

Rappel / Révision :

ex problème de sac à dos

	1	2	3	4	5	6	7
P:	3	8	2	5	4	6	3
W:	30	64	14	35	16	18	6

$$P = 15$$

w_i/p_i : 10 8 7 7 4 3 2

Calcul majorant et minorant du projet optimal.

Majorant :

- contrainte de poids :

$$3 + 11 + 13 + 15$$

1	2	3	4
---	---	---	---

215 de l'objet

$$30 + 64 + 14 + 14 = 122$$

Pour $P = 17$:

majorant :

3	11	13	17
1	2	3	7

4/5 de l'objet 4

$$30 + 64 + 14 + 28 = 136$$

minorant :

3	11	13	17
1	2	3	5

$$30 + 64 + 14 + 16 = 124$$

Pour $P = 13$

on a une

solution optimale

$$30 + 64 + 14 = 108$$

maj

3	11	13
1	2	3

$$3 + 11 + 13$$

1	2	3	
---	---	---	--

$$30 + 64 + 14 = 108$$

Programme linéaire :

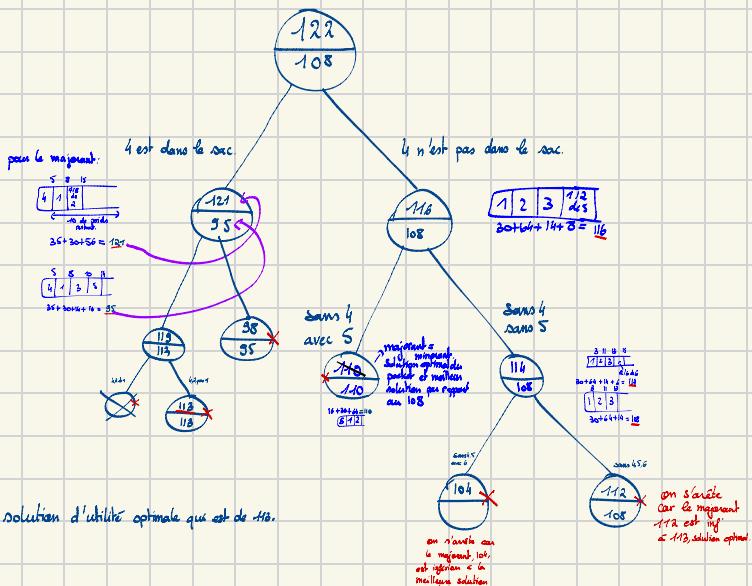
$$\begin{cases} \text{Max } \sum_{i=1}^n w_i x_i \\ \sum p_i x_i \leq P \\ x_i \in \{0,1\} \end{cases}$$

Méthode arborescente.

on a choisi 108 au début car c'était une solution possible sans dégit fractionnaire

$$f(X_0) = -108 \cancel{+ 112} \underline{-112}$$

qd on arrête une branche c'est qd le Majorant ↗ à la meilleure solution renconter



Cours:

en vert, c'est par rapport à un autre noeud.

en rouge, c'est les différences pour la minimisation.

Un problème de minimisation

fin

$$\begin{aligned} f(x) \\ \{x \in D \end{aligned}$$

Une méthode arborescente nécessite de mettre en place les définitions suivantes:

- Nœud S de l'arbre sera associé à un ensemble de solutions $D(S)$

$S \rightarrow$ associé à des objets choisis $C \rightarrow C(S)$

rejetés $R \rightarrow R(S)$

$D(S)$ l'ensemble des nœuds qui contiennent tous les objets de $C(S)$ et aucun objet de $R(S)$

- La racine est associée à toutes les solutions possibles.

$$C(R) = R(R) = \emptyset$$

- Définir comment se fait la séparation d'un nœud S

→ Comment sont constitutifs ses fils S_1, \dots, S_k de sorte

$$\text{que } D(S) \subseteq D(S_1) \cup \dots \cup D(S_k)$$

Séparation d'un noeud S :

