Exercice 2 :

1. longueur max 17 ; Glouton ;

1	2
2	13
3	9
4	3
5	3
6	3
7	5
8	14

⇝ placer les mots

La programmation $| B = -16 + 17 = 1$
dynamique est une $| B = -17 + 17 = 0$
des bases $| B = -8 + 17 = 8$
algorithmiques $| B = -14 + 17 = 3$

↳ coût $1^3 + 0^3 + 8^3 + 3^3 = 540$

Meilleure solution :

La programmation $| B = 1$

dynamique est $| B = 4$

une des bases $| B = 4$

algorithmiques $| B = 3$

↳ coût $1^3 + 4^3 + 4^3 + 3^3 = 156 < 540$

2. Entrée : $w_1, w_2 \dots w_n$

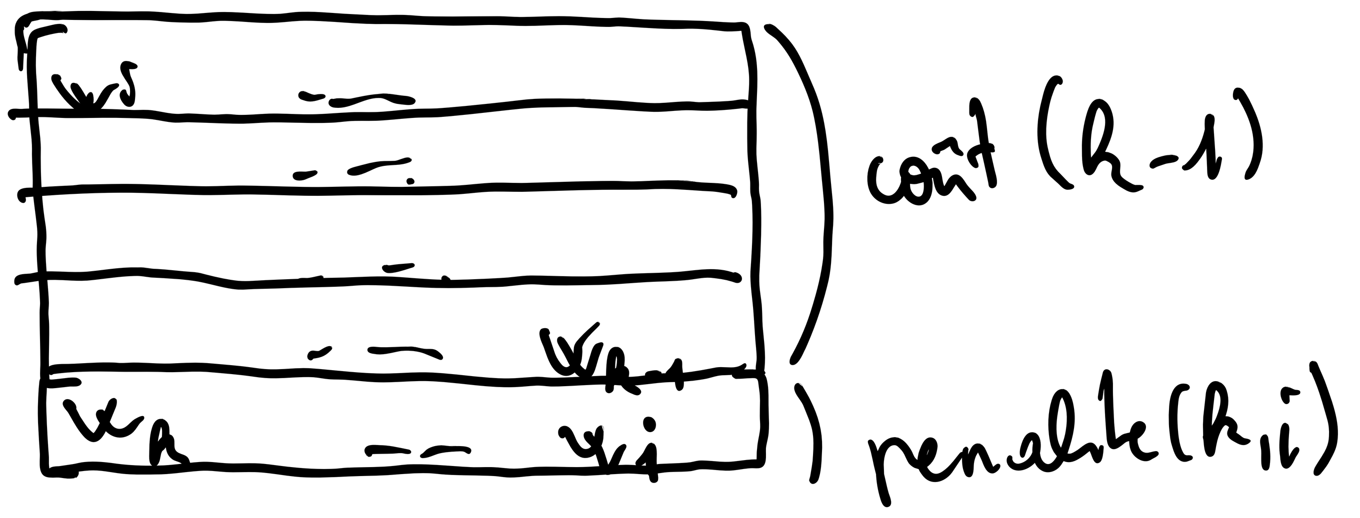
nb de blanc en fin de ligne : $M - (l_i + l_{i+1} + \dots + l_j) - (j - i)$ si ceci est ≥ 0

pénalité (i, j) =

$[M - (l_i + \dots + l_j) - (j - i)]^3$ si ceci est ≥ 0

$+\infty$ sinon (condition pour pas avoir de lignes négatives).

3. $w_1, w_2, \dots w_i$



cas général : $(i \geq 1)$;

$$\text{coût}(i) = \min_{1 \leq k \leq i} (\text{coût}(k-1) + \text{penalité}(k, i))$$

cas de base : $i = \emptyset$

coût $(0) = 0$

Autre méthodes

<u>cas de base</u> $i \geq 1$

coût $(1) =$ pénalité $(1,1)$

<u>cas général</u> : $i \geq 2$

$$\text{coût}(i) = \min\left[\text{penalité}(1,i), \min_{2 \leq k \leq i} \left(\text{coût}(k-1) + \text{penalité}(k,i)\right)\right]$$

Rq : au lieu de $1 \leq k \leq i \rightsquigarrow \max(i - \frac{M+1}{2}, 1) \leq k \leq i$ (dans la 1ère méthode)

