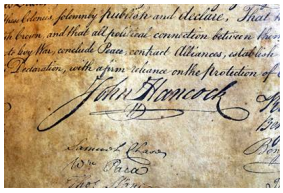
IFT2015 A17 :: Examen intra[1]

[1] The English translation follows.

L'EXAMEN vaut 100 points, et vous pouvez avoir jusqu'à 10 points de boni additionnels.

- ⋆ Aucune documentation n'est permise.
- ⋆ Décrivez vos algorithmes en pseudocode ou en Java(-esque).
- ⋆ Répondez à toutes les questions dans les cahiers d'examen.

Exo	points	boni
F0	1	
F1	$7 \times 2 = 14$	
F2	$3 + 12 = 15$	
F3	20	
F4	$10 + 10 + 10 = 30$	10
F5	20	
$\sum$	100	10



F0 Votre nom (1 point)

▶ Mettez votre nom et code permanent sur tous les cahiers soumis.

F1 Ordres de croissance (14 points)

▶Donnez une caractérisation asymptotique sur l'échelle standard (n^a, $\log_a n$, a^n et leurs produits) des fonctions ci-bas. (Utilisez Θ, O ou une expansion asymptotique avec $\sim$). $\lg = \log_2$. Chaque réponse vaut 2 points, et il n'est pas nécessaire de les justifier.

a $f(n) = \sum_{i=0}^{n-1} \lfloor i/2 \rfloor$

b $f(n) = \ln(n!)$

c $f(n) = \sum_{i=1}^{n} n/i$

d $$f(n) = \begin{cases} 1 & \{n = 0\} \\ f(n-1) + 2n - 1 & \{n > 0\} \end{cases}$$

e $$f(n) = \begin{cases} n & \{n = 0, 1\} \\ \big(f(n-1)\big)^2 & \{n > 1\} \end{cases}$$

f $$f(n) = \begin{cases} 0 & \{n = 0\} \\ f(n-1) + \lg(n) & \{n > 0\} \end{cases}$$

g $f(n) = \lg(1 + 2n + 3n^3)$

F2 Types abstraits (15 points)

i. Œufs de coucou (3 points) ▶ Identifiez les trois types abstraits sur la liste suivante :

dictionnaire, liste chaînée, Willem de Kooning, pile, queue (file FIFO), tableau.

ii. Interface (12 points) ▶ Spécifiez et décrivez[2] les opérations principales pour les trois types abstraits identifiés.

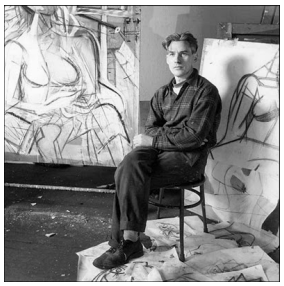


FIG. 1: Le peintre De Kooning dans son studio [1950].

[2] sans entrer dans les détails d'une implémentation quelconque !

F3 Chemins dans un DAG (20 points)

On a un graphe orienté acyclique, representé par listes d'adjacence.
▶Donner un algorithme[3] qui compte le nombre de chemins entre deux nœuds s, t fournis à l'entrée, en un temps linéaire.

[3] ***Indice:*** Modifiez le tri topologique.

F4 Renverser une liste chaînée (30+10 points)

a. (10 points) ▶ Donner un algorithme **récursif** appellé deleteLast qui supprime et retourne le dernier nœud sur une liste simplement chaînée. L'algorithme prend le nœud à la tête comme entrée.

b. (10 points) ▶ Donner un algorirhme **récursif** appellée reverse pour renverser une liste chaînée selon la récurrence : enlever le dernier nœud u, renverser la liste restante, l'ajouter après u et retourner u (la nouvelle tête de la liste).

c. (10 points) ▶ Analyser le temps de calcul et usage de mémoire de la solution en **b.**

d. (10 points boni) ▶ Donner un algorithme qui renverse la liste simplement chaînée en un temps linéaire[4].

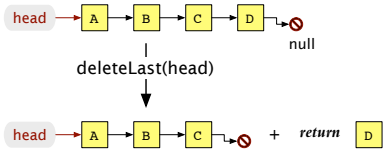


FIG. 2: Suppression du dernier nœud.

[4] ***Indice:*** On peut renverser les liens en un seul parcours. Ou bien, il est possible de renverser la liste par récursion : couper à la moitié (sous-listes L_1, L_2), et employer la récurrence $\mathsf{reverse}(L_1 \rightarrow L_2) = \mathsf{reverse}(L_2) \rightarrow \mathsf{reverse}(L_1)$. (On peut identifier le nœud au centre à l'aide de deux curseurs en parcourant la liste : un pointeur avance par un nœud à la fois, l'autre avance par deux — si ce dernier arrive à la fin, alors le premier est au milieu.)

F5 Quadtree (20 points)

On a un arbre quaternaire avec n nœuds internes dont chacun possède 4 enfants.

▶ Quel est le nombre de nœuds externes ? Justifiez la réponse.

