LIA - DEIS - Facoltà di Ingegneria - Università di Bologna

Elementi di informatica LB

Corso di laurea in Ingegneria Elettrica - Anno Accademico 2007-2008

Progetti su più file su memoria dinamica, liste e File

Esercizio n.1

Si scriva un programma C che:

a. inserisca in un vettore V1 (supposto allocato dinamicamente, chiedendo all'utente la dimensione massima) il numero di chilometri percorsi dalle automobili che vengono registrate presso una nota carrozzeria di Bologna, unitamente alle loro targhe.

b. Chiami una funzione che riceva come parametri di ingresso il vettore V1 e la sua dimensione e restituisca come parametro di uscita un nuovo vettore Vnew contenente i valori di V1 in cui i chilometri percorsi da ciascuna macchina risultino minori di un valore Max specificato.

c. Chiami una funzione che riceva come parametri di ingresso il vettore V1 e la sua dimensione e restituisca come parametro di uscita una lista in memoria dinamica contenete i dati di solo quelle targhe che iniziano per 'B'. Si stampi un messaggio di errore nel caso non ne esistano.

i dati in ingresso sono nel seguente formato:
BD234AH 28000
AE648BD 24000
CM900ER 31000
...

```
#include <stdio.h>
#include <stdlib.h>

typedef struct
{
char targa[8];
int km;
}veicolo;
```

```
typedef struct list_element
{ veicolo value;
struct list_element *next;
} item;

typedef item* list;
typedef int boolean;

boolean empty(list);
veicolo head(list);
list tail(list);
list cons(element, list);
void showlist(list);
void minore(int max,int *N,int dimOld, veicolo *V, veicolo *Vn);
list targaB(int dimOld, veicolo *V);

void main ()
{
veicolo *V1,*Vnew;
int M,dim,dimNew=0,i;
list l;
printf ("inserisci il numero di auto\n");
scanf("%d",&dim);
V1=(veicolo*)malloc(dim*sizeof(veicolo));
Vnew=(veicolo*)malloc(dim*sizeof(veicolo));
printf("inserisci i km massimi?\n");
scanf("%d",&M);
printf("inserisci km e targhe\n");
for(i=0;i<dim;i++)
{
printf("%d° targa\n",i+1);
scanf("%s",V1[i].targa);
printf("%d° km\n",i+1);
scanf("%d",&V1[i].km);
}
minore (M,&dimNew,dim,V1,Vnew);
printf("Le auto che hanno fatto km<%d: \n",M);
for(i=0;i<dimNew;i++)
{
printf("targa: %s\n",Vnew[i].targa);
printf("km: %d\n",Vnew[i].km);
}
l=targaB(dim,V1);
showlist(l);
}

void minore(int max,int *N,int dimOld, veicolo *V, veicolo *Vn)
{
int i = 0;
*N=0;
while (i<dimOld)
{
if (V[i].km < max)
{
Vn[*N]=V[i];
*N= ++(*N);
}
i++;
}
}

boolean empty(list l) /* verifica se lista vuota */
{ return (l==NULL); }
```

```
veicolo head(list l) /* selettore testa lista */
{ if (empty(l)) abort();
else return (l->value);
}

list tail(list l) /* selettore coda lista */
{ if (empty(l)) abort();
else return (l->next);
}

list cons(veicolo e, list l)
{ list t; /* costruttore che aggiunge in testa alla lista */
t=(list)malloc(sizeof(item));
t->value=e;
t->next=l;
return(t);
}

void showlist(list l)
{ /* VERSIONE ITERATIVA: */
printf("[ ");
while (!empty(l))
{ printf("%s, %d ",head(l).targa,head(l).km);
l=tail(l);
}
printf("]\n");
if (empty(l)) printf("non ci sono veicoli targati B...!\n");
}
list targaB(int dimOld, veicolo *V)
{
int i = 0;
list ls=NULL, tmp=NULL;
while (i<dimOld)
{
if (V[i].targa[0]=='B')
{
tmp=cons(V[i],ls);
}
i++;
}
return tmp;
}
```

Esercizio n.2

Un file di testo (TEMP.TXT) contiene i dati relativi alle medie di tutti gli studenti che devono accedere ad una sessione di laurea.
Si realizzi un programma C che:
1. Costruisca in memoria centrale una lista che memorizzi, in modo ordinato crescente tali medie (intere) e la stampi.
2. Letti due valori interi da console N e M, utilizzando la lista, visualizzi il valore delle medie comprese fra N e M ed un opportuno messaggio se non ne esistono.