'choisir un objet i , $i \in C(S)$
 $i \notin R(S)$

'on crée deux fils: S_g , S_d

$$C(S_g) = C(S) \cup \{i\}$$

$$R(S_g) = R(S)$$

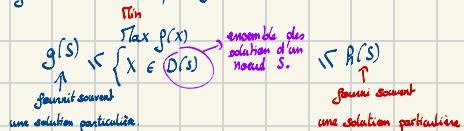
$$C(S_d) = C(S)$$

$$R(S_d) = R(S) \cup \{i\}$$

chaque noeud S peut être évalué:

$f(S) \rightarrow$ évaluation par excès (majoren)

$g(S) \rightarrow$ évaluation par défaut (minoren)



l'énumération de la meilleure solution rencontrée x_0

soit une solution d'un noeud où $D(S) = \{x\}$

soit une solution fourni par une évaluation par défaut

Un noeud S peut être tronqué/junié x

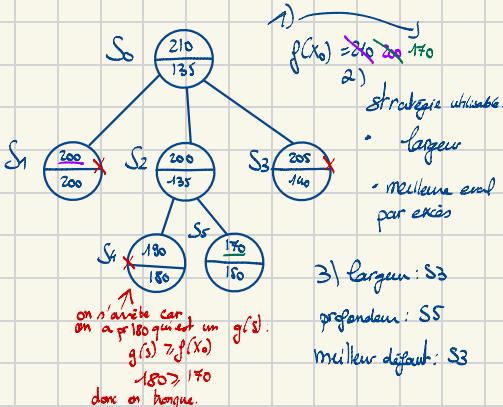
parce que $g(S) = f(S)$

ou $f(S) \leq f(x_0)$

$g(S) > f(x_0)$

Un noeud S peut avoir X fils. Dans la méthode du sac à dos c'est deux fils par noeuds.

Exercice:



Exercice 2.1 Etat d'une arborescence

Considérons une arborescence valide pour un problème de minimisation, en supposant qu'en chaque noeud, la fonction d'évaluation par excès est définie par une heuristique qui fournit une solution particulière associée au noeud. La numérotation des noeuds correspond à l'ordre de création de ces noeuds.

— la racine S_0 possède trois fils S_1, S_2, S_3 .

— S_2 possède deux fils S_4, S_5 .

noeuds	S_0	S_1	S_2	S_3	S_4	S_5
excès	210	200	200	205	190	170
défaut	135	200	135	140	180	150

Question 1 : Indiquer la valeur de la meilleure solution rencontrée.

Question 2 : Quelles opérations de troncature peut-on effectuer sur cette arborescence à chaque étape de sa création ?

Question 3 : Quel est le prochain sommet à séparer selon la stratégie choisie (profondeur d'abord, largeur d'abord ou meilleur d'abord) ?

Algo générique de _____ d'arborescence.

Les noeuds sont ouverts ou fermés.

[On crée la racine R et on calcule $g(R)$, $f_i(R)$ on ne veut la racine sauf si:

$$g(R) = f_i(R)$$

tant qu'il existe un noeud ouvert:

choisir S ouvert

regarder s'il peut être fermé avec les règles de truncature

sinon on le ferme et on crée les fils de S, qu'en évalue et qu'en ouvre.

↳ STTJ de $F(x_i)$

Stratégie en profondeur d'arbre

→ gestion des sommets ouverts avec une pile.



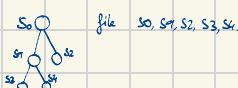
dans la pile, on aura:

un noeud qui a un des fils est bloqué.

Stratégie en largeur d'abord

→ gestion des sommets ouverts avec une file. Premier arrivé, premier servi.

Analyse étape par étape:



Stratégie meilleure (min ou max) évaluation par défaut d'abord.

