Devoir Maison 2

Pour le 9 janvier 2014

Consignes

Vous répondez sur papier aux questions qui appellent une réponse rédigée et argumentée. En parallèle, vous rédigez les fonctions et programmes Python demandés dans deux fichiers : un par partie ; indiquez clairement en commentaire à quelle question vous répondez.

Problème. Listes au père noël binaire

Étant donné un entier $n \in \mathbb{N}^*$, on désigne par $\mathcal{L}_n$ l'ensemble des listes de longueur n contenant uniquement des 0 et des 1.

Exemples. `[1,0,1]` est un élément de $\mathcal{L}_3$; `[0,0,0,0,0]` est un élément de $\mathcal{L}_5$.

1. À n fixé, combien existe-t-il de listes dans $\mathcal{L}_n$?
   On a défini une fonction Python `ensemble_liste` qui a pour argument un entier $n \geq 0$ et retourne la liste de **toutes** les listes de $\mathcal{L}_n$; cette fonction est rédigée dans un fichier accessible sur la page :
   `etablissementbertrandeborn.net/blogprepasciences/`
   dans le billet consacré à ce devoir.

Partie I. Plages de valeurs identiques

Les algorithmes et fonctions définies ci-dessous ont vocation à opérer sur des listes de $\mathcal{L}_n$.

2. Définir une fonction Pyhton qui prend une liste en argument et affiche le nombre de 0 et le nombre de 1 dans la liste.
3. Concevoir un algorithme qui donne la taille de la plus longue plage de 0 consécutifs d'une liste.
   Exemple. Sur la liste `[0,0,1,0,0,0,1,0,0,1,1,1,0]` l'algorithme retourne le nombre 3 car la liste se décompose comme suit :
   $$[\underbrace{0,0}_{2},1,\underbrace{0,0,0}_{3},1,\underbrace{0,0}_{2},1,1,1,\underbrace{0}_{1}]$$
   Indication. *Une variable locale `Max0`, initialisée à 0, va recevoir la valeur maximale cherchée ; parcourir la liste avec une boucle* ***`Tant que`*** *. À chaque rencontre d'un élément nul, un compteur démarre et s'incrémente d'une unité* ***tant que*** *les éléments rencontrés sont nuls ; à chaque fin de cette seconde boucle, la valeur de `plageMax` est actualisée si besoin.*
4. Traduire votre algorithme en une fonction Python `plageMax0` puis tester-le.
5. On se donne un entier $n \geq 1$.
   (a) En s'aidant de la fonction `ensemble_liste`, écrire un programme qui donne le nombre de listes de $\mathcal{L}_n$ n'ayant jamais (au moins) trois zéro consécutifs : pour cela, on parcourt tous les éléments de `ensemble_liste(n)` puis on incrémente un compteur à chaque fois qu'on passe sur une liste dont `plageMax0` retourne une valeur inférieure ou égale à 2.
   (b) Écrire un programme qui donne le nombre de listes de $\mathcal{L}_n$ n'ayant jamais trois valeurs identiques consécutives. On peut utiliser une fonction `plageMax1`.
6. On souhaite toujours réaliser un programme qui donne le nombre de listes de $\mathcal{L}_n$ n'ayant jamais trois valeurs identiques consécutives.
   (a) Expliquer pourquoi la stratégie employée à la question 5(b) nous amène à faire plus de calculs que nécessaire.
   (b) Proposer une méthode plus efficace pour détecter une liste contient plus de trois valeurs identiques à la suite. Reprendre la question 5 et comparer les vitesses d'exécution des méthodes.

Partie II. Décorations de noël

On se propose de réaliser une frise décorative de noël avec deux motifs : une **boule rouge** et une **étoile dorée**.

On place comme ci-dessous les motifs les uns à la suite des autres avec une seule contrainte : **ne jamais aligner trois motifs identiques à la suite**[1]. On appelle cela une « 3-frise ».

On cherche à calculer le nombre de 3-frises distinctes que l'on peut réaliser avec un nombre fixé de motifs (par exemple, la 3-frise ci-dessus contient 9 motifs).

7. **Petits dénombrements.** Trouver à la main et donner toutes les 3-frises faites avec 1, 2, 3 et 4 motifs.
8. **Dénombrements un peu plus grands.**
   (a) Expliquez comment les listes et méthodes étudiées dans le paragraphe précédent peuvent s'appliquer dans ce contexte.
   (b) Faire une fonction Python qui, pour un entier n donné en argument, affiche les 3-frises fabriquées avec n motifs. On affiche les frises sous forme de liste ; pour désigner une boule rouge : utiliser le caractère `'o'` ; pour désigner une étoile : utiliser le caractère `'*'`.
   (c) Reprendre la fonction précédente et la modifier pour lui faire retourner seulement le nombre de 3-frises.
   (d) À partir de quelle valeur de n (donner un ordre de grandeur) cette méthode est-elle mise en défaut ?
9. **Dénombrements beaucoup plus grands.** Pour entier $n \geq 1$, on note T_n le nombre total de 3-frises avec n motifs puis :

$$\begin{cases} A_n : & \text{le nombre de 3-frises avec } n\text{-motifs qui se terminent par } \texttt{*} ; \\ B_n : & \text{le nombre de 3-frises avec } n\text{-motifs qui se terminent par } \texttt{o} ; \end{cases}$$

   On note également $(A_{n+1}|_{B_n})$ le nombre de 3-frises avec $n+1$ motifs qui se terminent par `*` et dont le n-ième motif est `o`. La notation $(B_{n+1}|_{A_n})$ se définie naturellement en échangeant les rôles de `*` et `o` dans la définition précédente.
   (a) Pour tout $n \geq 1$ expliquez les égalités suivantes :

$$\begin{cases} T_n = A_n + B_n \\ A_n = B_n \\ A_{n+2} = B_{n+1} + (A_{n+1}|_{B_n}) \\ (A_{n+1}|_{B_n}) = B_n \end{cases}$$

   (b) En déduire que pour tout $n \geq 1$, $T_{n+2} = T_{n+1} + T_n$.
   (c) Déterminer le nombre de 3-frises que l'on peut réaliser avec 1000 motifs. *Dès que vous avez obtenu une valeur qui vous semble convenable : postez cette valeur en commentaire au billet consacré à ce devoir maison sur la page internet de la classe. Le premier à poster une valeur juste remporte 1 point comptant pour le chalenge 2013-2014 des défis maths-info.*
   (d) Déterminer le nombre de 3-frises que l'on peut réaliser avec 2013^{2014} motifs. *Même défi qu'à la question précédente.*

1. Il est peu connu que le père noël souffre d'un trouble obsessionnel compulsif qui lui empêche de livrer en cadeaux les maisons où au moins trois décorations identiques se suivent. Les experts nomment cela la *morpionophobie*