

Fondamenti di Informatica

Prof. M. Gavanelli, E. Lamma

16 Giugno 2016

Esercizio 1 (Punti 15 su 31 + 20 su 32) (2h)

In un file binario `alimenti.bin`, sono scritti le giacenze dei prodotti alimentari di un piccolo grossista alimentare. Per ciascun prodotto, il file `alimenti.bin` contiene

- il nome del prodotto (stringa di 50 char),
- il numero di scatole (intero).

Ad esempio, per il prodotto *cracker* si ha:

"cracker" 439

perché ci sono 439 scatole di cracker.

In un secondo file, di testo, `pranzo.txt`, sono scritti gli alimenti ordinati da un supermercato cliente del grossista. Ad esempio:

| |
|--------------|
| "cracker" 40 |
| "pasta" 100 |
| "tonno" 80 |

se si è ordinato 40 scatole di cracker, 100 di pasta e 80 di tonno.

Si realizzi un programma C, contenente almeno le seguenti funzioni, rispettivamente dedicate a:

1. a partire dal file `alimenti.bin`, creare una lista L in memoria centrale che contiene i dati dei prodotti, ordinata in base al nome del prodotto; la **funzioneA** riceve come parametri:

- il puntatore al file,
- il puntatore a L (inizializzato a `NULL` nel `main`),

più eventuali parametri a scelta, e restituisce il puntatore al primo elemento della lista L;

2. stampare la lista L a video; la **funzioneB** riceve come parametri:

- il puntatore a L,

più eventuali parametri a scelta, e restituisce `void` ;

3. caricare il contenuto del file `pranzo.txt` in un opportuno array di strutture.
4. mostrare a video il contenuto dell'array.
5. ordinare l'array in base al nome del prodotto utilizzando la funzione `qsort`.
6. a partire dall'array creato al punto ?? e dalla lista `L` creata al punto ??, determinare quali alimenti ordinati possono essere consegnati (sono solo quelli ordinati in quantità minore o uguale a quella disponibile presso il grossista), e stampare i loro nomi su un file di uscita `output.txt` da consegnare con i codici sorgente; la **funzioneC** riceve come parametri
 - l'array creato al punto ??
 - il puntatore a `L`,

più eventuali parametri a scelta, e restituisce `void`; si noti che la quantità di un prodotto ordinato può essere maggiore di quella disponibile in magazzino; in questo caso il prodotto ordinato non è inserito nella consegna.

FACOLTATIVO: Considerando che sia la lista, sia l'array sono ordinati rispetto al nome del prodotto, è possibile scrivere un algoritmo simile al merge di due array ordinati ...

NOTA BENE: Si consegnino i sorgenti e il file di uscita generato. È possibile utilizzare **libreria C** (ad esempio per le stringhe). Nel caso si strutturi a moduli l'applicazione qualunque **libreria utente** va riportata nello svolgimento.

Esercizio 2 (Punti 3 su 31) (15 min)

NOTA BENE: Per questo esercizio si consegna la soluzione in un file `complex.txt`.

Sia data la seguente funzione `fun` che riceve un carattere e un albero binario di ricerca di caratteri

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

Si indichi cosa fa la funzione `fun`. Se ne valuti anche la complessità asintotica come numero di test `i==T->value`, motivando adeguatamente, nel caso in cui il carattere `i` non appartenga all'albero `T`.