Correction du DS 1

Informatique pour tous, deuxième année

Julien Reichert

Exercice 1

Voir le cours et le TP 1, les deux implémentations, naturelle et à l'aide de tableaux, étaient acceptées. Toute autre éventuelle implémentation cohérente pouvait également être acceptée.

Exercice 2

D'après la formule du cours, on a $c_n = ac_{\frac{n}{b}} + \mathcal{O}(n^\alpha)$ avec $\log_b(a) = \alpha = 0$. Ainsi, $c_n = \mathcal{O}(n^0 \log n)$ soit une complexité logarithmique. C'est le cas d'une dichotomie simple.

Exercice 3

C'est la combinaison de deux exercices du TP 2 :

```
def multiparenthesage(L):
    parentheses = []
    couples = []
    for i in range(1,len(L)+1):
        if L[-i] < 0:
            parentheses.append((L[-i],len(L)-i))
        elif L[-i] > 0:
            try:
                (type,position) = parentheses.pop()
                assert(type == -L[-i])
                couples.append((len(L)-i,position))
            except:
                raise ValueError("Mauvais parenthesage")
    if parentheses == []:
        return list(reversed(couples))
    else:
        raise ValueError("Une parenthese fermante au moins est de trop")
```

On notera que la lecture de la liste se fait à l'envers. La raison est qu'alors les couples seront ordonnés par position croissante des parenthèses ouvrantes.

Exercice 4

Le principe est de trouver la récursion comme dans le problème standard des tours de Hanoï. Ici, on veut que les anneaux de 0 à k (c'est-à-dire les $k+1$ plus petits) soient alignés, et c'est vrai pour $k = 0$ de base. Ensuite, pour passer de k à $k+1$, on doit savoir comment envoyer k anneaux de n'importe quelle position à n'importe quelle autre position (c'est le problème classique!), en l'occurrence la position où se site l'anneau $k+1$, ce qui peut être en l'occurrence déjà fait.

D'où le programme :

```
def hanoi_base(n, dep, arr):
    if n == 1:
        print("Envoyer le sommet de la pile %d vers la pile %d."%(dep, arr))
    else:
        hanoi_base(n-1, dep, 6-dep-arr)
        hanoi_base(1, dep, arr)
        hanoi_base(n-1, 6-dep-arr, arr)

def hanoi_en_cours(l1, l2, l3):
    l = [l1, l2, l3]
    n = len(l1)+len(l2)+len(l3)
    positions = [0] * n
    for i in range(3):
        for element in l[i]:
            positions[element] = i+1
    dep = positions[0]
    for k in range(1, n):
        arr = positions[k]
        if dep != arr:
            hanoi_base(k, dep, arr)
            dep = arr
```

Exercice 5

Premier puzzle (numéro 24)

F1 : ↑↷ F2
F2 : ↑↶ F1

Deuxième puzzle (numéro 39)

F1 : F3 F2 F3 F3 F2 F2 F3 F2 F2 F3
F2 : ↑↑↷
F3 : ↑↑↶

On peut finir F1 par F2, le tout est de ne pas oublier de lancer une fonction qui avance encore. L'avantage est que le nombre d'instructions allouées permet d'éviter cette omission.

Troisième puzzle (numéro 539)

F1 : ↑ ↶ F2 F1
F2 : ↶↶ F3
F3 : ↑ ↷ F3

D'après le site, il existe une solution avec sept instructions.

Quatrième puzzle (numéro 648)

Une fonction est inutile, et dans l'idée on se sert de trois fonctions pour simuler une fonction avec trois instructions plus un appel récursif à la fonction elle-même :

F1 : ↶ F2
F2 : ↑ F3
F3 : ↶ F4
F4 : ↶ F2

Cinquième puzzle (numéro 587)

Certaines instructions sont automatiques, ensuite il reste à broder :

F1 : ↶ ↑ F2 F1
F2 : ↑↷↷↑

Autre solution :

F1 : ↑ ↶ F2 F1
F2 : ↑ ↷↷ F2

D'après le site, il existe une solution avec sept instructions.

Sixième puzzle (numéro 656)

C'était le puzzle le plus compliqué. En pratique, les colonnes étaient de taille si chaotique qu'on devait se douter que ces tailles n'avaient pas d'importance (peut-être y a-t-il un message codé, ce serait amusant !). Ainsi, il fallait empiler des instructions d'avancement mises en attente par des appels récursifs prioritaires. À l'instar du premier TP de première année, la question qu'on peut se poser est « Comment passer de la configuration de départ à la même configuration mais un cran en avant et en ayant visité la colonne ? », et la réponse est par la fonction F1 dont la partie concernant la colonne se fait dans F2, au rétablissement de l'orientation près.

F1 : ↶ F2 ↶↑ F1
F2 : ↑ F2 ↶↶ ↑

L'ordre des trois instructions du milieu de F2 n'est pas important.

D'après le site, il existe une solution avec neuf instructions.