

Programmazione e Algoritmica

Check List

1. Sintassi e grammatiche
 - a. Definizione di grammatica
 - b. Definizione di linguaggio
 - c. Le fasi di un compilatore
 - d. BNF e sintassi di un linguaggio di programmazione
2. Linguaggio L (vedi *LinguaggioL-CheatSheet*)
 - a. BNF e Sintassi
 - b. Regole semantica statica
 - i. Espressioni
 - ii. Comandi
 - iii. Dichiarazioni
 - iv. Funzioni: dichiarazione e chiamata
 - ~~v. Funzioni ricorsive (dichiarazione e chiamata)~~
 - c. Scoping e blocchi (identificatori liberi e legati)
 - d. Record di attivazione
 - e. Regole semantica dinamica
 - i. Espressioni
 - ii. Comandi
 - iii. Dichiarazioni
 - iv. Funzioni: dichiarazione (lambda astrazione e chiusure) e chiamata
 - ~~v. Funzioni ricorsive (dichiarazione e chiamata)~~
3. Complessità degli algoritmi
 - a. Definizione formale di O , Ω , Θ , o , ω (vedi *Cormen*)
4. Algoritmi di ordinamento
 - a. Ordinamento **per confronti**
 - i. Insertion e Selection sort
 1. Dimostrazione di correttezza e complessità per Insertion (vedi *Cormen*)
 - ii. Mergesort
 1. Dimostrazione di correttezza e complessità (vedi *Cormen*)
 - iii. Quicksort (vedi *Cormen*)
 1. Dimostrazione di correttezza
 2. Complessità al caso peggiore
 3. Dimostrazione complessità al caso medio
 - iv. Dimostrazione lower bound $\Omega(n \log n)$ problema di ordinamento per confronti
 - b. Ordinamenti **senza confronti** (vedi *Cormen* e appunti)
 - i. Counting sort
 - ii. Radix sort
5. **Divide-et-impera**: definizione e problemi
- ~~6. Il problema delle selezioni: selezione randomizzata (vedi *Cormen*)~~
7. Equazioni di ricorrenza e loro risoluzione (vedi *Cormen*)

- a. Metodo iterativo
 - b. Risoluzione mediante albero
 - c. Master Theorem
 - i. Enunciato
 - ii. ~~Dimostrazione~~
- 8. Ricerca di un elemento in una collezione
 - a. Ricerca lineare (progettazione algoritmo, correttezza, limite inferiore al problema e complessità della soluzione progettata)
 - b. Ricerca binaria mediante **divide-et-impera** (progettazione algoritmo, correttezza e complessità)
- 9. Heap (vedi *Cormen*)
 - a. Proprietà strutturale e sulle informazioni (di massimo e di minimo)
 - b. Costruire uno heap (correttezza e complessità)
 - c. Inserimento di un nodo e estrazione della radice
 - d. Heapsort (correttezza e complessità)
- 10. Tabelle e funzioni hash (vedi *Cormen*)
 - a. Gestione collisioni
 - i. Chaining (liste di trabocco)
 - ii. Probing-Open hash (lineare, quadratico, doppio hash)
 - iii. Costi e complessità (dimostrazioni al caso medio, almeno l'idea intuitiva)
- 11. Alberi binari
 - a. Definizione, e altezza nel caso peggiore e ottimo
 - b. Visite: simmetrica, anticipata e posticipata
- 12. Alberi Binari di Ricerca (vedi *Cormen*)
 - a. Definizione, e altezza nel caso peggiore e ottimo
 - b. Interrogazioni (ricerca, Min, Max, Successore, Predecessore) e operazioni di modifica (inserimento, cancellazione) e loro costi
- 13. 2-3 Alberi (dispensa di Pino Italiano, su Classroom alla Lezione 46 - 2022)
 - a. Definizione, e altezza nel caso peggiore e ottimo
 - b. Operazioni di ricerca, inserimento e cancellazione e loro costi
- 14. Programmazione dinamica
 - a. Longest Common Subsequence
 - b. Edit Distance
 - c. Zaino
- 15. Greedy (zaino frazionario)
- 16. Grafi
 - a. BFS (vedi *Cormen*, con dimostrazione di: Lemma 22.1, 22.2, e Teorema 22.5, almeno l'idea intuitiva)
 - b. DFS (vedi *Cormen*, con dimostrazione dei Teoremi 22.7, 22.8 e 22.10, almeno l'idea intuitiva)
 - c. Topological sort (vedi *Cormen*, con dimostrazione del Lemma 22.11 e del Teorema 22.12)
 - d. Cammini minimi (Dijkstra, Bellman-Ford - da appunti, no dimostrazioni)
- 17. P - NP (vedi appunti)
 - a. Definizioni di P, NP, NP-arduo, NP-completo
 - b. Esempio di riduzione 3SAT