FONDAMENTI DI INFORMATICA II; FONDAMENTI DI INFORMATICA – modulo B
(tempo 2h, punti 30/30) – 15 Luglio 2010

Esercizio n. 1 (punti 5)

Dato il seguente codice in linguaggio C, dove il tipo **tree** rappresenta un albero binario di caratteri e **ord_ins** la funzione di inserimento per alberi binari di ricerca:

```
#include <stdio.h>
. . .

int m (char i, tree T)
{ if (T==NULL) return 0;
  else  if (i==T->value) return (1 + m(i,T->left));
        else
           if (i<T->value) return m(i,T->left);
           else return m(i,T->right); }

void main(void)
{ char i;
  tree T=NULL;
  scanf("%c",&i);
  do { T=ord_ins(i,T);
       scanf("%c",&i); }
  while ((i>='A') and (i<='Z'));
  scanf("%c",&i);
  printf("%d",m(i,T)); }
```

a) Si indichi cosa fa questo frammento di programma e la funzione **m** in particolare.
b) Si supponga che il ciclo **do** legga da input N caratteri compresi tra 'A' e 'Z', poi '*'. Si valuti la complessità come numero di attivazioni della funzione **m**, nel caso in cui il valore letto successivamente all'uscita dal ciclo **do** non sia presente nell'albero. Si discutano i casi migliore e peggiore e li si motivi adeguatamente.

Esercizio 2 – PROGRAMMAZIONE IN C (punti 20)

Un elenco di parole uniche, in ordine lessicografico, è contenuto nel file testo PAROLE.TXT. Il file riporta una parola per linea.

Un secondo file di testo, CAPITOLO.TXT, contiene un testo in lingua italiana, privato della punteggiatura. La stessa parola può comparire più volte in CAPITOLO.TXT.

Si chiede di scrivere un programma C che chiami almeno tre funzioni (da definire) dedicate rispettivamente a:

a) creare una lista L in memoria centrale, a partire dal file PAROLE.TXT e CAPITOLO.TXT, che memorizzi, per ciascuna parola presente sia nel primo sia nel secondo file, la parola stessa e il numero di volte (occorrenze) in cui compare in CAPITOLO.TXT. Si richiede che L sia ordinata in ordine lessicografico sul campo parola; la funzione da definire per il punto a) riceve i puntatori ai due file e restituisce il puntatore alla lista creata, più eventuali parametri a scelta;

b) a partire da L, calcoli in modo ricorsivo il numero totale di occorrenze delle parole di L; la funzione da definire per il punto b) riceve il puntatore a L e restituisce un intero, più eventuali parametri a scelta;

c) letto un intero N a terminale, a partire da L, produrre una seconda lista L2 in cui sono inserite le parole di L che hanno un numero di occorrenze maggiore di N; la funzione da definire per il punto c) riceve l'intero N, il puntatore a L e restituisce il puntatore alla lista L2 creata, più eventuali parametri a scelta.

È possibile utilizzare ***librerie C*** (ad esempio per le stringhe). Nel caso si strutturi a moduli l'applicazione riportare il codice di tutti i compnenti.

Esercizio n. 3 (punti 5)
Domanda di teoria su OOP.