→ les nœuds ouverts sont gérés par une file de priorité avec comme clé: $g(s)$

Stratégie par meilleure évaluation par excès d'abord

→ file de priorité, clé $f_i(s)$

Exercice 2.6 Sac à dos bi-dimensionnel

Des étudiants ont réalisé le projet suivant : proposer et programmer une méthode arborescente pour le problème de sac à dos bidimensionnel : chaque objet i a un poids p_i , un volume v_i et une utilité w_i . Le sac a un poids maximum P et un volume maximum V , et l'on cherche à construire un sous-ensemble d'objets de poids total inférieur ou égal à P , de volume inférieur ou égal à V qui soit d'utilité maximale.

Question 1 : Rappeler la modélisation de ce problème comme un programme linéaire en nombres entiers.

Les étudiants sont tous accordés sur le principe de séparation suivant :

Chaque noeud S de l'arbre est associé à un sous-ensemble d'objets $C(S)$ et un sous-ensemble d'objets $R(S)$. On notera $N(S)$ l'ensemble des objets qui ne sont ni dans $C(S)$ ni dans $R(S)$. Les solutions associées au noeud S sont l'ensemble des sacs qui contiennent tous les objets de $C(S)$ et aucun objet de $R(S)$. Pour séparer un noeud S , on choisit un objet i dans $N(S)$ et on construit les fils S' et S'' , définis comme suit :

$$C(S') = C(S) \cup \{i\}, R(S') = R(S), \quad C(S'') = C(S), R(S'') = R(S) \cup \{i\}$$

Exercice 2.6:

Q1) $\text{Taux } \sum_{i=1}^n x_i w_i$

$$\left\{ \begin{array}{l} \sum_{i=1}^n x_i p_i \leq P \\ \sum_{i=1}^n x_i v_i \leq V \end{array} \right.$$

$$x_i \in \{0,1\}$$

Q2) vérifier que toute solution du problème appartient à un filo.

Soit X un sac de $D(S)$, X contient tous les objets de $C(S)$ et aucun de $R(S)$

Soit i l'objet qui permet de créer les filos S' et S''

deux cas :

① si $i \in X$
 $X \in D(S')$
 (ou contient $C(S')$)

② $i \notin X$
 $X \in D(S'')$

Q3) l'algorithmique pour l'éval par excès on fait la relaxation, le domaine des solutions doit être plus grand en relaxant des contraintes.

\Rightarrow on relaxe des contraintes, on calcule par

contrainte non majorante. On prend le plus petit des majorants.

\rightarrow pour le minimum on fait comme d'hab

• $h1(S) \rightarrow$ pas une relaxation mais une heuristique donc on peut dire une éval par défaut.

• $h2(S) \rightarrow$ on relaxe la contrainte de volume, le domaine des solutions n'étend.

\hookrightarrow c'est bien une relaxation du problème initial, dc c'est une éval par échec.

$$\begin{aligned} & \text{Taux } \sum_{i=1}^n w_i x_i \\ & \left\{ \begin{array}{l} \sum_{i=1}^n p_i x_i \leq P \\ \sum_{i=1}^n v_i x_i \leq V \\ x_i = 1 \text{ si } i \in C(S) \\ x_i = 0 \text{ si } i \in R(S) \end{array} \right. \end{aligned}$$

$$\Rightarrow \begin{aligned} & \text{Taux } \sum_{i=1}^n w_i x_i \\ & \left\{ \begin{array}{l} \sum_{i=1}^n p_i x_i \leq P \\ x_i \in \{0,1\} \\ \sum_{i=1}^n v_i x_i \leq V \\ -x_i = 1 \text{ si } i \in C(S) \\ -x_i = 0 \text{ si } i \in R(S) \end{array} \right. \end{aligned}$$

• $h3(S) \rightarrow$ pas une relaxation on découpé mais on a pas de justification du problème.

\hookrightarrow l'algorithme ne fournit pas la solution optimale d'une relaxation.

