


14. IFT2015 Structures de données: Liste détaillée de sujets¹

¹ Detailed List of Subjects for the Final Examination — English translation starts on Page 11

F0 Introduction

LE BUT DE CE DOCUMENT est de définir les compétences et connaissances requises dans le cours IFT2015 à l'examen final. L'examen constitue également la deuxième partie de l'examen pré-doctorale en structures de données (sous le sigle IFT6002).

- ♦ J'ai marqué les connaissances par niveau de maîtrise correspondant :
 - ★ satisfaisant (50% des exercices au final),
 - ★★ bon (25%),
 - ★★★ excellent (25 – 40%).
- ★ Les notes marginales sont des références aux ouvrages suivants
 - S* Sedgewick, R. *Algorithmes en Java*, 3^e édition (2004)
 - SW* Sedgewick, R. et K. Wayne. *Algorithms*, 4^e édition (2011)
 - CLR* Cormen, T., E. L. Leiserson, R. L. Rivest, et C. Stein. *Algorithmique*, 3^e édition.
- ★ Les notes de cours, présentations, et des liens vers ressources en-ligne sont affichés sur le site <http://ift2015a17.wordpress.com/>.

 Aucune documentation ne sera permise à l'examen final.



F1 Principes d'analyse d'algorithmes

Références

- ▷ Sedgewick chapitre 2
- ▷ Sedgewick & Wayne §1.4² ² <http://algs4.cs.princeton.edu/14analysis/>
- ▷ Cormen, Leiserson, Rivest & Stein chapitres 1–3
- ▷ Notes sur les fondations : [handout01-recursion.pdf](#). Diapos: [prez01-recursion.pdf](#).
- ▷ Notes sur l'analyse d'algorithmes : [handout05-analysis.pdf](#). Diapos: [prez05-analysis.pdf](#).

Sujets

- ★ Principes de base : pire cas, meilleur cas, moyen cas. S§2.1,2,2.2,7; CLR 1
- ★ Algorithmes simples : plus grand diviseur commun, recherche dichotomique (binaire), recherche séquentielle S§12.3–12.5
- ★ Validation expérimentale de temps de calcul S§2.3
- ★ Croissances communes : constante, logarithmique, linéaire, linéarithmique ($n \log n$), quadratique, polynomiale, exponentielle. Factorielle($n!$)
- ★★ Approximation de Stirling³, nombres Fibonacci⁴, nombres harmoniques⁵. ³ $n! \sim \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$; $\lg(n!) \sim n \lg n$
- ★ Notation asymptotique⁶ : définitions de grand $O(f)$ ⁴ $F_0 = 0; F_1 = 1; F_n = F_{n-1} + F_{n-2} \{n > 1\}$
- ★★ Petit $o(f)$, $\Theta(f)$ et $\Omega(f)$. Application de la définition pour démontrer $f = O(g)$ ou $f = o(g)$. ⁵ $H_n = \sum_{i=1}^n 1/i = \ln n + \gamma + o(1)$
- ★★ Asymptotiques exactes $f \sim g$. Expressions avec $O()$ ou $o()$, règles d'arithmétique : $O(f) + O(g)$, $O(f) \cdot O(g)$. Relations avec la limite ⁶ $W_{(fr)}$:comparaison asymptotique
S§2.4; CLR 3

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c > 0 \quad \Rightarrow \quad f(n) = O(g(n));$$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \quad \Leftrightarrow \quad f(n) = o(g(n));$$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 1 \quad \Leftrightarrow \quad f(n) \sim g(n)$$

- ★★ Détermination du temps de calcul et d'usage de mémoire pour algorithmes (itératifs) simples, et pour algorithmes récursifs (comme expression récursive). CLR 2
- ★★ Récurrences communes. S§2.5,2.6

$$\begin{array}{ll} f(n) = f(n-1) + O(1) & f(n) = O(n); \\ f(n) = f(n/2) + O(1) & f(n) = O(\log n); \\ f(n) = 2f(n/2) + O(1) & f(n) = O(n); \\ f(n) = 2f(n/2) + O(n) & f(n) = O(n \log n); \end{array}$$

