Programmation dynamique
Pyramide de nombres

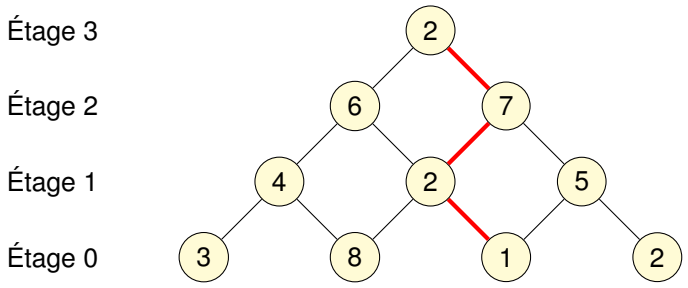
M. GEORGES-SAINT-MARC - NSI - Lycée Mendès France Rennes

23 mars 2021

L'objectif de ce TP est de mesurer l'intérêt de la programmation dynamique sur la programmation récursive "standard".

1 Problème étudié

On s'intéresse ici à une pyramide de nombres : à chaque étage, des nombres entiers sont empilés les uns sur les autres. Chaque nombre repose sur 2 nombres de l'étage du dessous, comme l'illustre l'exemple ci-dessous :



L'objectif du problème est de déterminer la somme la plus importante en partant du sommet de la pyramide et en se rendant jusqu'à sa base (étage 0), en descendant d'un étage à chaque étape et en additionnant les valeurs rencontrées à chaque sommet.

Ainsi le chemin en rouge vaudra $2 + 7 + 2 + 1 = 12$.

Exercice 1 :

1. Combien y a-t-il de chemins au total dans l'exemple ci-dessus ?

   .....

2. Déterminer la somme la plus grande en partant du sommet ci-dessus.

   .....

3. Combien y a-t-il de chemins pour une pyramide à 2 étages ? à 3 étages ? à n étages (pour tout entier naturel $n \geqslant 2$) ?

   .....

   .....

   .....

2 Créer une pyramide pour résoudre le problème en Python

Exercice 2 :

On cherche ici à représenter une pyramide de la façon la plus simple.
On va la construire comme étant une liste de listes de longueurs variables, en respectant la propriété suivante : l'index n de la liste contiendra l'étage n de la pyramide, selon la logique du schéma ci-dessus.

1. Donner en Python la liste $L1$ qui décrit la pyramide ci-dessus.

.....

.....

.....

.....

2. Saisir cette liste $L1$ dans le fichier élève du TP qui vous a été fourni.

→ ***On complétera aussi les fonctions étudiées dans les questions suivantes dans ce même fichier.***

3. Créer une fonction $creerPyramide(n, max)$ qui prend en paramètres :

- n le nombre d'étages de la pyramide
- max le nombre entier maximal que peut contenir la pyramide.
  Cette fonction renvoie la liste associée à une pyramide remplie de valeurs aléatoires entre 1 et max.

3 Résolution du problème par récursivité

On souhaite résoudre le problème par récursivité.

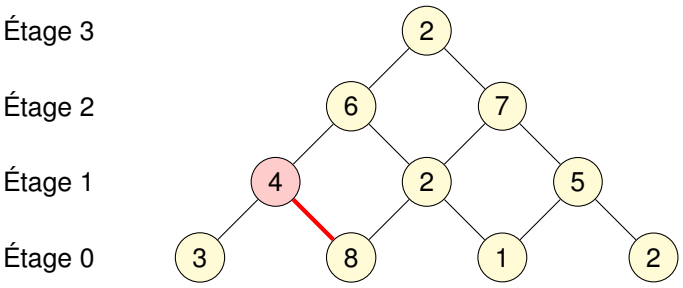
Exercice 3 :

Pour une pyramide à N étages, on numérotera toujours les étages de la base (0) vers son sommet $(N-1)$.
À l'étage n, on notera :

- i la position de chaque sommet à partir de la gauche (i va donc de 0 jusqu'à $n-1$) ;
- $c_n(i)$ le coefficient de la pyramide situé à la position i ;
- $m_n(i)$ le chemin maximal en partant de la base de la pyramide jusqu'à ce sommet i de l'étage n.

Ainsi, dans l'exemple initial, on a :

- $c_1(0) = 4$;
- $m_1(0) = 12$
  → entre les deux chemins possibles jusqu'à $c_1(0)$, le plus grand est 12 (il y a $3+4=7$ et $8+4=12$).



→ Les c_i sont donc tous connus au départ (ce sont les valeurs de la pyramide)

→ l'objectif est de calculer le chemin maximal jusqu'au sommet de la pyramide : $m_{N-1}(0)$ (pour une pyramide à N étages).

1. Dans l'exemple de départ, donner les valeurs de $c_1(1)$ et $m_1(1)$.

.....

.....

2. Après avoir donné $c_2(0)$, déduire la valeur de $m_2(0)$ à partir des exemples donnés précédemment (dans introduction de l'exercice et à la question 1).

.....

.....

.....

.....

