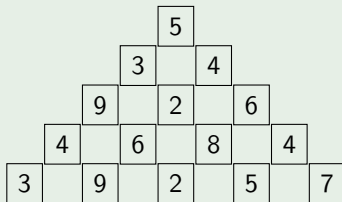


C7 Algorithme glouton

1. Exemple introductif

Chemin de somme maximale dans une pyramide

On s'intéresse à la recherche de la somme maximale d'un chemin qui part du haut de la pyramide et atteint sa base.

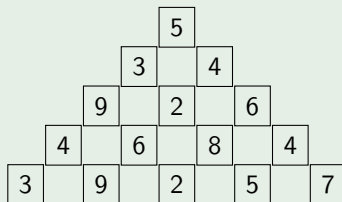


C7 Algorithme glouton

1. Exemple introductif

Chemin de somme maximale dans une pyramide

On s'intéresse à la recherche de la somme maximale d'un chemin qui part du haut de la pyramide et atteint sa base.



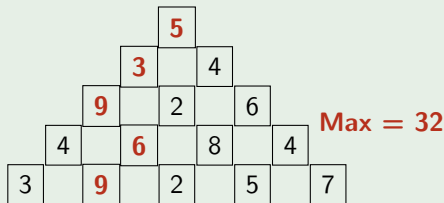
- 1 déterminer cette somme et un chemin possible sur l'exemple ci-dessus.

C7 Algorithme glouton

1. Exemple introductif

Chemin de somme maximale dans une pyramide

On s'intéresse à la recherche de la somme maximale d'un chemin qui part du haut de la pyramide et atteint sa base.



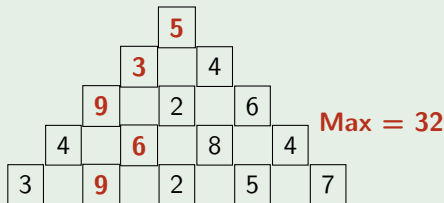
- 1 déterminer cette somme et un chemin possible sur l'exemple ci-dessus.

C7 Algorithme glouton

1. Exemple introductif

Chemin de somme maximale dans une pyramide

On s'intéresse à la recherche de la somme maximale d'un chemin qui part du haut de la pyramide et atteint sa base.



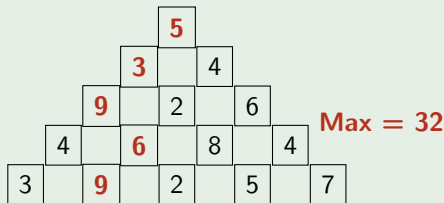
- 1 déterminer cette somme et un chemin possible sur l'exemple ci-dessus.
- 2 On suppose qu'on applique la stratégie suivante : « à chaque niveau on choisit de descendre vers le cube de plus grand valeur ». Quel chemin obtient-on et quelle somme à ce chemin ?

C7 Algorithme glouton

1. Exemple introductif

Chemin de somme maximale dans une pyramide

On s'intéresse à la recherche de la somme maximale d'un chemin qui part du haut de la pyramide et atteint sa base.



- 1 déterminer cette somme et un chemin possible sur l'exemple ci-dessus.
- 2 On suppose qu'on applique la stratégie suivante : « à chaque niveau on choisit de descendre vers le cube de plus grand valeur ». Quel chemin obtient-on et quelle somme à ce chemin ?

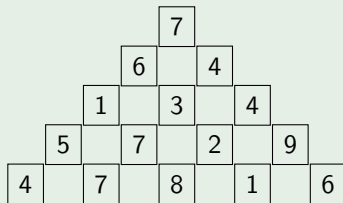
On obtient le chemin 5-4-6-8-5 de somme 28.

C7 Algorithme glouton

1. Exemple introductif

Chemin de somme maximale dans une pyramide

Même question pour la pyramide suivante :

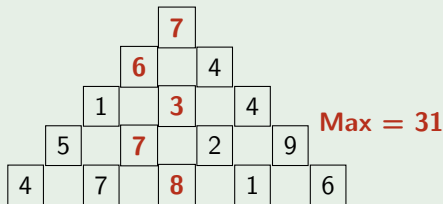


C7 Algorithme glouton

1. Exemple introductif

Chemin de somme maximale dans une pyramide

Même question pour la pyramide suivante :

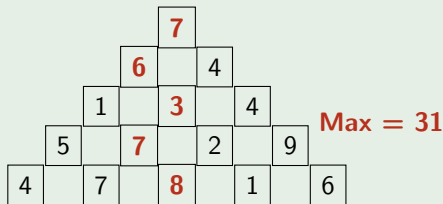


C7 Algorithme glouton

1. Exemple introductif

Chemin de somme maximale dans une pyramide

Même question pour la pyramide suivante :



Cette fois l'algorithme qui consiste à choisir de descendre vers le cube de plus grand valeur permet d'obtenir la plus grande somme.