- ★★★ Preuve par induction pour récurrences asymptotiques.
- ★★ Notion de temps amorti.
- ★★★ Preuves de résultats sur le coût amorti d'opérations. Principe d'analyse crédit/débit⁷. CLR§17.4

⁷ $W_{(en)}$:accounting method

F2 Structures élémentaires et types abstraits

Références

- ▷ Sedgewick chapitres 3 et 4
- ▷ Sedgewick & Wayne §1.1⁸, §1.2⁹, §1.3¹⁰
- ▷ Cormen, Leiserson, Rivest & Stein §10.1, §10.2
- ▷ Notes sur les types abstraits : [handout02-tad.pdf](#). Diapos: [prez02-tad.pdf](#).
- ▷ Notes sur les tableaux : [handout03-tableaux.pdf](#). Diapos: [prez03-tableaux.pdf](#).
- ▷ Notes sur les listes : [handout04-chaining.pdf](#). Diapos: [prez04-chaining.pdf](#).

Sujets

- ★ Blocs de construction pour programmes Java. S§3.1;SW§1.1
- ★ Notions de type abstrait, interface, implémentation, client. S§4.1;SW§1.2
- ★ Types abstraits : pile et queue/file FIFO. S§4.2,4.7
- ★ Listes chaînées¹¹. Variations : listes circulaires, doublement chaînées. ¹¹ W_(fr):liste chaînée
- Sentinelles¹² pour la tête et/ou la queue. Manipulation d'éléments sur la S§3.3,3.4;CLR§10.2
- liste, insertion et suppression. Parcours d'une liste. ¹² W_(en):sentinel
- ★ Tableaux¹³. ¹³ W_(fr):tableau
- ★ Implantations de pile et de queue par tableau ou liste chaînée. Efficacité d'implantations différentes (temps de calcul pour les opérations S§3.2
- standards). Débordement. S§4.4,4.5,4.7;SW§1.3;CLR§10.1

F3 Arbres

Références

- ▷ Sedgewick §4.3, §5.4–5.7
- ▷ Cormen, Leiserson, Rivest & Stein §10.4.
- ▷ Notes sur les listes et les arbres : [handout04-chaining.pdf](#). Diapos: [prez04-chaining.pdf](#).

Sujets

- ★ Algorithmes récursifs. Diviser pour régner. S§5.1,5.2
- ★ Terminologie pour structures arborescentes : arbre k -aire, hauteur, niveau, profondeur. Implémentation d'un arbre. S§5.4
CLR§10.4
- ★ Propriétés d'arbres binaires (relations entre le nombre de nœuds internes et externes ou la hauteur). S§5.5
- ★ Parcours d'un arbre : préfixe/préordre, infixe/dans l'ordre, post-fixe/postordre, ordre de niveau. S§5.6
- ★ Algorithmes récursifs sur les arbres : calcul de taille, hauteur ou profondeur de sous-arbres. S§5.7
- ★ Arbre syntaxique. Conversions d'expressions arithmétiques : notations infixe, postfixe et préfixe. S§4.3

F4 Algorithmes sur graphes

Références

- ▷ Sedgewick §3.7, §5.8
- ▷ Sedgewick & Wayne §4.1¹⁴, §4.2¹⁵ ¹⁴ <http://algs4.cs.princeton.edu/41graph/>
¹⁵ <http://algs4.cs.princeton.edu/42digraph/>
- ▷ Cormen, Leiserson, Rivest & Stein chapitre 22
- ▷ Notes sur les graphes : [handout06-graphes.pdf](#). Diapos: [prez06-graphes.pdf](#).

