Laboratorio di Algoritmi e Strutture Dati

Esercitazioni del 15 Gennaio 2013

Esercizio 1: listsort

Esercizio sull'uso delle liste.

Si può liberamente utilizzare l'implementazione di liste di `int` vista a lezione, che trovate nel materiale didattico predisposto per questa lezione (`intlist.c`, `intlist.h`).

Il programma deve poter essere lanciato come:

`listsort` $file$ n

dove $file$ è un file che contiene una sequenza di numeri interi (vedi `numeri.txt` nel materiale didattico predisposto), e n è un intero positivo.

Il programma deve:

1. Leggere n numeri dal $file$ (si suppone che $file$ contenga sempre almeno n numeri) e inserirli in un array di interi allocato dinamicamente.
2. Creare una lista di interi contenente gli n elementi dell'array (un nodo della lista per ogni elemento dell'array; se l'array contiene k occorrenze del numero m, allora la lista deve avere k nodi in cui è memorizzato il numero m), attraverso un'opportuna funzione:

   ```
   intlist *buildlist(int *array, int n)
   ```

   che restituisca la radice della lista creata.
3. Stampare il contenuto della lista.
4. Ordinare (in senso crescente) la lista attraverso un'opportuna funzione:

   ```
   intlist *listsort(intlist *root)
   ```

   che ha come argomento la radice di una lista di interi e restituisce la radice di una lista di interi ordinata.
5. Stampare il contenuto della lista.

In ogni momento non devono essere allocati in memoria dinamica più di $n + 1$ nodi di lista.

NOTA: per convertire una stringa che rappresenta un numero intero nel numero corrispondente si può utilizzare la funzione di libreria standard:

```
int atoi(const char *s); /* prototipo in stdlib.h */
```

che restituisce l'intero corrispondente alla stringa `s`.

Supplemento

Implementare il programma sopra descritto in modo tale che dopo il punto 2 (creazione della lista dall'array) non venga eseguita alcuna ulteriore allocazione dinamica di memoria.