Laurea in "Scienze di Internet"
Corso di "Algoritmi e Strutture Dati"
12 Luglio 2005

1. *Tempo disponibile 180 minuti. È ammesso ritirarsi entro 90 minuti.*
2. *Sono ammessi al più 3 scritti consegnati per l'A.A. 2004/05 (Gennaio-Settembre 2005)*
3. *Non è possibile consultare appunti, libri o persone, né uscire dall'aula.*
4. *Per raggiungere la sufficienza occorrono* almeno *3 esercizi risolti senza alcun errore.*
5. *Le soluzioni degli esercizi devono:*

a. *spiegare a parole l'algoritmo usato (anche con eventuali disegni)*
b. *commentare l'eventuale procedura Pascal (dettagliando il significato delle variabili)*
c. *giustificare la correttezza e tutti i passaggi matematici*
d. *dimostrare la complessità (con equazioni di ricorrenza se necessario)*

1. Data una lista *L* di interi, si vuole modificarla lasciando un solo elemento per ogni sequenza di elementi adiacenti aventi lo stesso valore e duplicando gli altri, mantenendo lo stesso ordine che gli elementi avevano inizialmente (p.e. se l'ingresso è *L* = 7, 7, 7, 2, 5, 5, 2, 2, 8, 9 allora il risultato è *L* = 7, 2, 2, 5, 2, 8, 8, 9, 9). Si scriva una procedura Pascal efficiente *utilizzando gli operatori* per le liste visti a lezione.

2. Dato un albero binario *T* e un nodo *v* in T, si definisce *altezza minima* di *v* la minima distanza di *v* da una foglia. Si definisce *peso* di *T* la somma delle altezze minime dei suoi nodi. Si scriva una funzione Pascal di complessità ottima per determinare il peso di *T* assumendo che l'albero sia realizzato con *puntatori*.

3. Si definisca, attraverso una descrizione dettagliata, figure e codice Pascal, una struttura di dati per gestire *n* chiavi numeriche tale che:

- l'inserimento di una nuova chiave abbia complessità O(log *n*)
- la ricerca del massimo abbia complessità O(1)
- l'estrazione del massimo abbia complessità O(log *n*)

4. Un dizionario è realizzato mediante una tabella hash con liste (bidirezionali) di trabocco. Si usi la funzione hash H(*k*) = *k* mod 6 per la *k*-esima lettera dell'alfabeto italiano e si assuma l'inserimento in *testa* alle liste. Si indichi il contenuto della tabella dopo avervi inserito, nell'ordine, le chiavi: G, U, E, R, R, E, S, T, E, L, L, A, R, I. Si indichi poi il contenuto della tabella dopo avervi cancellato, nell'ordine: S, I, T, H, e indi inserito, nell'ordine: L, O, R, D, F, E, N, E, R. Si discuta la complessità di tale realizzazione della struttura di dati.

5. Si scriva la procedura Pascal *Depth-First Search (DFS)* vista a lezione. Si esegua la procedura DFS sul grafo *non orientato G*=(*N*, *A*), *N* = {1, 2, 3, 4, 5}, *A* = {[1,3], [2,4], [2,5], [3,4], [3,5]} a partire dal nodo 1, assumendo che i vettori di adiacenza siano ordinati in modo *crescente* e mostrando il contenuto dei vettori di adiacenza.

6. Dato un grafo non orientato *G*=(*N*,*A*), un sottoinsieme *S* di nodi è *coprente* se ogni nodo di *N* - *S* è adiacente ad almeno un nodo di *S*. Nel grafo dell'Esercizio 5, l'insieme {1, 5} è coprente? E l'insieme {2, 5}? E l'insieme {3, 4}? Dati *G* ed un intero *k*, si scriva

un algoritmo *non deterministico* di complessità polinomiale per trovare un sottoinsieme coprente di al più *k* nodi.