TD2 : Exo 1, Exo 2 (1-2), Exo 3, Exo 4, Exo 6

Exercice 1 : (1) Idée possible : générer un compteur binaire de 0..0 à 1..1 de taille n (= taille X)
Pour chaque valeur $x = x^0 \dots x^{n-1}$, calculer la somme

$$\sum_{x^i = 1} T[i]$$

vérifier si on obtient t ou pas

ex : [-5, 2, 7, -3, 2, -6]

0
1
1 0
1 1
⋮

Toutes les combinaisons avec le compteur binaire

On utilisera la fonction suivant(x) pour passer à l'élément suivant

```
sous_somme(T, n) :
    x ← [0, 0, ..., 0] de taille n = |T| ;
    Tant que (1) :
        S ← 0 ;
        Pour i = 0 à n-1 :
            S ← S + x[i] × T[i] ;
            # On somme les T[i]
              avec x[i]
```

```
si S = T :
|    retourner x;
si x = [1,...,1] :
|    retourner "Pas de solutions";

x ← Suivant(x)
```

Complexité : While fait au max toutes les combinaisons binaires avant de dire "aucune solutions", donc au maximum 2^n opérations et dans chaque while, une boucle de $n-1$ opérations ET un appel vers suivant de complexité $O(n)$

Donc $2^n \times (n + \cancel{n}) \rightsquigarrow 2^n \times n \rightsquigarrow O(n2^n)$

(3) L'idée est d'itérer en enlevant des cases, on choisit deux cas, l'élément est solution (on soustrait à t), ou pas (t reste le même)

$$T_{[0,5[}[1,3,7,11,2]\ ,\ t = 12$$

on prend T[4] — On prend pas T[4]

$T_{[0,4[}[1,3,7,11], t = 10$ ($\uparrow$ 12-2) — $T_{[0,4[}[1,3,7,11], t = 12$

$T_{[0,3[}\ [1,3,7],\ t=-1$ $\qquad$ $T_{[0,3[}\ [1,3,7],\ t=10$

... ... $\qquad$ ... ...

$[1,3,7,11]\ t=10$

~~$[1,3,7]\ t=-1$~~ $\qquad$ $[1,3,7]\ t=10$

$[1,3]\ t=3$

$[1]\ t=0$ ✓

algo_backtrack(T, n, t) :

avec T le tableau, t la solution cherchée

si $t=0$:
  retourner vrai
si $n=0$ ou $t<0$
  retourner faux

retourner algo_backtrack$(T, n-1, t-T[n-1])$
  ou algo_backtrack$(T, n-1, t)$

Complexité : Pas de Master Theorem (car $n-1$).

$$n \leqslant 2 \times ((n-1) + O(1))$$

On a (cf TD1) $\rightarrow C(n) = O(2^n)$

Exercice 2 :

1. Même [illegible] que la [illegible] binaire :

(1) Même principe que le compteur binaire :

On commence par $u = (\underbrace{0,0\dots0}_{n})$ et on termine par $u = (\underbrace{k-1,\dots,k-1}_{n})$

Suivant (u) :

```
i ← 0;
Tant que (u[i] = k-1):
|    u[1] ← 0;
|    i ← i+1;
si i = n:
|    retourner fin;
u[i] ← u[i]+1

retourner u;
```

exec:

↓

0,0,0

1,0,0

2,0,0

0,1,0

(2,2,2,1,0)

↓

(0,0,0,2,0)

Complexité : au plus n tour de boucles Tant que

Nombre d'éléments à énumérer : k^n

(k possibilités par case, n cases)

(2) Exemple avec $n=3$ et $k=3$

(0 0 0)

(0, 0, 1) on va modifier notre compteur

(0, 0, 2)

(0, 1, 1)

(0, 1, 2)

(0, 2, 2)

⋮

On part de $u=(0,0\dots0)$ jusqu'à $u=(k-1,\dots,k-1)$

suivant_croissant (u) :

```
i ← k;
Tant que (u[i] = k-1):
    i ← i-1;
    si i < 0:
        retourner "fin"

u[i] ← u[i] + 1;
Pour j = i+1 à n-1:
    u[j] ← u[i];

retourner u;
```

Complexité en $O(n)$

Exercice 3: ①

Si il y a moins de 10 lettres dans les mots choisis, on rajoute des lettres en plus inutiles (X, Y, Z ...).
Aux 10 lettres apparaissant on associe les valeurs (0, 1, 2, ... 9) et on effectue toutes les permutations possibles des valeurs

② On veut $O \rightsquigarrow \emptyset$

$A \rightsquigarrow 1$

$S \rightsquigarrow 2$

$\vdots$

```
Dico(a, b, c):   # Pour chaque lettre on associe
```

d ← dictionnaire vide une valeur
pour m ∈ {a, b, c} : # m, un des 3 mots
    pour l ∈ m : # l, lettre de m
        si l ∉ d :
            d[l] ← i ;
            i ← i + 1 ;

retourner d ;

③ une partition de {0..9}

i	0	1	2	3	4	5	6	7	8	9
$\pi(i)$	4	2	0	8	3	5	1	7	9	6

O A S I (au-dessus des colonnes 0, 1, 2, 3)

↓

Pour OASIS, O → 4, A → 2, S → 0, I → 8
donc 42080

valeur(m, d, π) :
    v ← 0 ;
    Pour l ∈ m (de gauche à droite)
        v ← v × 10 + π(d[l]) ;

    retourner v ;

v=0

O A S I
4
4 0 +2
4 2 0 +0
4 2 0 8
4 2 0 8 0

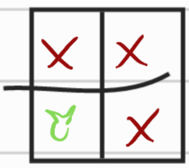
④
criptarithme (a, b, c) :
    π ← (0, 1, 2, ..., 9) ;
    d ← dico(a, b, c) ;

```
Tant que (1) :
    Si valeur(a, d, π) + valeur(b, d, π) = valeur(c, d, π) :
        r ← dictionnaire vide
        Pour l ∈ d :
            r ← (l, π(d[l]));
        retourner r  # une solution

    Si π = dernière permutation : (9, 8, 7..., 0) :
        retourner "Pas de solution"

    π ← permutation_suivante(π)
```

Exercice 4 :

(1) (1) 

(2) On a au plus 1 reine par ligne et par colonne, et au plus n reines

(3) Il y a n lignes, n reines et au plus une reine par ligne donc on a exactement une reine par ligne

(2) $Q[i]=j$ s'il y a plusieurs reines en (i, j)

Si $i \neq i'$ alors $Q[i] \neq Q[i']$ sinon on aurait 2 reines sur la meme ligne