$i \leftarrow 5$ $i \leftarrow i+5$

Examen 1 2018 Q11

$a-2$ a $a-1$

Quelle est la sortie de l'algorithme suivant sur l'entrée $L = (-3, 5, 12, -4, 3, 8, -1, -6, 4)$:

$\leftarrow a$

i j k

algo11

entrée : L *liste de valeurs*

sortie : *???*

- $a \longleftarrow$ **taille**(L)
- $R \longleftarrow \emptyset$ // Liste vide
- **Si** $a \geq 3$
  - **Pour** i de 1 à $a-2$
    - **Pour** j de $i+1$ à $a-1$
      - **Pour** k de $j+1$ à a
        - **Si** $L[i] + L[j] + L[k] = 0$
          - $R \longleftarrow (i, j, k)$
          - Sortir R
- **Sortir :** R

~~**A]**~~ $\emptyset$ (liste vide) $\simeq 10$

B] $(2, 4, 7)$ 50%

~~**C]**~~ $(5, -4, -1)$ $\simeq 15$

D] $(1, 7, 9)$ < 10

~~**E]**~~ $(1, 7, 9, 2, 4, 7)$ bcp (trop)

~~**F]**~~ $(7, 4, 2)$ 4-7

i / $L[i]$

$R \leftarrow (\ldots)$ / $R \leftarrow R \oplus (\ldots)$

Examen 1 2018 Q11

Quelle est la sortie de l'algorithme suivant sur l'entrée $L = (-3, 5, 12, -4, 3, 8, -1, -6, 4)$:

n = taille de L

algo11

entrée : L *liste de valeurs*

sortie : *???*

$a \longleftarrow$ **taille**(L) $\mathcal{O}(n)$ $\Theta(n)$

$R \longleftarrow \emptyset$ // Liste vide $\Theta(1)$

Si $a \geq 3$ $\Theta(1)$

    Pour i de 1 à $a-2$ $\Theta(1)$

        Pour j de $i+1$ à $a-1$

            Pour k de $j+1$ à a

                Si $L[i]+L[j]+L[k] = 0$ $\Theta(1)$

                    $R \longleftarrow (i,j,k)$ $\Theta(1)$ $\Big\}\ \Theta(1)$

Sortir : R

A] $\emptyset$ (liste vide)

*B] $(2, 4, 7)$

C] $(5, -4, -1)$

D] $(1, 7, 9)$

E] $(1, 7, 9, 2, 4, 7)$

F] $(7, 4, 2)$

Combien de fois?

$\Theta(n^3)$

$\Theta(n^3) \times \Theta(1)$

$\in \Theta(n^3)$

Examen 1 2018 Q12

$$\sum_{i=1}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} op \in \Theta(n^3)$$

Si l'on note n la taille de la liste L, quelle est la complexité de l'algorithme de la question précédente ?

A] $\Theta(n^3)$
B] $\Theta(2^n)$
C] $\Theta(n^2)$
D] $\Theta(n^4)$

où n est la taille L

Leçon I.2 (conception d'algorithmes) – Points clés

- approche descendante : **DÉCOMPOSEZ** le problème
- algorithmes récursifs :
  - ramener le problème à la résolution du *même* problème sur moins de données
  - penser à la condition d'arrêt
- programmation dynamique :
  stocker/mémoriser au lieu de recalculer
- problèmes de plus courts chemins
  complexité polynomimale : $\Theta(n^3)$, $\Theta(n^2)$ ou $\Theta(n)$ en fonction de la nature du problème (nombre de villes de départ/d'arrivée fixées)

Leçon I.2 (conception d'algorithmes) – Difficultés connues (1/2)

- **concevoir un algorithme récursif :**
  1. essayer de voir « le schéma qui se répète »
  2. pensez à la/aux condition(s) d'arrêt(s)
  3. si 1 est trop difficile : essayez, de partir d'à peine plus compliqué que 2 (conditions d'arrêt), et d'avancer « d'un cran de plus » pour arriver aux conditions d'arrêt pendant quelques pas (pour avoir une idée de 1)
- **méthodes usuelles pour avancer « d'un cran de plus » :**
  - enlever un
  - couper en deux

Leçon I.2 (conception d'algorithmes) – Difficultés connues (2/2)

- **calculer la complexité d'algorithmes** (en particulier récursifs)

  Vous avez trois moyens « pratiques » :

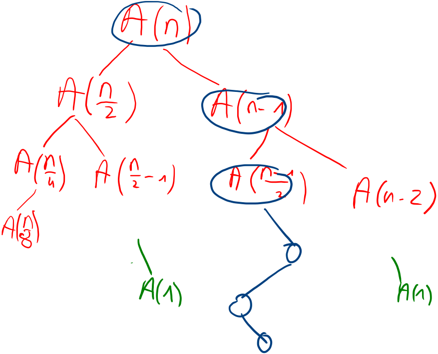
  1. ***Compter* les instructions**
     l'inconvénient est que cela conduit à une équation sur la complexité (fonction), qu'il est parfois (souvent ?) difficile de résoudre

  2. ***dessiner* le graphe des appels** depuis la taille *n* jusqu'à toutes les terminaisons et compter alors le nombre d'arcs (pour le parcourt du pire cas)

     (revoir l'exemple du cours des appels du calcul récursif des coefficients du binôme)

  3. utiliser la méthode « **incrémenter et compter** » :
     de combien augmente la complexité si j'augmente la taille de l'entrée de 1 ?

     ☞ cela donne une estimation de la dérivée de la complexité (fonction)



$A(n)$
$A(\frac{n}{2})$
$A(n-1)$
$A(\frac{n}{4})$
$A(\frac{n}{2}-1)$
$A(\frac{n-1}{2})$
$A(n-2)$
$A(\frac{n}{8})$
$A(1)$
$A(1)$

Leçon I.2 (conception d'algorithmes) – Dichotomie

Écrire complètement l'algorithme de recherche par dichotomie dans une liste *ordonnée* :

- spécifier le problème
- « couper la liste en deux » : comment faire ?
  - ☞ je vous impose de passer 2 paramètres supplémentaires en entrée : indice de début de recherche et indice de fin de recherche (inclus)

entrée : L, x, a, b
sortie : vrai ou faux ($x \in L$?)

L = (3, 7, 8, 12, 25, 32)

algo(L, 27, 1, 6)

a = 2

nouveau b = 4

b = 5

Recherche

Entrée : L, x, a, b

Sortie : vrai / faux

$L[1]$

$n \leftarrow$ taille(L)

Si $a > b$ ou L est vide

Sortir FAUX

Si $a = b$

sortir $(L[a] == x)$ bool

pivot = $\left\lfloor \frac{b+a}{2} \right\rfloor$

Si $x < L[\text{pivot}]$

Sortir Recherche(L, x, a, pivot)

Sinon sortir Recherche(L, x, pivot, b)

est-ce correct ?