m \geq n$ entiers.

```
def f(tab,n):
    print("Appel␣avec␣n=",n)
    if (n <= 0):
        res = 0
    else:
        if (n % 2) == 1:
            res = tab[n-1]+f(tab,n-2)
        else:
            res = f(tab,n-1)
    print("f(tab,",n,")␣=␣",res)
    return res
```

L'instruction `n%2` retourne 0 si n est pair, 1 sinon.

Question 1

1. Exécuter l'appel de `f(tab,6)` pour `tab = [3, -2, 2, 5, 3, 2]` en ne donnant que les affichages et la valeur finale retournée.
2. Que calcule `f(tab,n)` pour un entier $n \geq 0$ et une liste `tab` de $m \geq n$ entiers ? Aucune justification n'est demandée. On suppose que la somme des éléments d'un tableau vide est égal à 0.

Solution:

1. ```
   Appel avec n= 6
   Appel avec n= 5
   Appel avec n= 3
   Appel avec n= 1
   ```
   ```

```
Appel avec n= -1
f(tab, -1 ) =  0
f(tab, 1 ) =  3
f(tab, 3 ) =  5
f(tab, 5 ) =  8
f(tab, 6 ) =  8
8
```

2. La fonction retourne la somme des éléments d'indice pair du sous-tableau $tab[0..n-1]$: $f(tab,n) = \sum_{i=0,i\ pair}^{n-1} tab[i]$.

Question 2
Soit `tab` est une liste de m entiers avec $m \geq n \geq 0$; on définit la suite d'entiers $S_{-1}, S_0, \ldots, S_n$ par $S_{-1} = S_0 = 0$ et $S_n = \sum_{i=0,i\ pair}^{n-1} tab[i]$ pour $n \geq 1$. Montrer que, pour tout $n > 0$,

1. $S_n = S_{n-1}$ si n est pair;
2. $S_n = S_{n-2} + tab[n-1]$ si n est impair.

__**Solution**__:
Soit $n > 0$.

1. Si n est pair, $n-1$ est impair, et donc on ne peut avoir de terme $tab[n-1]$ dans S_n. Ainsi, $S_n = \sum_{i=0,i\ pair}^{n-2} tab[i] = S_{n-1}$.
2. Si n est impair, $S_n = S_{n-1} + tab[n-1]$. Comme précédemment, $n-1$ est pair. De plus, $n \geq 2$, donc en appliquant l'égalité précédente, on obtient $S_n = S_{n-2} + tab[n-1]$.

Question 3
Montrer par récurrence la propriété $\mathcal{P}$ définie de la manière suivante pour tout $n \in \{-1\} \cup \mathbb{N}$:
$\mathcal{P}(n)$: pour toute liste `tab` de $m \geq n$ entiers, l'appel `f(tab,n)` se termine et retourne S_n.
__**Solution**__:

On démontre que pour tout $n \geq -1$, la propriété est vérifiée par *récurrence forte*.

Base : Pour $n = 0$ ou $n = -1$, la liste est vide et la fonction retourne 0. $\mathcal{P}(0)$ et $\mathcal{P}(-1)$ sont donc vérifiées.

Induction : Soit $n > 0$ tel que, pour tout $k \in \{-1, 0, \ldots, n-1\}$, $\mathcal{P}(k)$ soit vérifiée.
Soit alors tab une liste de $m \geq n$ éléments. Il faut considérer deux cas selon la parité de n.

1. Si n est pair, alors $n \geq 2$. Dans ce cas, la fonction retourne `f(tab,n-1)`. Par hypothèse de récurrence, cet appel se termine et retourne S_{n-1}. Or, $S_n = S_{n-1}$ dans ce cas, donc la propriété est vérifiée.
2. Si n est impair, alors $n \geq 1$. Là aussi, l'appel de fonction `f(tab,n-2)` se termine et retourne S_{n-2}. Or, on a vu que $S_n = tab[n-1] + S_{n-2}$, donc la propriété est vérifiée.

On en déduit que $\mathcal{P}(n)$ est vérifiée.

Conclusion : Pour tout $n \geq -1$, $\mathcal{P}(n)$ est vérifiée par récurrence forte.

Question 4
Soit u_n, $n \geq 0$, le nombre d'additions effectuées pour une liste de n éléments.

1. Que valent u_0 et u_{-1} ? Quelle est la définition récurrente de u_n selon la parité de n ?

2. Soit alors la suite $v_p = u_{2p+1}$ pour $p \geq 0$. Donner le terme général de cette suite ;
3. En déduire les valeurs u_{2p} et u_{2p+1} pour $p \geq 0$ et le terme général de la suite u_n ;
4. En déduire la complexité de la fonction `f`.

Solution:

1. $u_0 = u_{-1} = 0$. Si $n > 0$ et n pair, $u_n = u_{n-1}$. Sinon, si $n > 0$ et n impair, $u_n = 1 + u_{n-2}$.
2. Soit la suite $v_p = u_{2p+1}$. Alors, $v_0 = u_1 = 1$ et $v_p = 1 + u_{2p-1} = 1 + v_{p-1}$. Par substitution, on obtient $v_p = p + 1$.
3. $u_{2p+1} = v_p = p + 1$; $u_{2p} = u_{2p-1} = v_{p-1} = p$. Donc, $u_n = \lceil \frac{n}{2} \rceil$ pour $n \geq 0$.
4. On en déduit que la complexité est en $\Theta(n)$.