Sujets

- ★ Représentation d'un graphe : matrice d'adjacence et listes d'adjacence¹⁶. S§3.7;SW§4.1;CLR§22.1
¹⁶ [W_{\(en\)}:adjacency list](#)
- ★ Parcours d'un graphe par profondeur et par largeur. S§5.8;SW§4.1;CLR§22.2,§22.3
- ★★ Applications de parcours : composantes connexes, bipartition, tri topologique, plus courts chemins (à partir d'une source). SW§4.2;CLR§22.4

F5 Appartenance-union

Références

- ▷ Sedgewick §1.2–1.3
- ▷ Sedgewick & Wayne §1.5¹⁷ ¹⁷ <http://algs4.cs.princeton.edu/15uf/>
- ▷ Cormen, Leiserson, Rivest & Stein §21.1–21.4
- ▷ Notes sur Union-Find : [handout08-unionfind.pdf](#). Diapos: [prez08-extreme.pdf](#).

Sujets

- ★ Problème de connexité, TA et opérations d'appartenance-union. S§1.2
- ★ Structure Union-Find¹⁸. Techniques algorithmiques : union-par-rang/union-par-taille, compression de chemin. ¹⁸ $W_{(tr)}$:union-find
S§1.3;SW§1.5
- ★★ Coût amorti d'opérations : $O(\alpha(m, n))$ pour Union-Find avec union équilibrée et compression de chemin ; fonction d'Ackermann¹⁹ et son inverse. ¹⁹ $W_{(tr)}$:fonction d'Ackermann
- ★★★ Croissance incalculable, fonction de Radó²⁰ ²⁰ $W_{(tr)}$:fonction de Radó et le castor affairé

F6 File de priorité

Références

- ▷ Sedgewick §9.1–9.6
- ▷ Sedgewick & Wayne §2.4²¹
- ▷ Cormen, Leiserson, Rivest & Stein chapitre 6
- ▷ Notes sur la file de priorité : [handout09-heap.pdf](#). Diapos: [prez09-pq.pdf](#).
Diapos: [prez09-heapsort.pdf](#).

²¹ <http://algs4.cs.princeton.edu/24pq/>

Sujets

- ★ Type abstrait de file de priorité²² min-tas/max-tas : opérations insert, deleteMin ou deleteMax. Implantations par tableau ou liste chaînée.
- ★ Arbre en ordre de tas²³. Tas binaire²⁴, sa représentation dans un tableau.
- ★★ Manipulation du tas : nager/swim et couler/sink (heapisation montante et descendante).
- ★ heapify (établissement de l'ordre de tas dans un tableau) ; tri par tas²⁵, son temps de calcul et usage de mémoire.
- ★★★ Tas d -aire²⁶.

S§9.1,9.5

²² W_(en):priority queue

²³ W_(fr):tas

S§9.2,9.3;SW§2.4

²⁴ W_(fr):tas binaire

S§9.4

²⁵ W_(fr):tri par tas

²⁶ W_(en): d -ary heap

F7 Tableau de hachage

Références

- ▷ Sedgewick chapitre 14
 - ▷ Sedgewick & Wayne §3.4²⁷, §3.5²⁸.
 - ▷ Cormen, Leiserson, Rivest & Stein §11.1–11.4
 - ▷ Notes sur le hachage : [handout10-hashing.pdf](#). Diapos: [prez10-hashing.pdf](#).
- ²⁷ <http://algs4.cs.princeton.edu/34hash/>
²⁸ <http://algs4.cs.princeton.edu/35applications/>

Sujets

- ★ Type abstrait de dictionnaire/table de symboles/*map*. S§12.1,12.2
- ★ Notions de base pour tableaux de hachage²⁹ : facteur de charge/remplissage, collisions. S§14.1 ; CLR§11.2
²⁹ W_(H):table de hachage
- ★ Fonctions de hachage : méthodes de la division et de la multiplication. CLR§11.3
- ★ Résolution de collisions par chaînage séparé. Coût moyen des opérations de l'interface (table de symboles) en fonction de la facteur de charge. S§14.2
- ★ Addressage ouvert : notion de sondage/test. Procédures de recherche et d'insertion avec addressage ouvert. Suppression paresseuse et impatiente ; hachage dynamique (*rehashing*). Sondage linéaire, grappe forte. Double hachage. S§14.3–14.6 ; CLR§11.4
- ★ Performance des opérations (pire cas, moyen cas)
- ★ ★ ★ Coût moyen des opérations de l'interface avec sondage linéaire et double hachage en fonction de la facteur de charge.