C7 Algorithme glouton

1. Exemple introductif

Implémentation en Python

- On représente une pyramide par une *liste de listes*, chacune des listes contient les coefficients d'un niveau de la pyramide.

Par exemple, la pyramide précédente correspond à la liste

```
[[7], [6,4], [1,3,4], [5,7,2,9], [4,7,8,1,6] ]
```

C7 Algorithme glouton

1. Exemple introductif

Implémentation en Python

- On représente une pyramide par une *liste de listes*, chacune des listes contient les coefficients d'un niveau de la pyramide.

Par exemple, la pyramide précédente correspond à la liste

`[[7], [6,4], [1,3,4], [5,7,2,9], [4,7,8,1,6]]`

- On écrit alors une fonction `glouton` qui prend en argument une liste de liste et renvoie la somme obtenue en suivant la stratégie gloutonne (descendre vers le cube de plus grand valeur). Pour cela, on se donne un variable `colonne` qui contiendra la colonne dans laquelle on se trouve dans la pyramide.

C7 Algorithme glouton

1. Exemple introductif

Implémentation en Python

- On représente une pyramide par une *liste de listes*, chacune des listes contient les coefficients d'un niveau de la pyramide.

Par exemple, la pyramide précédente correspond à la liste
[[7], [6,4], [1,3,4], [5,7,2,9], [4,7,8,1,6]]

- On écrit alors une fonction `glouton` qui prend en argument une liste de liste et renvoie la somme obtenue en suivant la stratégie gloutonne (descendre vers le cube de plus grand valeur). Pour cela, on se donne un variable `colonne` qui contiendra la colonne dans laquelle on se trouve dans la pyramide.
- Pour chaque niveau i de la pyramide, on doit donc comparer les deux cubes se trouvant en dessous c'est à dire les cubes :
 - Situé juste en dessous c'est à dire en $(i+1, \text{colonne})$
 - Situé en dessous et à droite c'est à dire en $(i+1, \text{colonne}+1)$

C7 Algorithme glouton

1. Exemple introductif

Fonction en python

```
1 def glouton(pyramide):
2     colonne = 0
3     somme = pyramide[0][0]
4     for i in range(1, len(pyramide)):
5         if pyramide[i][colonne] > pyramide[i][colonne+1]:
6             somme += pyramide[i][colonne]
7         else:
8             somme += pyramide[i][colonne+1]
9             colonne += 1
10    return somme
```

C7 Algorithme glouton

2. Notion d'algorithme glouton

Stratégie gloutonne

Afin de résoudre un problème d'optimisation, on peut adopter une stratégie dite **gloutonne** :

C7 Algorithme glouton

2. Notion d'algorithme glouton

Stratégie gloutonne

Afin de résoudre un problème d'optimisation, on peut adopter une stratégie dite **gloutonne** :

- à chaque étape on effectue le choix qui correspond à l'optimal **local** d'une grandeur.

Stratégie gloutonne

Afin de résoudre un problème d'optimisation, on peut adopter une stratégie dite **gloutonne** :

- à chaque étape on effectue le choix qui correspond à l'optimal **local** d'une grandeur.
- Ces choix successifs ne conduisent pas forcément à la solution optimale **globale**.

Stratégie gloutonne

Afin de résoudre un problème d'optimisation, on peut adopter une stratégie dite **gloutonne** :

- à chaque étape on effectue le choix qui correspond à l'optimal **local** d'une grandeur.
- Ces choix successifs ne conduisent pas forcément à la solution optimale **globale**.
- Cette stratégie ne fournit donc pas toujours la **meilleure solution**.

C7 Algorithme glouton

2. Notion d'algorithme glouton

Stratégie gloutonne

Afin de résoudre un problème d'optimisation, on peut adopter une stratégie dite **gloutonne** :

- à chaque étape on effectue le choix qui correspond à l'optimal **local** d'une grandeur.
- Ces choix successifs ne conduisent pas forcément à la solution optimale **globale**.
- Cette stratégie ne fournit donc pas toujours la **meilleure solution**.

Définition

Un algorithme est dit **glouton** lorsqu'il procède par choix successifs, en sélectionnant à chaque étape l'option qui correspond à un optimum local sans garantie que cela conduise à l'optimum globale.







C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Position du problème

On dispose d'un sac à dos et d'une liste objet ayant chacun un poids et une valeur. Le problème du sac à dos consiste à remplir ce sac en maximisant la valeur des objets qu'il contient tout en respectant une contrainte sur le poids du sac.

Exemple

180 €  0.3 kg	2050 €  4.1 kg	280 €  0.6 kg	810 €  1.7 kg
990 €  2 kg	1275 €  2.9 kg	2570 €  5.7 kg	920 €  2.1 kg

Quels objets doit-on prendre pour maximiser la valeur contenu si le poids doit rester inférieur à **8 kg** ?

