Esercitazione 6

- Ricorsione – powerset (5.3.13)
- Liste (2.3.1 e 2.3.4)

Ricorsione

Consideriamo una sequenza di N frecce adiacenti, ognuna orientata verso destra o verso sinistra.

►◄►►►◄

Assumiamo N pari

Definiamo uno scontro come due frecce consecutive che puntano una verso l'altra: ►◄

Dato uno scontro, definiamo la parte sinistra come la sequenza di frecce contigue orientate verso destra di lunghezza massima

Analogamente, definiamo la parte destra come la sequenza di frecce contigue orientate verso sinistra di lunghezza massima

Ricorsione

Definiamo uno scontro in equilibrio se la sua parte destra e la sua parte sinistra hanno la stessa lunghezza

Definiamo una sequenza equilibrata se tutti gli scontri che contiene sono equilibrati

Vogliamo determinare il numero minimo di frecce da girare per rendere equilibrata una sequenza

Ricorsione

Soluzione ricorsiva: tra tutti i possibili sottoinsiemi di frecce da girare, vogliamo selezionare un sottoinsieme che soddisfa le seguenti proprietà:

- Rende la sequenza equilibrata
- Ha cardinalità minima

Modello: powerset – generiamo tutti i possibili sottoinsiemi contenenti frecce da girare

- Disposizioni ripetute: ad ogni passo decido se la freccia va girata o meno (equivalente a generare numeri binari)
- Combinazioni semplici: per ogni cardinalità k seleziono tutti gli insiemi di k frecce da girare

Liste / 1

Un file di testo stringhe.txt contiene un numero indefinito di stringhe (di al massimo 30 caratteri) da memorizzare in una lista

Ogni elemento della lista deve contenere una stringa in un vettore allocato dinamicamente della dimensione opportuna

Si prevedano:

- Una opportuna struttura dati per rappresentare la lista
- Una funzione di inserimento ordinato nella lista
- Una funzione di salvataggio della lista su file

Liste / 2

Sia data una lista di interi ordinati in ordine ascendente. Alla lista si accede mediante puntatore alla testa.

Si scriva una funzione C che, senza usare vettori o liste di appoggio, completi la sequenza di interi, inserendo nella posizione corretta all'interno della lista tutti i numeri mancanti e visualizzi la lista così generata.

La chiamata alla funzione sia nella forma:

```
int aggiungi(list *l);
```

dove la variabile l rappresenta la lista stessa. La funzione ritorna come valore intero il numero di nodi aggiunti.