F8 Méthodes de tri

Références

- ▷ Sedgewick §6.1–6.4, §6.6, §6.9; chapitres 7, 8 et 10.
- ▷ Sedgewick & Wayne §2.1³⁰, §2.2³¹, §2.3³², §5.1³³, §6.2.
- ▷ Cormen, Leiserson, Rivest & Stein chapitres 2 et 7, §8.1–8.3.
- ▷ Notes sur les tris : [handout11-sorting.pdf](#). Diapos: [prez11-sorting.pdf](#).
Diapos: [prez13-strings.pdf](#).

³⁰ <http://algs4.cs.princeton.edu/21elementary/>

³¹ <http://algs4.cs.princeton.edu/22mergesort/>

³² <http://algs4.cs.princeton.edu/23quicksort/>

³³ <http://algs4.cs.princeton.edu/51radix>

Sujets

- ★ Terminologie : tri interne et externe.
- ★ Tri par sélection³⁴ et tri par insertion³⁵.
- ★ Performances des tris élémentaires (pire cas, meilleur cas, cas moyen).
- ★★ Tri de liste chaînée.
- ★ Fusion de 2 tableaux triés.
- ★ Tri par fusion³⁶ (descendant), sa performance.
- ★ Tri rapide³⁷ : algorithme de base. Améliorations : partition par la médiane-de-trois, petits sous-fichiers.
- ★ Performances du tri rapide (pire cas, meilleur cas, cas moyen)
- ★★ Génération d'une permutation aléatoire
- ★★★ Preuve de la performance moyenne $O(n \log n)$ du tri rapide.
- ★★ Preuve de la borne inférieure $\lg(n!) \sim n \lg n$ sur le nombre de comparaisons au pire pour trier
- ★★ Tri en temps linéaire : clés binaires, tri radix LSD et MSD

S§6.1

S§6.3,6.4; SW§2.1

³⁴ $W_{(n)}(\text{tri par sélection})$

³⁵ $W_{(n)}(\text{tri par insertion})$

S§6.6

S§3.4,6.9

S§8.1

S§8.3,8.4; SW§2.2

³⁶ $W_{(n)}(\text{tri fusion})$

³⁷ $W_{(n)}(\text{tri rapide})$

S§7.1,7.4,7.5; SW§2.3

S§7.2,7.3; SW§2.3; CLR 7

SW§2.2; CLR§8.1

SW§5.1;S§10.2,10.3,10.5;CLR§8.2,8.3

F9 *Arbre binaire de recherche**Références*

- ▷ Sedgewick chapitre 12, §13.1–13.4, §16.1–16.3
- ▷ Sedgewick & Wayne §3.1³⁸, §3.2³⁹, §3.3⁴⁰, §6.2 (B-trees)
- ▷ Cormen, Leiserson, Rivest & Stein chapitres 12, 13, et 18
- ▷ Diapos: [prez12-abr.pdf](#). Diapos: [prez13-strings.pdf](#).