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Mise en place d'une stratégie gloutonne

On propose de résoudre ce problème en adoptant la stratégie gloutonne suivante :

- 1 On classe les objets selon un critère pertinent

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Mise en place d'une stratégie gloutonne

On propose de résoudre ce problème en adoptant la stratégie gloutonne suivante :

- 1 On classe les objets selon un critère pertinent

Ici la valeur par unité de poids semble un critère intéressant

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Mise en place d'une stratégie gloutonne

On propose de résoudre ce problème en adoptant la stratégie gloutonne suivante :

- 1 On classe les objets selon un critère pertinent
Ici la valeur par unité de poids semble un critère intéressant
- 2 On parcourt dans l'ordre la liste *triée* des objets

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Mise en place d'une stratégie gloutonne

On propose de résoudre ce problème en adoptant la stratégie gloutonne suivante :

- 1 On classe les objets selon un critère pertinent
Ici la valeur par unité de poids semble un critère intéressant
- 2 On parcourt dans l'ordre la liste *triée* des objets
- 3 Si l'objet considéré rentre dans le sac en respectant la contrainte de poids on l'ajoute (en diminuant le poids restant), sinon on passe à l'objet suivant.

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Utilisation de `sorted` pour classer les objets

- La fonction `sorted` de Python prend en argument une liste et renvoie cette liste triée dans l'ordre croissant

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Utilisation de `sorted` pour classer les objets

- La fonction `sorted` de Python prend en argument une liste et renvoie cette liste triée dans l'ordre croissant

Par exemple `sorted([2, 1, 6, 4])` renvoie la liste `[1, 2, 4, 6]`

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Utilisation de `sorted` pour classer les objets

- La fonction `sorted` de Python prend en argument une liste et renvoie cette liste triée dans l'ordre croissant
Par exemple `sorted([2, 1, 6, 4])` renvoie la liste `[1, 2, 4, 6]`
- Le paramètre optionnel `reverse` classe dans l'ordre *décroissant* lorsqu'il vaut `True`
Par exemple `sorted([2, 1, 6, 4], reverse=True)` renvoie la liste `[6, 4, 2, 1]`

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Utilisation de `sorted` pour classer les objets

- La fonction `sorted` de Python prend en argument une liste et renvoie cette liste triée dans l'ordre croissant
Par exemple `sorted([2, 1, 6, 4])` renvoie la liste `[1, 2, 4, 6]`
- Le paramètre optionnel `reverse` classe dans l'ordre *décroissant* lorsqu'il vaut `True`
Par exemple `sorted([2, 1, 6, 4], reverse=True)` renvoie la liste `[6, 4, 2, 1]`

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Utilisation de `sorted` pour classer les objets

- La fonction `sorted` de Python prend en argument une liste et renvoie cette liste triée dans l'ordre croissant
Par exemple `sorted([2, 1, 6, 4])` renvoie la liste `[1, 2, 4, 6]`
- Le paramètre optionnel `reverse` classe dans l'ordre *décroissant* lorsqu'il vaut `True` Par exemple `sorted([2, 1, 6, 4], reverse=True)` renvoie la liste `[6, 4, 2, 1]`
- On peut préciser une *clé de tri* avec le paramètre `reverse`.

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Utilisation de `sorted` pour classer les objets

- La fonction `sorted` de Python prend en argument une liste et renvoie cette liste triée dans l'ordre croissant
Par exemple `sorted([2, 1, 6, 4])` renvoie la liste `[1, 2, 4, 6]`
- Le paramètre optionnel `reverse` classe dans l'ordre *décroissant* lorsqu'il vaut `True` Par exemple `sorted([2, 1, 6, 4], reverse=True)` renvoie la liste `[6, 4, 2, 1]`
- On peut préciser une *clé de tri* avec le paramètre `reverse`.
Par exemple si on souhaite trier une liste de points (représentés par des tuples) par ordonnée croissante, on crée une fonction renvoyant l'ordonnée d'un point :

```
1 def ordonnee(p):  
2     x,y = p  
3     return y
```

Puis on l'utilise comme clé de tri :

```
res = sorted([(2,2), (1,5), (6,2), (4,7)], key = ordonnee)
```

C7 Algorithme glouton

3. Exemple résolu : problème du sac à dos

Application

En supposant que les objets d'un problème de sac à dos soient représentés par des tuples (nom, valeur, poids) :

```
sac = [("Hamburger", 180, 0.3), ("Marteau", 2050, 4.1), ...]
```

Utiliser `sorted` pour classer ces objets par rapport valeur/poids décroissant.

Exercice

En supposant les objets déjà triés, écrire une fonction `glouton` qui prend en argument une liste d'objets ainsi qu'un poids maximal et renvoie la valeur maximale du sac en utilisant la stratégie gloutonne.