Prova di Laboratorio di Algoritmi
19 Settembre 2000, turno 1

Il testo è composto da quattro parti. Per superare l'esame è necessario superare almeno la prima parte. Le altre tre possono essere risolte in un ordine qualsiasi, essendo indipendenti le une dalle altre. Ad ogni parte deve corrispondere un solo file, contenente un programma scritto in linguaggio C *standard*. È ammesso utilizzare solamente le due librerie `stdio` e `stdlib`.

Si consiglia di leggere attentamente il testo in modo da seguirne correttamente tutte le direttive.

1. (file `parte1.c`, punti 15) Realizzare il tipo di dato insiemi di interi ≥ 0, utilizzando **alberi binari di ricerca**.

   Il file `parte1.c` deve contenere le tre seguenti funzioni C:

   - `empty`: restituisce un insieme vuoto;
   - `insert`: prende un intero `i` ≥ 0 ed un insieme `s`, ed inserisce `i` in `s`;
   - `print`: stampa sullo standard output il contenuto di un insieme.

   La funzione `main` del file `parte1.c` deve realizzare (in ordine) i seguenti punti:

   - genera un insieme vuoto `s`, utilizzando la funzione `empty`;
   - inserimento nell'insieme `s` di una sequenza di interi ≥ 0 letti da un file di input (usando la `insert` di cui sopra);
   - stampa (sullo standard output) di `s` (usando la `print` di cui sopra).

   Il programma deve prevedere la lettura da tastiera del nome del file di input. Il file di input è una sequenza (anche vuota) di interi ≥ 0 (soliti separatori: blank, tabbing, new-line), terminata dal numero -1.

   Esempio:

   ```
   45 32 80 0 78 65 8 20 98 67
   - 1
   ```

2. (file `parte2.c`, punti 3) Estendere il file `parte1.c` in modo che contenga la funzione `decr` che prende un insieme e decrementa ogni elemento $\neq 0$ di una unità e restituisce l'insieme modificato.

   Funzione `main`: come quella del file `parte1.c` più

   - applicazione della funzione `decr`, per decrementare gli elementi di una unità;
   - stampa sullo standard output del nuovo insieme.

3. (file `parte3.c`, punti 5) Estendere il file `parte1.c` in modo che contenga la funzione `leaves_number` che prende un insieme `s`, rappresentato come albero binario di ricerca, e determina il numero di foglie dell'albero. Tenere presente che:

   - il numero di foglie di un albero vuoto è 0;
   - il numero di foglie di un albero non vuoto è dato dalla somma del numero di foglie del sottoalbero di sinistra con quelle del sottoalbero di destra.

   Funzione `main`: come quella del file parte1.c più

   - determinazione del numero di foglie dell'albero, usando la funzione `leaves_number`;
   - stampa sullo standard input del numero di foglie sopra determinato.

4. (file `parte4.c`, punti 7) Estendere il file `parte1.c` in modo che contenga la funzione `even_odd` che prende un insieme `s` e genera due insiemi `s1` e `s2`, il primo contenente gli elementi pari di `s` e il secondo contenente gli elementi dispari. **La funzione non deve allocare nuove celle di memoria**, ma utilizzare le celle di `s`. Deve inoltre distruggere `s`.

   Funzione `main`: come quella del file `parte1.c` più

   - generazione di `s1` ed `s2`, utilizzando la funzione `even_odd`, sopra definita;
   - stampa (sullo standard output, usando la `print`), di `s1` ed `s2`;
   - stampa (sullo standard output, usando la `print`), di `s`.