3. Lorsqu'on se place à l'étage $n=0$ et sur le sommet i, expliquer simplement pourquoi on a $m_0(i) = c_0(i)$?

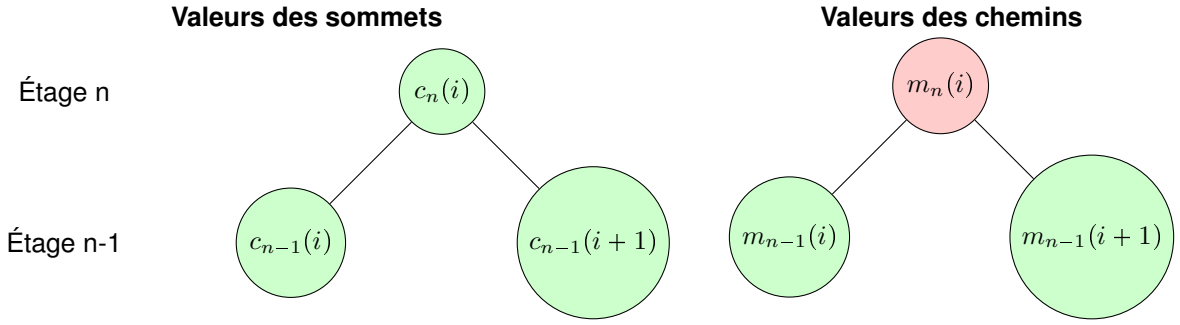
.....

.....

.....

.....

4. On suppose connue la valeur du chemin maximal jusqu'aux sommets i et $i+1$ de l'étage $n-1$: on connaît donc $m_{n-1}(i)$ et $m_{n-1}(i+1)$.



→ En vert : les valeurs (sommets / chemins) connues

→ En rouge : ce qu'il faut déterminer à partir des valeurs connues.

En déduire la valeur du maximum au sommet i de l'étage n.

.....

.....

.....

.....

5. Compléter le code Python suivant permettant, pour une liste donnée, de calculer le chemin de valeur maximale en partant du sommet de la pyramide.

```
1 def cheminMax(L):
2     return cheminMaxRecursif(L,len(L)-1,0)
3
4 def cheminMaxRecursif(L,n,i):
5     if n == 0:
6         return ...................
7     else:
8         return ..........................................................
```

6. Tester l'algorithme avec les deux listes fournies dans le fichier de TP puis indiquer la valeur du chemin maximal pour ces listes :

- $L2$: .....
- $L3$: .....

7. En appelant $cheminMax(L1)$, combien de fois $cheminMaxRecursif(L1,1,2)$ sera-t-il appelé ?

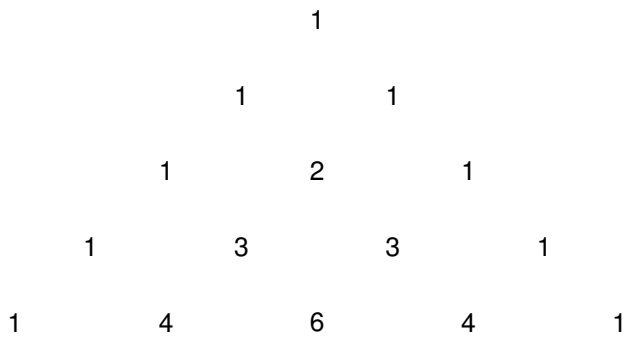
.....

.....

.....

.....

8. On peut trouver une règle générale pour connaître le nombre d'appels à $cheminMaxRecursif$ selon l'emplacement concerné dans la pyramide. Ce nombre d'appels est donné par le **triangle de Pascal** :



Déterminer la règle pour calculer les coefficients de la ligne suivante du triangle, puis compléter la 6ème ligne ci-dessus.

.....

.....

.....

4 Programmation dynamique

4.1 Méthode descendante

On fonctionnera ici par méthode descendante (*top-down*) :

Exercice 4 :

Compléter les fonctions suivantes, en s'inspirant de l'algorithme récursif et des exemples vus dans le TP précédent.

```
def cheminMaxDynamique(L):
    ......................
    tableauCheminsMax = [[0 for j in range(i)] for i in range(len(L),0,-1)]
    return cheminMaxRecursifDynamique(L,len(L)-1,0)

def cheminMaxRecursifDynamique(L,n,i):
    if

    elif

    else:

        return tableauCheminsMax[n][i]
```

4.2 Pour aller plus loin

Exercice 5 :

On indique dans les schémas suivants le nombre de passages par chaque emplacement de la pyramide selon les méthodes utilisées.

On a pour l'exemple pris une pyramide à 5 étages.

Méthode dynamique	Méthode récursive standard
1 1 1 1 2 1 1 2 2 1 1 2 2 2 1	1 1 1 1 2 1 1 3 3 1 1 4 6 4 1

1. Combien d'étapes sont économisées avec la méthode dynamique ?

.....

2. Qu'en sera-t-il pour une pyramide à 6 étages ?

.....

3. Reprenons les pyramides à 5 étages ci-dessus.

a) Pour chaque méthode, faire la somme des coefficients de chaque étage (par exemple, $1+2+2+2+1=8$ pour l'étage 0 de la méthode dynamique).

.....

.....

.....

.....

.....

b) Que constate-t-on pour la méthode dynamique ? Et pour la méthode récursive standard ?

.....

.....

c) En déduire l'écart entre les méthodes pour une pyramide à 10 lignes.

.....

.....

Exercice 6 : *Bonus*

Proposer une méthode dynamique *bottom-up* permettant la résolution du problème.