³⁸ <http://algs4.cs.princeton.edu/31elementary/>

³⁹ <http://algs4.cs.princeton.edu/32bst/>

⁴⁰ <http://algs4.cs.princeton.edu/33balanced>

Sujets

- ★ Arbre binaire de recherche⁴¹. Procédures fondamentales sur un ABR : recherche, insertion, suppression. Recherche de minimum ou maximum, successeur ou prédécesseur. S§12.6–12.9
- ★ Performance moyenne des opérations sur un ABR standard avec clés aléatoires. S§13.1
- ★ Notion d'un ABR équilibré. Maintenance d'équilibre : rotations simples et doubles.
- ★★ ABR *splay*. Principe de déploiement (*splaying*). Coût amorti des opérations de l'interface (table de symboles). S§13.2
- ★★ Règles de déploiement.
- ★ ABR rouge et noir⁴². Définition par rang (hauteur noire) ou coloriage ; équivalence des deux définitions. Coût des opérations dans le pire cas. S§13.4
- ★★ Hauteur maximale de l'arbre rouge et noir.
- ★★ Techniques de base avec l'ABR rouge et noir : promotion/rétrogradation, changement de couleur, rotation. Déroulement général d'une insertion ou suppression. CLR§13.1,13.2
- ★★ Déroulement détaillé de l'insertion et de la suppression dans l'arbre rouge et noir. CLR§13.3,13.4
- ★★ Arbre 2-3-4⁴³, et son équivalence avec l'arbre rouge et noir. Techniques de base sur les arbres 2-3-4 : décalage et découpage, leur relation aux rotations et promotions. S§13.3
- ★★ Arbre AVL, sa hauteur maximale ; insertion dans l'arbre AVL.
- ★★ Recherche externe, recherche séquentielle indexée, arbre B S§16.1–16.3; CLR 18

⁴¹ W_(fr):ABR

⁴² W_(fr):arbre bicolore

⁴³ W_(en):2-3-4 tree



◀ français

English ▶

E0 Introduction

THIS DOCUMENT defines the skills and knowledge for the final examination in IFT2015, which is also the second part of the *examen pré-doctoral* in data structures (as IFT6002).

♦ I marked the subjects by levels of mastery :

★ satisfactory (50% at the final),

★★ good (25%),

★★★ excellent (25 – 40%).

★ The margin notes refer to the following books :

S Sedgewick, R. *Algorithms in Java*, Parts 1–4, 3rd edition (2003)

SW Sedgewick, R. and K. Wayne. *Algorithms*, 4th edition (2011)

CLR Cormen, T., E. L. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*, 3rd edition.

★ The class notes and links to Wikipedia articles are available on the webpage <http://ift2015a17.wordpress.com/>.

📖 This is a closed-book exam.



E1 Principles of algorithm analysis

References

- ▷ Sedgewick chapter 2
- ▷ Sedgewick & Wayne §1.4⁴⁴ ⁴⁴ <http://algs4.cs.princeton.edu/14analysis/>
- ▷ Cormen, Leiserson, Rivest & Stein chapters 1–3
- ▷ Notes on the foundations: [handout01-recursion.pdf](#). Slides: [prez01-recursion.pdf](#).
- ▷ Notes on algorithm analysis: [handout05-analysis.pdf](#). Slides: [prez05-analysis.pdf](#).

Topics

- ★ Basic principles : worst case, best case, average case. S§2.1,2.2,2.7; CLR 1
- ★ Experimental validation of running time
- ★ Simple algorithms : sequential search, binary search, greatest common divisor S§2.3
- ★ Common growth orders : constant, logarithmic, linear, linearithmic ($n \log n$), quadratic, polynomial, exponential. Factorial ($n!$),
- ★★ Stirling's formula⁴⁵, Fibonacci numbers⁴⁶, harmonic numbers⁴⁷. ⁴⁵ $n! \sim \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$; $\lg(n!) \sim n \lg n$
- ★ Asymptotic notation⁴⁸ : definitions of big-Oh $O(f)$ ⁴⁶ $F_n = F_{n-1} + F_{n-2}$
- ★★ small-oh $o(f)$, $\Theta(f)$, and $\Omega(f)$. Using the definitions to prove $f = O(g)$ or $f = o(g)$. ⁴⁷ $H_n = \sum_{i=1}^n 1/i = \ln n + \gamma + o(1)$
- ★★ Arithmetic expressions involving asymptotics, rules : $O(f) + O(g)$, $O(f) \cdot O(g)$. Connections to \lim ⁴⁸ $W_{(en)}$:big-O notation
S§2.4;CLR 3

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c > 0 \quad \Rightarrow \quad f(n) = O(g(n));$$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \quad \Leftrightarrow \quad f(n) = o(g(n));$$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 1 \quad \Leftrightarrow \quad f(n) \sim g(n)$$

