

IN110 Algoritmi e Strutture Dati

A.A. 2023/2024

Prof. Marco Liverani

1. Problemi ed algoritmi

Introduzione alle caratteristiche del calcolatore ed al rapporto programmatore/esecutore; compiti ed abilità del programmatore; principali caratteristiche ed abilità dell'esecutore, operazioni di base (logiche, aritmetiche e di confronto). Modelli di macchina calcolatrice: cenni sul modello di Von Neumann e sulla macchina di Turing.

Linguaggi di programmazione: linguaggi imperativi e dichiarativi. Istruzioni fondamentali di un linguaggio di programmazione procedurale generico. Algoritmi e programmi; diagrammi di flusso. Regole della programmazione strutturata, cenni sul teorema di Jacopini-Böhm; approccio *top-down* alla soluzione di un problema.

2. Il linguaggio C

Organizzazione della memoria di un calcolatore, indirizzi, parole, puntatori. Codifica binaria. Tipi di dato, strutture dati (*array*, matrici, pile, code, code di priorità, liste, alberi, grafi). Linguaggio macchina, linguaggi di alto livello; compilatori ed interpreti, compilazione ed esecuzione di un programma C in ambiente UNIX/Linux. Il linguaggio C: scopi e principali caratteristiche.

La struttura di un programma C, l'inclusione degli *header*, dichiarazione delle variabili; le librerie. Tipi di dato elementari in linguaggio C: interi, *floating point*, *double*, *char*. Operatori aritmetici, valutazione di espressioni logiche e connettori logici. Puntatori; aritmetica sui puntatori. *Array* e matrici e loro rappresentazione in memoria. Strutture dati complesse: liste, alberi, grafi; l'istruzione "**struct**". Operatore di assegnazione, operatori aritmetici in C in forma estesa e compatta. Strutture di controllo: "**if ... else ...**", "**while ...**", "**do ... while**", "**for ...**".

Funzioni: funzioni di libreria e funzioni definite dall'utente. Passaggio di parametri per valore e per indirizzo alle funzioni. Funzioni ricorsive. Funzioni di input/output: "**printf**", "**scanf**", "**fprintf**", "**fscanf**"; funzioni per la gestione della memoria: "**malloc**", "**free**", "**sizeof**"; gestione di liste di record collegati tramite puntatori.

3. Algoritmi di ordinamento

Algoritmi di ordinamento elementari: *Insertion sort*, *Selection sort*, *Bubble sort*; l'approccio "divide et impera", l'algoritmo *Quick sort*. Strutture di tipo LIFO (*Last In First Out*), pile; strutture di tipo FIFO (*First In First Out*), code; code di priorità, gli *heap*. Algoritmi ottimi per l'ordinamento: *Heap sort*, *Merge sort*.

Complessità di un algoritmo nel caso peggiore, la notazione “O grande”, analisi della complessità degli algoritmi di ordinamento.

4. Algoritmi elementari sui grafi

Definizioni principali: grafo, grafo orientato; sottografo, sottografo indotto; cammino, cammino semplice, grafo connesso, grafo fortemente connesso, grafo completo, *clique*, ciclo, grafo aciclico; alberi, foreste, *spanning tree* di un grafo.

Strutture dati per la rappresentazione di grafi mediante un calcolatore: liste di adiacenza e matrici di adiacenza. Algoritmi di visita di un grafo: visita in ampiezza (BFS), visita in profondità (DFS), ordinamento topologico di un grafo orientato aciclico. Problemi di cammino di costo minimo su un grafo, l'algoritmo di Dijkstra.

Alberi binari di ricerca: definizione e algoritmi per l'ordinamento degli elementi, la ricerca di un elemento, la ricerca del massimo e del minimo l'inserimento e la cancellazione di un vertice dell'albero.

Analisi della complessità degli algoritmi presentati, notazione “O grande” per esprimere la complessità computazionale di un algoritmo. Cenni sulle classi di complessità dei problemi P, NP, NP-completi. Il problema “P=NP”.

TESTI CONSIGLIATI

- [1] T.H. CORMEN, C.E. LEISERSON, R.L. RIVEST, C. STEIN, *Introduzione agli algoritmi e strutture dati*. McGraw–Hill, (Terza edizione, 2010).
- [2] M. LIVERANI, *Programmare in C*. Esculapio, (Seconda edizione, 2013).
- [3] A. BELLINI, A. GUIDI, *Linguaggio C - Guida alla programmazione*. McGraw-Hill, (Quarta edizione, 2009).

BIBLIOGRAFIA SUPPLEMENTARE

- [4] P. CRESCENZI, G. GAMBOSI, R. GROSSI, *Strutture dati e algoritmi*. Pearson - Addison Wesley, (2006).
- [5] H.M. DEITEL, P.J. DEITEL, *C, corso completo di programmazione*. Apogeo, (Seconda edizione, 2004).
- [6] A. KELLEY, I. POHL, *C, didattica e programmazione*. Pearson - Addison Wesley, (Quarta edizione, 2004).
- [7] M. LIVERANI, *Qual è il problema? Metodi, strategie risolutive, algoritmi*. Mimesis, (2005).
- [8] M. LIVERANI, *Appunti su alcuni argomenti ed esercizi svolti, disponibili sul sito web del corso IN110 (<http://www.mat.uniroma3.it/users/liverani/IN110>) e sul gruppo Microsoft Teams del corso.*

MODALITÀ D'ESAME

- valutazione in itinere (“esoneri”)	<input checked="" type="checkbox"/> SI	<input type="checkbox"/> NO
- esame finale	scritto <input checked="" type="checkbox"/> SI	<input type="checkbox"/> NO
	orale <input checked="" type="checkbox"/> SI	<input type="checkbox"/> NO
- altre prove di valutazione del profitto (meglio descritte sotto)	<input type="checkbox"/> SI	<input checked="" type="checkbox"/> NO

L'esame si articola su due fasi successive: una prova scritta e una prova orale. Per accedere alla prova orale è necessario superare con un voto sufficiente la prova scritta, o le due prove di esonero previste. La prova scritta consiste nella soluzione di due esercizi: viene richiesto di presentare una codifica in linguaggio C di un algoritmo che risolva il problema dato. La prova orale consiste nella discussione di alcuni tra i principali argomenti trattati durante il corso (algoritmi di ordinamento, algoritmi di visita di grafi, strategia risolutiva, analisi della complessità, esempi, diagrammi di flusso; aspetti importanti del linguaggio C: allocazione della memoria, puntatori, ecc.).