4. Optimisation :

$M = 17 \quad n = 10^7$

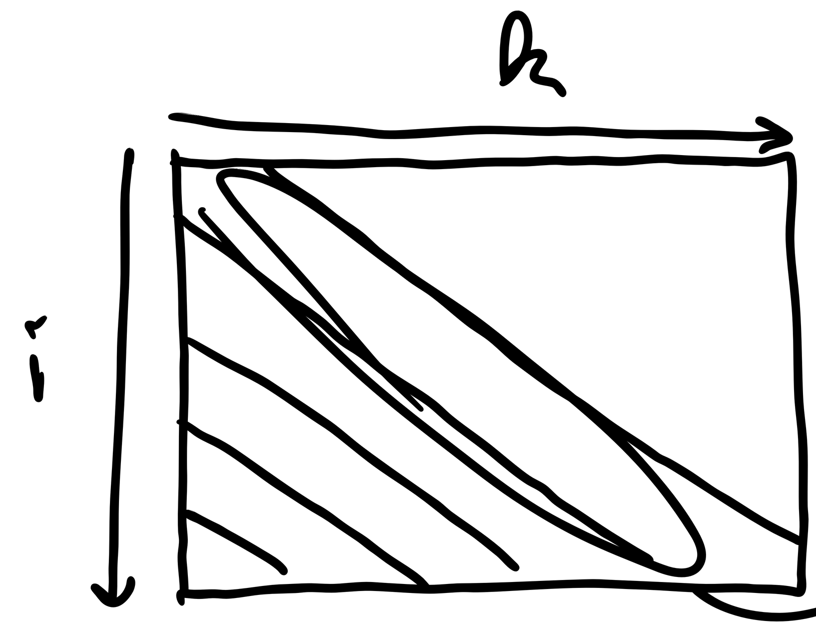
chaque mot à au moins 1 caractère donc il y a au plus $\frac{M+1}{2}$ mots sur une ligne.

- 1 tableau penalité$[k,i]$ pour $k \geq i - \frac{M+1}{2}$
- 1 tableau coût $[i]$

remplir la case coût$[i]$ prend temps $O(M)$

On a $n+1$ cases à remplir.

Le tableau coût[ ] se calcule en temps $O(nM)$ (si la penalité est déjà rempli).



commence par la diagonale puis

$O(n \times M)$ cases à remplir le calcul pour remplir une case prend $O(M)$

$\Rightarrow O(n \times M^2)$

```
cout : tableau d'indices 0...n
cout[0] ← 0
pour i allant de 1 à n
  . m ← +∞
  . pour k allant de ⌊i - (M+1)/2⌋ à i
      si cout[k-1] + penalti[k,i]
  cout[i] ← m
retourner cout[n]
```

```
penalti
pour i allant de 1 à n
   pour k allant de max(1, i - (M+1)/2) à i
     . s ← 0
     . pour x allant de k à i
         s ← s + l[x]
     . penalti[k,i] ←
         (M - s + j - i)^3
```

Exercice 3 :

n=8

J1 enlève 1 à 7 allumettes

si J1 enlève

7, 6, 5, 4, ou 3 allumettes

alors J2 enlève le reste et gagne

$\boxed{1}$ → J(7,2) $\boxed{2}$ → J(6,4)

n \ m	1	2	3	4	5	6	7	8
1	V	V	V	V	V	V	V	V
2	F	V	V	V	V	V	V	V
3	F	F	V					
4				V			V	
5		⋮			V			
6		⋮				V		
7		⋮					V	
8	TODO:							V

J1 choisit la stratégie tq ça conduise à une configuration perdante pour J2

```
gagne(n,m)
// n allumettes, on peut en retirer jusqu'à m
. si m ≥ n alors retourner "vrai"
. sinon :
    Si ∃ i, 1 ≤ i ≤ m alors retourner "vrai"
    tq gagne(n-i, 2i) = "faux" sinon retourner "faux"
```

```
pour n allant de 1 à N
  pour m allant de 1 à 2N
    // calcul de gagne[n,m]
    si m ≥ n alors gagne[n,m] ← vrai
    sinon :
      res ← faux
      pour i allant de 1 à m
```

| | | | res ← res ou NOT (gagne [n-i, 2i])
| | | gagne [n, m] ← res
retourner gagne [N, N-1]

alors retourner "vrai"
sinon retourner "faux"

Kpolynomial : taille des appels
" exponentiel : si rec car recalculer val