- ★★ Determination of space and time complexity for simple (iterative) algorithms, and for recursive algorithms (as a recursive expression). CLR 2
- ★ Basic recurrences. S§2.5,2.6

$$f(n) = f(n-1) + O(1) \quad f(n) = O(n);$$

$$f(n) = f(n/2) + O(1) \quad f(n) = O(\log n);$$

$$f(n) = 2f(n/2) + O(1) \quad f(n) = O(n);$$

$$f(n) = 2f(n/2) + O(n) \quad f(n) = O(n \log n);$$

- ★★★ Proof by induction for asymptotic recurrences.
- ★★ Notion of amortized cost.
- ★★★ Proving amortized cost. Credit/debit method. CLR §17.4

E2 Elementary structures and abstract data types

References

- ▷ Sedgewick chapters 3 et 4
- ▷ Sedgewick & Wayne §1.1⁴⁹, §1.2⁵⁰, §1.3⁵¹
- ▷ Cormen, Leiserson, Rivest & Stein §10.1, §10.2
- ▷ Notes on abstract data types: [handout02-tad.pdf](#). Slides: [prez02-tad.pdf](#).
- ▷ Notes on tables: [handout03-tableaux.pdf](#). Slides: [prez03-tableaux.pdf](#).
- ▷ Notes on linked lists: [handout04-chaining.pdf](#). Slides: [prez04-chaining.pdf](#).

⁴⁹ <http://algs4.cs.princeton.edu/11model/>

⁵⁰ <http://algs4.cs.princeton.edu/12oop/>

⁵¹ <http://algs4.cs.princeton.edu/13stacks/>

Topics

- ★ Java building blocks. S§3.1;SW§1.1
- ★ Concept of an abstract data type, interface, implementation, client. S§4.1;SW§1.2;
- ★ Abstract types for stacks and queues. S§4.2,4.7
- ★ Linked lists⁵². Variations : circular, doubly-linked lists. Sentinels⁵³ for head and/or tail. Manipulation of elements, insertion and deletion. List traversal. ⁵² W_(en):linked list
S§3.3,3.4;CLR§10.2
- ★ Arrays⁵⁴. ⁵³ W_(en):sentinel
⁵⁴ W_(en):array
- ★ Implementations of stack and queue by tables or linked lists. Running time for standard operations in different implementations. Overflow/underflow. S§3.2
S§4.4,4.5,4.7;SW§1.3;CLR§10.1

E3 Trees

References

- ▷ Sedgewick §4.3, §5.4–5.7
- ▷ Cormen, Leiserson, Rivest & Stein §10.4.
- ▷ Notes on lists and trees: [handout04-chaining.pdf](#). Slides: [prez04-chaining.pdf](#).

Topics

- ★ Recursive algorithms. Divide-and-conquer. S§5.1.5.2
- ★ Terminology for tree structures : k -ary tree, height, level, depth. Tree implementations. S§5.4
CLR §10.4
- ★ Mathematical properties of binary trees (relationships between number of internal and external nodes, height) S§5.5
- ★ Tree traversal : preorder, inorder, postorder, level-order. S§5.6
- ★ Recursions on trees : computing the size, height, or depth of subtrees. S§5.7
- ★ Syntax tree. Conversion between arithmetic notations : infix, prefix and postfix. S§4.3

E4 Graph algorithms

References

- ▷ Sedgewick §3.7, §5.8
- ▷ Sedgewick & Wayne §4.1⁵⁵, §4.2⁵⁶
- ▷ Cormen, Leiserson, Rivest & Stein chapter 22
- ▷ Notes on graphs: [handout06-graphes.pdf](#). Slides: [prez06-graphes.pdf](#).

Topics

- ★ Graph representations by adjacency matrix and adjacency lists⁵⁷. S§3.7;SW§4.1;CLR§22.1
- ★ Depth-first and breadth-first search (DFS and BFS) in a graph ⁵⁷ [W_{\(en\)}:adjacency list](#) S§5.8;SW§4.1;CLR§22.2
- ★★ Applications of graph traversal : connected components, bipartite graph, topological sort, (single-source) shortest paths S§4.2;CLR§22.4

*E5 Union-find**References*

- ▷ Sedgewick §1.2–1.3
- ▷ Sedgewick & Wayne §1.5⁵⁸
- ▷ Cormen, Leiserson, Rivest & Stein §21.1–21.4
- ▷ Notes on Union-Find: [handout08-unionfind.pdf](#). Slides: [prez08-extreme.pdf](#).

