UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Dispense del Corso di Laboratorio di Fondamenti di Informatica II e Lab

Esercitazione 08: Liste

Ultimo aggiornamento: 08/05/2019

Introduzione

- Sono date le implementazioni delle seguenti primitive:
  - `list EmptyList();`
  - `list Cons(const element *e, list l);`
  - `bool IsEmpty(list l);`
  - `element Head(list l);`
  - `list Tail(list l);`
  - `element Copy(const element *e);`

- Di cui trovate la documentazione provvisoria al seguente link https://vivappz.appspot.com/liste__int_8h.html

Liste: Rubrica 1/2

- Siano date le seguenti definizioni di tipi:

```
typedef struct indirizzo {
        char nome[40];
        char via[50];
        int civico;
        char citta[30];
        char provincia[3];
        char cap[6];
} indirizzo;

typedef indirizzo element;

typedef struct list_element {
        element value;
        struct list_element *next;
} item;

typedef item* list;
```

Liste: Rubrica 1/3

Nel file `rubrica.c` si implementi la definizione delle seguenti funzioni:

1) `list InsertBack(list l, const element *e)`

La funzione prende in input una lista di indirizzi (anche vuota) `l` e un indirizzo (`e`) e lo aggiunge in coda alla lista. La funzione deve quindi restituire la lista risultante.

2) `void StampaLista(list l)`

La funzione prende in input una lista di indirizzi e la visualizza su standard output nel seguente modo:

```
nome via civico citta provincia cap\n
```

Liste: Rubrica 2/3

3) `element* Cerca(list l, const char *nome)`

La funzione prende in input una lista di indirizzi (anche vuota) e un nome. La funzione deve cercare il nome tra gli indirizzi della lista e ritornare il puntatore all'elemento corrispondente o NULL in caso il nome non sia presente.

4) `list Elimina(list l, const char *nome)`

La funzione prende in input una lista di indirizzi (anche vuota) e un nome. La funzione deve cercare il nome tra gli indirizzi della lista e eliminare l'elemento corrispondente (se presente) e restituire la lista risultante.

Liste: Rubrica 3/3

5) `list Ordina(list l)`

La funzione prende in input una lista di indirizzi (anche vuota) e la ordina per nome usando la funzione `strcmp`. La funzione deve ritornare la lista ordinata.

6) `list Filtra(list l, const char *citta)`

La funzione prende in input una lista di indirizzi (anche vuota) e una citta e costruisce una nuova lista contenente tutti e soli gli indirizzi di quella città. Se non ce ne sono ritorna NULL.

Liste: Rubrica 3/3

7) `list Reverse(list l)`

La funzione prende in input una lista di indirizzi (anche vuota) e ritorna una nuova lista contenente gli stessi indirizzi ma in ordine inverso. La lista originale non deve essere modificata.

8) `list Append(list l1, list l2)`

La funzione prende in input due liste di indirizzi (anche vuote) e ritorna una nuova lista contenente tutti gli indirizzi della prima seguiti da tutti quelli della seconda. Le liste originali non devono essere modificate.

9) `list AppendMod(list l1, list l2)`

La funzione prende in input due liste di indirizzi (anche vuote), concatena la seconda lista alla prima e ritorna l'indirizzo del primo elemento della lista risultante.