Exercice 2 – Découpe d'une corde

Le magasin "A la vieille campeuse" achète des cordes d'escalade de longueur n et les découpe (soigneusement) en cordes plus petites pour les vendre à ses clients. On souhaite déterminer un découpage optimal pour maximiser le revenu, sachant que les prix de ventes p_i d'une corde de i mètres sont donnés. Par exemple, supposons qu'on dispose d'une corde de $n = 10$ mètres, avec les prix de ventes indiqués dans le tableau suivant :

i	1	2	3	4	5	6	7	8	9	10
p_i	1	5	8	9	10	17	17	20	24	26

Q 2.1 Dans un premier temps, on suppose que chaque découpe est gratuite. On définit la *densité* d'une corde de longueur i comme étant le rapport p_i/i, c'est-à-dire son prix au mètre. Une stratégie gloutonne naturelle consisterait à découper une première corde de longueur i dont la densité soit maximale. On continuerait ensuite en appliquant la même stratégie sur la portion de corde restante de longueur $n - i$. Montrer que cette stratégie ne conduit pas toujours à un découpage optimal.

d_i	1	2,5	2,7	2,25	2	2,83	2,42	2,5	2,7	2,6
	1	2	3	4	5	6	7	8	9	10

Sol gloutonne : 6 + 3 + 1 → 26

Pourtant, 6 + 2 + 2 rapporte 27

Q 2.2 On cherche à établir un algorithme de programmation dynamique pour déterminer le revenu de ventes maximal que l'on peut obtenir pour une corde, en supposant toujours que chaque découpe est gratuite.

a. Quelle est la sous-structure optimale dont on a besoin pour résoudre ce problème ?

$r(i)$: revenu max qu'on peut tirer d'une corde de i mètres

b. Caractériser (par une équation) cette sous-structure optimale. $r(0) = 0$

$$r(i) = \max\{r(i-j) + p_j : 1 \leq j \leq i\}$$

c. En déduire le revenu de ventes maximal qu'on peut tirer d'une corde.

$r(n)$

$$r(i) = \max\{r(i-j) + p_j : 1 \le j \le i\}$$

i	0	1	2	3	4	5	6	7	8	9	10
p_i	0	1	5	8	9	10	17	17	20	24	26
$r(i)$	0	1	5	8	10	13	17	18	22	25	27
$t(i)$	-	1	2	3	2	2	6	1	2	3	2

OPT

découpe optimale : 2 + 2 + 6

d. Ecrire un algorithme de programmation dynamique pour calculer ce revenu.

```
r[0] ← 0
pour i de 1 à n faire
    r[i] ← -∞
    pour j de 1 à i faire
        r[i] ← max{r[i], r[i-j] + p[j]}
retourner r[n]
```

e. Quelle est la complexité de cet algorithme ?

Le nombre de + réalisées par l'algorithme est :

$$\sum_{i=1}^{n} i = \frac{n(n+2)}{2} \in \Theta(n^2)$$

f. Appliquer l'algorithme à l'exemple.

voir tableau précédent !

Q 2.3 Modifier l'algorithme précédent afin qu'il renvoie également la découpe optimale. Appliquer à l'exemple.

Soit $t(i)$ la longueur du 1er morceau à découper dans une corde de longueur i