DM - Algorithmique - Master 1 Bio-informatique

2020

Exercice 1

Proposez un algorithme qui prends en paramètres une liste Python d'entiers l et qui renvoie vrai si un élément apparait deux fois ou plus.

Déterminer la complexité dans le pire cas de votre algorithme.

Exercice 2

Donnez la définition d'une pile. Illustrez votre définition par un exemple.

Exercice 3

Dans cet exercice, vous devez utilisez uniquement les piles données à l'aide des fonctions primitives suivantes :

```
def creer_pile( ):
    # renvoie une nouvelle pile vide

def empiler( p, e ):
    # empile dans la pile p l'élément e.

def depiler( p ):
    # dépile l'élément situé en heut de la pile p et le renvoie.

def taille( p ):
    # renvoie le nombre d'éléments contenus dans la pile p.
```

Dans cet exercice, il n'est pas autorisé d'utiliser des listes, tuples et dictionnaires du langage python.

1. Proposez un algorithme *copie_et_renverse*(p) qui prends en paramètre une pile p et qui renvoie une copie de p dont les éléments sont ordonnées dans l'ordre inverse. La pile p doit, à la fin du processus, rester inchangée.
2. Un mot w est un carré si il existe un mot u tel que $w = uu$. Par exemple, *aabaab* est un carré, mais pas *aababb*.

   Proposez un algorithme qui n'utilise que les primitives de piles et qui détermine si un mot passé en paramètre est un carré.

Exercice 4

Le but de cet exercice est d'écrire un programme récursif qui renvoie toutes les partitions de n. Une partition de n est une liste d'entier ordonné du plus grand au plus petit telle que la somme des entiers fassent n.

Par exemple, les partitions de 5 sont :

$$[5], [4, 1], [3, 2], [3, 1, 1], [2, 2, 1], [2, 1, 1, 1], [1, 1, 1, 1, 1].$$

En fait, on peut remarquer que si l'on prend une partition de n et que l'on retire le premier élément a de cette partition, alors on obtient une partition de $n - a$.

Plus précisément, si je retire le premier entier de chaque partition de 5, j'obtiens :

$$[], [1], [2], [1, 1], [2, 1], [1, 1, 1], [1, 1, 1, 1],$$

qui sont respectivement :

- pour $a = 5$, toutes les partitions de $5 - a = 0$ dont les éléments sont plus petits que l'entier 5 retiré : [] ;
- pour $a = 4$, toutes les partitions de $5 - a = 1$ dont les éléments sont plus petits que l'entier 4 retiré : $[1]$;
- pour $a = 3$, toutes les partitions de $5 - a = 2$ dont les éléments sont plus petits que l'entier 3 retiré : $[2], [1, 1]$;
- pour $a = 2$, toutes les partitions de $5 - a = 3$ dont les éléments sont plus petits que l'entier 2 retiré : $[2, 1], [1, 1, 1]$;
- pour $a = 1$, toutes les partitions de $5 - a = 4$ dont les éléments sont plus petits que l'entier 1 retiré : $[1, 1, 1, 1, 1]$.

1. Proposez une fonction récursives $partitions\_bornee(n, b)$ qui prends en paramètre deux entiers n et b et qui renvoie toutes les partitions de n dont les éléments sont plus petits que b.
2. Proposez une fonction $partitions(n)$ qui prends en paramètre un entier n et qui renvoie une liste de toutes les partitions de n.