⁵⁸ <http://algs4.cs.princeton.edu/15uf/>*Topics*

- ★ Connectivity problems, union-find operations. S§1.2
- ★ Union-Find⁵⁹ data structure. Techniques : union-by-rank/union-by-size, path compression. ⁵⁹ [W_{\(en\)}:union-find](#)
S§1.3;SW§1.5
- ★★ Amortized cost per operation : $O(\alpha(m, n))$ for Union-Find with balanced trees and path compression ; Ackermann⁶⁰ function and its inverse ⁶⁰ [W_{\(en\)}:Ackermann function](#)
- ★★★ Non-computable function, Radó's busy beaver function⁶¹ ⁶¹ [W_{\(en\)}:busy beaver function](#)

E6 Priority queue

References

- ▷ Sedgewick §9.1–9.6
- ▷ Sedgewick & Wayne §2.4⁶²
- ▷ Cormen, Leiserson, Rivest & Stein Chapter 6
- ▷ Notes on priority queue: [handout09-heap.pdf](#). Slides: [prez09-pq.pdf](#).
Slides: [prez09-heapsort.pdf](#).

⁶² <http://algs4.cs.princeton.edu/24pq/>

Topics

- ★ ADT for priority queue⁶³ : operations insert, deleteMin or deleteMax.
Implementations by table or linked list.
- ★ Heap⁶⁴ order for a tree. Binary heap⁶⁵, its representation in a table.
- ★★ Heap manipulation : swim and sink.
- ★ heapify (linear-time construction of heap order in a table) ; Heapsort⁶⁶,
its running time and memory use.
- ★★★ d -ary heap⁶⁷.

S§9.1,9.5

⁶³ W_(en):priority queue

⁶⁴ W_(en):heap

S§9.2,9.3;SW§2.4

⁶⁵ W_(en):binary heap

S§9.4

⁶⁶ W_(en):heapsort

⁶⁷ W_(en): d -ary heap

*E7 Hash table**Reference*

- ▷ Sedgewick chapter 14.
- ▷ Sedgewick & Wayne §3.4⁶⁸, §3.5⁶⁹.
- ▷ Cormen, Leiserson, Rivest & Stein §11.1–11.4
- ▷ Notes on hashing: [handout10-hashing.pdf](#). Slides: [prez10-hashing.pdf](#).

⁶⁸ <http://algs4.cs.princeton.edu/34hash/>⁶⁹ <http://algs4.cs.princeton.edu/35applications/>*Topics*

- ★ Abstract data type for dictionary = symbol table = map.
- ★ Basic notions for hash tables⁷⁰: load factor, collisions.
- ★ Hash functions: division and multiplication methods.
- ★ Collision resolution by separate chaining. Average-case performance with separate chaining as function of the load factor.
- ★ Open addressing: probe sequence. Search and insertion with open addressing. Lazy and eager deletion, dynamic hashing. Linear probing, primary clustering. Double hashing.
- ★ Average and worst-case performance of operations
- ★ ★ ★ Cost of operations with linear and quadratic probing in function of the load factor.

S§12.1,12.2

S§14.1; CLR §11.2

⁷⁰ [W_{\(en\)}:hashtable](#)

CLR §11.3

S§14.2

S§14.3–14.6; CLR §11.4

E8 Sorting algorithms

References

- ▷ Sedgewick §6.1–6.4, §6.6, §6.9; chapters 7, 8 and 10.
- ▷ Sedgewick & Wayne §2.1⁷¹, §2.2⁷², §2.3⁷³, §5.1⁷⁴, §6.2.
- ▷ Cormen, Leiserson, Rivest & Stein Chapters 2 and 7, §8.1–8.3.
- ▷ Notes on sorting algorithms: [handout11-sorting.pdf](#).
- ▷ Slides: [prez11-sorting.pdf](#). Slides: [prez13-strings.pdf](#).