• $h4(S) \rightarrow$ évaluation par excès, on ajoute une relaxation.

$$\begin{aligned} & \text{Taux } \sum_{i=1}^n w_i x_i \\ & \left\{ \begin{array}{l} \sum_{i=1}^n (p_i + v_i) x_i \leq P + V \\ 0 \leq x_i \leq 1 \\ x_i = 1 \text{ si } i \in C(S) \\ x_i = 0 \text{ si } i \in R(S) \end{array} \right. \end{aligned}$$

proposition d'éval par défaut:

- $h3$ sans le volume.

Question 2 : Justifiez le choix de ce principe de séparation dans une méthode arborescente.

Les étudiants ont proposé plusieurs fonctions d'évaluation par excès :

— $h1(S)$: trier les objets de $N(S)$ par $\frac{w_i}{p_i}$ décroissant et les placer un à un dans le sac contenant déjà les objets de $C(S)$: si l'objet ne rentre pas dans l'une des dimensions (poids ou volume) on ne le met pas et on passe au suivant. $h1(S)$ est alors égal à la somme des utilités des objets dans le sac.

— $h2(S)$: trier les objets de $N(S)$ par $\frac{w_i}{p_i + v_i}$ décroissant et les placer un à un dans le sac contenant déjà les objets de $C(S)$ sans tenir compte du volume. Si un objet dépasse, on le découpe de sorte qu'il rentre dans les deux dimensions et on s'arrête. $h2(S)$ est alors égal à la somme des utilités des objets dans le sac et à la fraction éventuelle d'utilité de l'objet découpé.

— $h3(S)$: trier les objets de $N(S)$ par $\frac{w_i}{p_i + v_i}$ décroissant et les placer un à un dans le sac contenant déjà les objets de $C(S)$: si un objet ne rentre pas dans l'une des dimensions, on le découpe de sorte qu'il rentre dans les deux dimensions et on s'arrête. $h3(S)$ est alors égal à la somme des utilités des objets dans le sac et à la fraction éventuelle d'utilité de l'objet découpé.

— $h4(S)$: trier les objets de $N(S)$ par $\frac{w_i}{p_i + v_i}$ décroissant et considérer un problème une dimension où chaque objet i a un poids $p_i + v_i$ et un sac avec un poids maximum $P + V$. Ajouter les objets un à un dans le sac contenant déjà les objets de $C(S)$ jusqu'à ce que l'un d'eux dépasse (par rapport à ces poids fictifs). Dans ce cas, le découper pour qu'il rentre dans le sac à une dimension. $h4(S)$ est alors égal à la somme des utilités des objets dans le sac et à la fraction éventuelle d'utilité de l'objet découpé.

Q3)

①	②	③	④	⑤	
p_i	3	4	2	6	3
v_i	1	5	3	2	3
w_i	40	81	40	56	36
$\frac{w_i}{p_i}$	13	20	20	9	12
$\frac{w_i}{p_i + v_i}$	10	16	13	28	12
$\frac{w_i}{p_i + v_i}$	10	9	8	7	6

$P = 10$

$V = 9$

 $\rightarrow h_1$

4ds	6ds	9ds
2	3	1

$40 + 81 + 40 = 161$

f_1	161
f_2	96
f_3	161
f_4	237
f_5	168

le plus grand des minorants
et le plus petit des majorants

par défaut
↳ i dans
son entière
arionne
t'as
6 avec
partition de:

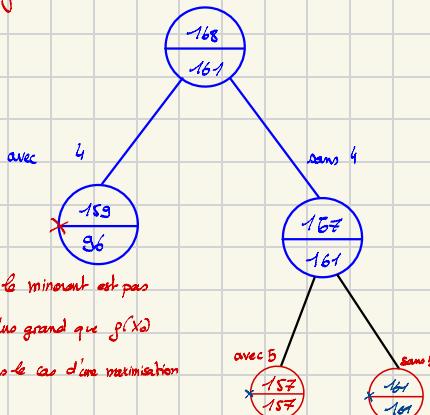
relaxation ↲
car p're
contraints qu'on va
combiner de la
f'reg. linéaire et la
fonction objectif.
 $\frac{168}{161}$

question 4 : déjà faites okip

question 5 :

choisir parmi les minorants et majorants trouvés, deux valeurs, puis faire un arbre dessus.

