Prova di Laboratorio di Algoritmi
4 Luglio 2000, turno 2

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 successioni di interi ≥ 0, utilizzando un'implementazione **ad array**.

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

   - `empty`: prende un intero `n` e restituisce una successione vuota, utilizzando un array di dimensione `n`, allocato dinamicamente;
   - `conc_t`: prende un intero `i` ≥ 0 ed una successione `s`, ed inserisce `i` in testa ad `s`;
   - `print`: stampa sullo standard output il contenuto di una successione `s`, a partire dalla coda.

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

   - legge la dimensione `n` dell'array;
   - genera una successione vuota `s`, di dimensione `n`, utilizzando la funzione `empty`;
   - inserisce nella successione `s` una sequenza di interi ≥ 0 letti da un file di input (usando la `conc_t` 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 da `-1`.

   Esempio:

   ```
   3 6 1 34 7 1
   -1
   ```

2. (file `parte2.c`, punti 3) Estendere il file `parte1.c` in modo che contenga la funzione `my_delete` che prende un intero `i` ≥ 0 ed una successione `s` e cancella da `s` la penultima occorrenza di `i` a partire dalla testa. Se esiste una sola occorrenza, restituisce la successione di partenza. La funzione non deve modificare l'ordine degli elementi contenuti nella successione.

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

- lettura intero `i` da tastiera;
- cancellazione della penultima occorrenza di `i` da `s`, partendo dalla testa, usando la `my_delete` sopra definita;
- stampa sullo standard della nuova successione.

3. (file `parte3.c`, punti 5) Estendere il file `parte1.c` in modo che contenga la funzione `is_pal` che prende una successione `s` e restituisce `true` se `s` è uguale alla sua inversa, `false` altrimenti. Ad esempio, `is_pal(1 2 3 4 3 2 1) = true` mentre `is_pal(1 2 3 6 3 5) = false`. **La funzione non deve usare array di supporto**, ma utilizzare solo lo spazio relativo alla successione in input.

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

   - stabilire se `s` è uguale alla sua inversa, utilizzando la funzione `is_pal`, sopra definita;
   - stampa di un opportuno messaggio sullo standard output.

4. (file `parte4.c`, punti 7) Estendere il file `parte1.c` in modo che contenga la funzione `order` che prende due successioni `s1` ed `s2` e genera una nuova successione, ottenuta ordinando la concatenazione di `s1` ed `s2`. **La funzione non deve modificare le successioni di partenza e non può usare strutture ausiliarie, oltre a `s1`, `s2`, e la successione ordinata**.

   Funzione `main`:

   - legge la dimensione `n` degli array;
   - genera due successioni vuote `s1` ed `s2`, di dimensione `n`, utilizzando la `empty`;
   - inserisce in `s1` e in `s2` una sequenza di interi ≥ 0 letti da un file di input (usando la `conc_t` di cui sopra);
   - stampa (sullo standard output) di `s1` e `s2` (usando la `print` di cui sopra);
   - genera la nuova successione, usando la `order` di cui sopra;
   - stampa la nuova successione, usando la `print` di cui sopra;
   - stampa (sullo standard output) di `s1` e `s2` (usando la `print` di cui sopra).

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

   Esempio:

```
2 65 22 1 6
-1
 23 5 1 6 78  1 5
 -1
```