▶ Donner une borne inférieure sur la hauteur de l'arbre, avec preuve[5].

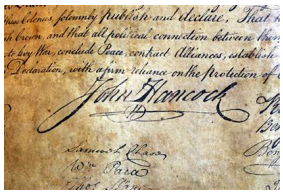
[5] ***Indice:*** Écrivez n_d pour le nombre de nœuds internes à profondeur d, et cherchez une borne sur le nombre des nœuds internes $n = \sum_d n_d$. Notez qu'il ne suffit pas de juste montrer un arbre complet sans démontrer qu'il est un cas extreme.

Mid-term examination

THE EXAMEN is worth 100 points, and you can collect up to 10 additional bonus points.

- ⋆ No documentation is allowed.
- ⋆ Write your algorithms in (Java-style) pseudocode.
- ⋆ You may write your answers in English or in French.
- ⋆ Answer each question in the exam booklet.

Exercise	points	bonus
E0	1	
E1	$7 \times 2 = 14$	
E2	$3 + 12 = 15$	
E3	20	
E4	$10 + 10 + 10 = 30$	10
E5	20	
$\sum$	100	10



E0 Your name (1 point)

▶ Write your name and *code permanent* on each booklet that you submit.

E1 Growth orders (14 points)

▶Characterize then growth order of the following functions on the standard scale (n^a, $\log_a n$, a^n and their products). (Use Θ, O or an asymptotic expansion with $\sim$). $\lg = \log_2$. Every answer is worth 2 points. No justification is necessary.

a $f(n) = \sum_{i=0}^{n-1} \lfloor i/2 \rfloor$

b $f(n) = \ln(n!)$

c $f(n) = \sum_{i=1}^{n} n/i$

d $$f(n) = \begin{cases} 1 & \{n = 0\} \\ f(n-1) + 2n - 1 & \{n > 0\} \end{cases}$$

e $$f(n) = \begin{cases} n & \{n = 0, 1\} \\ \big(f(n-1)\big)^2 & \{n > 1\} \end{cases}$$

f $$f(n) = \begin{cases} 0 & \{n = 0\} \\ f(n-1) + \lg(n) & \{n > 0\} \end{cases}$$

g $f(n) = \lg(1 + 2n + 3n^3)$

E2 Abstract types (15 points)

i. Cuckoo's egg (3 points) There are three abstract data types in the following list : ▶ identify which ones.

dictionary, linked list, Willem de Kooning, stack, queue, array.

ii. Interface (12 points) ▶ Specify and describe[6] the elementary operations for the three abstract data types you identified.

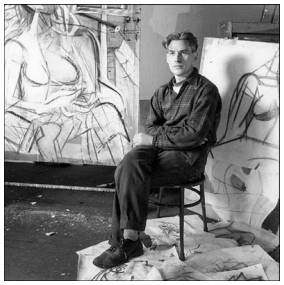


FIG. 3: The painter De Kooning in his studio [1950].

[6] without entering into the details of a particular implementation !

E3 Paths in a DAG (20 points)

We have a directed acyclic graph represented by lists of adjacence. ▶Give an algorithm[7] that counts the number of paths between two nodes s, t given as input, in linear time.

[7] ***Hint:*** Modify topological sort.

E4 Inverting a linked list (30+10 points)

a. (10 points) ▶ Give a **recursive** algorithm called deleteLast that deletes and returns the last node on a single-linked list. The algorithm's input is the head node.

b. (10 points) ▶ Give a **recursive** algorithm called reverse that reverses a linked list by using the following recursion : delete the last node u, reverse the remaining list, append it after u, and return u (the new head node).

c. (10 points) ▶ Analyze the running time and memory usage of the algorithm in **b.**

d. (10 bonus points) ▶ Give an algorithm that reverses a single-linked list in linear time[8].

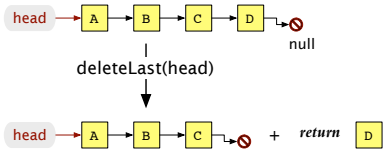


FIG. 4: Deletion of the last node.

[8] ***Hint:*** It is possible to invert the linkes in a single list traversal. Or it is possible to employ divide-and-conquer : cut into half lists in the middle (sublists L_1, L_2), and use the recursion $\mathsf{reverse}(L_1 \rightarrow L_2) = \mathsf{reverse}(L_2) \rightarrow \mathsf{reverse}(L_1)$. (You can identify the middle by using two cursors in a traversal : one pointer advances by one node, the other advances by two nodes at a time — when this latter arives to the end, the first one is in the middle.)

E5 Quadtree (20 points)

We have a quaternary tree with n internal nodes, each of them with 4 children.

▶ What is the number of external nodes ? Justify your answer.

▶ Give a lower bound on the tree height, with proof[9].

[9] ***Indice:*** Write n_d for the number of internal nodes at level d, and infer a bound on the number of internal nodes $n = \sum_d n_d$. Note that it is not enough to just show an extremal tree without proving that it is indeed an extreme case.