⁷¹ <http://algs4.cs.princeton.edu/21elementary/>

⁷² <http://algs4.cs.princeton.edu/22mergesort/>

⁷³ <http://algs4.cs.princeton.edu/23quicksort/>

⁷⁴ <http://algs4.cs.princeton.edu/51radix>

Topics

- ★ Terminology : internal and external sort.
- ★ Insertion⁷⁵ sort and selection⁷⁶ sort.
- ★ Performance of elementary sorting algorithms (worst case, best case, average case).
- ★★ Sorting a linked list.
- ★ Merging sorted arrays.
- ★ Mergesort⁷⁷ (top-down), its performance.
- ★ Quicksort⁷⁸ : basic algorithm. Improvements : pivoting by median-of-three, small subarrays.
- ★ Performance of quicksort (worst case, best case, average case).
- ★★ Generating a random permutation
- ★★★ Proof of $O(n \log n)$ average running time for quicksort.
- ★★ Proof of the lower bound $\lg(n!) \sim n \lg n$ for the worst-case number of comparisons
- ★ Linear-time sorting : binary keys, radix sort

S§6.1

⁷⁵ W_(en):insertion sort

⁷⁶ W_(en):selection sort

S§6.3.6.4; SW§2.1

S§6.6

S§3.4.6.9

S§8.1

⁷⁷ W_(en):merge sort

S§8.3.8.4; SW§2.2

⁷⁸ W_(en):quicksort

S§7.1.7.4.7.5; SW§2.3; CLR 7

S§7.2.7.3

SW§2.2; CLR§8.1

SW§5.1;S§10.2,10.3,10.5;CLR§8.2,8.3

E9 *Binary search trees**Reference*

- ▷ Sedgewick chapters 12, §13.1, §13.3 and §13.4, §16.1–16.3
- ▷ Sedgewick & Wayne §3.1⁷⁹, §3.2⁸⁰, §3.3⁸¹, §6.2 (B-trees)
- ▷ Cormen, Leiserson, Rivest & Stein chapters 12, 13 and 18
- ▷ Diapos: [prez12-abr.pdf](#). Diapos: [prez13-strings.pdf](#).

⁷⁹ <http://algs4.cs.princeton.edu/31elementary/>

⁸⁰ <http://algs4.cs.princeton.edu/32bst/>

⁸¹ <http://algs4.cs.princeton.edu/33balanced>

Topics

- ★ Binary search tree⁸². Basic techniques : search, insertion, deletion.
Searching for minimum or maximum, successor or predecessor. 82 $W_{(en)}$:BST
S§12.6–12.9
- ★ Average performance of a standard BST with random keys. S§13.1
- ★ Notion of a balanced BST. Maintaining the balance : simple and double rotations.
- ★★ Splay trees. Principle of splaying. Amortized cost of operations. S§13.2
- ★★★ Splaying rules.
- ★ Red-black tree⁸³. Definition by rank (black height) or coloring; equivalence of the two definitions. Time complexity for operations in the worst-case. 83 $W_{(en)}$:red-black tree
S§13.4
- ★★ Maximum height of a red-black tree.
- ★★ Basic techniques for red-black trees : promotion/demotion, recoloring, rotations. General outline of insertion and deletion. CLR§13.1,13.2
- ★★★ Detailed (case-by-case) steps in insertion and deletion in a red=black tree. CLR§13.3,13.4
- ★★ 2-3-4 tree⁸⁴, its equivalence with the red-black tree. Basic techniques with 2-3-4 trees : shifting and splitting, relationship with promotions and rotations in red-black tree. 84 $W_{(en)}$:2-3-4 tree
S§13.3
- ★★★ AVL tree, definition, and bound on height ; insertion in AVL tree
- ★★ External search, indexed sequential access, B tree S§16.1–16.3; CLR 18