$f(X_0) = 161$



Puisque on a plus de nœuds ouverts,
on en conclue que la solution optimale
est $f(X_0) = 161$.

h_1 : éval par défaut.
 h_1' comme h_1 par rapport au volume
 $h_1' \rightarrow$ si $w_i > v_i$ puis placement dans le sac non dépassé.

Question 2 : Justifiez le choix de ce principe de séparation dans une méthode arborescente.

Les étudiants ont proposé plusieurs fonctions d'évaluation par excès :
 $h_1(S)$: trier les objets de $N(S)$ par $\frac{w_i}{p_i}$ décroissant et les placer un à un dans le sac contenant déjà les objets de $C(S)$: si l'objet ne rentre pas dans l'une des dimensions (poids ou volume) on ne le met pas et on passe au suivant. $h_1(S)$ est alors égal à la somme des utilités des objets dans le sac.

$h_2(S)$: trier les objets de $N(S)$ par $\frac{w_i}{p_i + v_i}$ décroissant et les placer un à un dans le sac contenant déjà les objets de $C(S)$ sans tenir compte du volume. Si un objet dépasse, on le découpe de sorte qu'il rentre en poids, et on s'arrête. $h_2(S)$ est alors égal à la somme des utilités des objets dans le sac et à la fraction éventuelle d'utilité de l'objet découpé.

$h_3(S)$: trier les objets de $N(S)$ par $\frac{w_i}{p_i + v_i}$ décroissant et les placer un à un dans le sac contenant déjà les objets de $C(S)$: si un objet ne rentre pas dans l'une des dimensions, on le découpe de sorte qu'il rentre dans les deux dimensions et on s'arrête. $h_3(S)$ est alors égal à la somme des utilités des objets dans le sac et à la fraction éventuelle d'utilité de l'objet découpé.

$h_4(S)$: trier les objets de $N(S)$ par $\frac{w_i}{p_i + v_i}$ décroissant et considérer un problème une dimension où chaque objet a un poids $p_i + v_i$ et un sac avec un poids maximum $P + V$. Ajouter les objets un à un dans le sac contenant déjà les objets de $C(S)$ jusqu'à ce que l'un d'eux dépasse (par rapport à ce poids fictif). Dans ce cas, le découper pour qu'il rentre dans le sac à une dimension. $h_4(S)$ est alors égal à la somme des utilités des objets dans le sac et à la fraction éventuelle d'utilité de l'objet découpé.

Question 5 : Appliquer votre méthode arborescente à un problème à 5 objets de votre choix. Vous préciserez les évaluations de chaque noeud et indiquerez, le cas échéant vos décisions de troncature.

 R^1 : on classe les objets par $\frac{w_i}{p_i + v_i}$.

on parcourt les objets et on les met dans le sac s'ils rentrent en poids et volume.

 R^2 : $\frac{w_i}{p_i + v_i}$ et on remplit un sac virtuel de poids $P + V$
avec des objets de poids $p_i + v_i$.

sans 4
159 : sans j'aide pas.
161 :

1	2	3	1/2
4	5	6	1/2

avec 5 :

 $min - h_3'$: $3 \quad 6 \quad 10$

3	6	10
5	1	2

3	6	10
5	1	2

$3 + 6 + 10 = 157$

$5 + 1 + 2 = 8$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10 + 8 = 157$

$3 + 6 + 10$

Pour le projet.

- programmer l'algorithme de l'arbre
- programmer un générateur de jeu de domino pour les tester sur l'algorithme.
aléatoire