Ad esempio:
Contenuto di TEMP.TXT
90
100
98
110
88
87

intervallo
88 101
stampa
88 90 98 100

```
/* da dividere su più file considerando per le funzioni mancanti l'ADT delle li
sulla pagina WEB del corso */

#include <stdio.h>
#include <stdlib.h>
#include "list.h"

main(){
element e, min, max;
list L=emptylist();
FILE *f1;
int i;

/* DOMANDA 1 */
f1 = fopen("TEMP.TXT", "r");
while (fscanf(f1,"%d",&e) !=EOF ) L=insord(e,L);
showlist(L);
fclose(f1);

/* DOMANDA 2 */
printf("Dammi i due estremi (66-110): ");
scanf("%d%d", &min, &max);
while (!empty(L) && (head(L)<min)) L=tail(L);
if (empty(L)) printf("\nnessun valore");
else{
printf("\n");
while ( !empty(L) && (head(L)<max)){
printf("%d ", head(L));
L=tail(L);
}
}
fclose(f1);
}
```

Esercizio n.3

Un file binario (MEMO.DAT) contiene i dati relativi agli appuntamenti giornalieri di uno studio legale. Ciascun appuntamento è caratterizzato dal cognome della persona da

incontrare (al più di 20 caratteri) e l’ora (numero intero da 7 a 20).

Si realizzi un programma C che:
1. Realizzi una lista che memorizza, in modo ordinato in base ai
cognomi, i nominativi delle persone che saranno ricevute e l’ora
prevista.

2. Letto un cognome a terminale, utilizzi la lista per andare a
visualizzare a quale ora tale persona sarà ricevuta o un opportuno
messaggio se non è previsto nessun appuntamento.

3. Stampi a video l’elenco dei nominativi (uno per linea) contenuti
nella lista, ciascuno preceduto dall'ora del proprio appuntamento.

Ad esempio:
Contenuto di MEMO.DAT:
VERDI 17
ROSSI 7
BIANCHI 9
BLU 7
NERI 17

Risultato della stampa:
9 BIANCHI
7 BLU
17 NERI
7 ROSSI
17 VERDI

SCHEMA DELLA SOLUZIONE

Suddivido il programma nei seguenti file:

list.c funzioni di libreria per la gestione di liste
list.h header file associato a list.c
el.c funzioni di utilità dipendenti dalla rappresentazione di element_type
el.h header file associato ad el.c (contiene la dichiarazione di element_type)
main.c contiene il programma principale

```
/* LIST ELEMENT TYPE - file el.h */
typedef struct {char cognome[20];
int ora;} element_type;

typedef enum {false, true} bool;

bool isequal(element_type, element_type);
bool isless(element_type, element_type);
```

```
void showel (element_type);

/* LIST ELEMENT TYPE - file el.c */
#include "el.h"
#include <string.h>
bool isequal(element_type e1, element_type e2)

/* uguaglianza sul cognome */
{return !strcmp(e1.cognome,e2.cognome); }
bool isless(element_type e1, element_type e2)

/* relazione d'ordine sul cognome */
{return (strcmp(e1.cognome,e2.cognome) < 0);}
void showel (element_type e)
{ printf("%s\t%d\n",e.cognome,e.ora);}

/* LIST INTERFACE - file list.h */
#include "el.h"

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

typedef item* list;

/* PROTOTIPI DI FUNZIONE (extern) */
list emptylist();
bool empty(list);
element_type head(list);
list tail(list);
list cons(element_type, list);
list ordins(element_type, list);
element_type member(element_type, list);

/* LIST IMPLEMENTATION - file list.c */

#include "list.h"
#include <stdlib.h>

list emptylist() { return NULL; };

bool empty(list l) { return (l==NULL); };

element_type head(list l) {
if (empty(l)) { abort(); }
else return(l->value); }
list tail(list l) {
if (empty(l)) { return emptylist(); }
else { return (l->next); }
}

list cons(element_type e, list l)
{list t;
t=(list)malloc(sizeof(item));
t->value=e;
t->next=l;
return(t);
}
```

```
list ordins(element_type el, list l)
{element_type e;
/* inserimento ordinato con possibili duplicaz. */
if (empty(l)) return(cons(el,l));
else
{ if (isless(el,head(l)))
return(cons(el,l));
else return(cons(head(l),
ordins(el,tail(l))) );
}
}

element_type member (element_type el,
list l)
/* cerca nella lista l un elemento uguale a el */
{ if (!empty(l))
{ if (isequal(el,head(l)))
return head(l);
else return member(el,tail(l)); }
else abort();
}

/* PROGRAMMA PRINCIPALE - file main.c */
#include <stdio.h>
#include <stdlib.h>
#include "list.h"

main() {
element_type e; list L,L1;
FILE *f1; char C[20]; int orario, i;
L=emptylist();
/* creazione file memo.dat */
f1 = fopen("MEMO.DAT", "w");
do {printf("Inserisci cognome (fine per terminare):");
scanf("%s",C); if (strcmp(C,"fine")==0) break;
printf("Inserisci ora: ");
scanf("%d",&orario); strcpy(e.cognome,C);
e.ora=orario;
fwrite(&e,sizeof(element_type),1,f1);
} while (1);
fclose(f1);

/* DOMANDA 1 */
f1 = fopen("MEMO.DAT", "rb");
while (fread(&e,sizeof(element_type),1,f1) > 0)
L = ordins(e,L);
fclose(f1);

/* DOMANDA 2 */
printf("\nDammi il cognome da cercare: ");
scanf("%s", e.cognome);
L1=L;
while (!empty(L1))
{ if (isequal(head(L1), e)) showel(head(L1));
L1 = tail(L1); }

/* DOMANDA 3 */
while (!empty(L))
{ printf("%d\t%s\n",
(L->value).ora,(L->value).cognome);
L=tail(L); }
}
```

Esercizio n.4

È dato un file di testo CANZONI.TXT che contiene i dati di una serie di canzoni (non più di 20), una canzone per riga. Più precisamente, ogni riga contiene nell'ordine:
• autore (non più di 20 caratteri, senza spazi intermedi);
• uno e un solo spazio;
• titolo del brano (non più di 30 caratteri, senza spazi intermedi);
• uno e un solo spazio;
• durata in secondi (numero intero).

Si chiede di scrivere un programma C che, dopo aver definito una struttura brano nel modo appropriato a quanto sopra:

1. contenga una funzione leggi_brani() che, dato il nome del file
(ed eventualmente altri parametri se opportuno), legga i dati delle
canzoni dal file e li metta in un array di brano di nome vettoreb.
Si mostri a video l'array così costruito.

2. Chieda all'utente il nome di un autore. A partire dal vettore
costruito e dall'indicazione dell'utente, costruisca una lista di interi
i cui elementi sono dati dalla durata dei brani dell’autore indicato
(si supponga che l'autore inserito dall'utente sia presente nell'elenco). Si visualizzi la lista costruita.

3. contenga una funzione double media(list listab) con la quale visualizzi la media (con decimali) della durata dei brani
contenuti nella lista.

Soluzione: fare riferimento all'ADT delle liste Fornito sulla pagina WEB del co

```
#include <stdio.h>
#include <stdlib.h>
#include "list.h"
#define NUMEROBRANI 20
#define DIMAUTORE 21
#define DIMTITOLO 31

typedef struct{
char autore[DIMAUTORE];
```

```
char titolo[DIMTITOLO];
int durata;
} brano;

/* ---------- domanda 1 -------- */
void leggi_brani(char nomefile[], brano v[], int* pindice){
brano x;
FILE *f = fopen(nomefile, "r");
if (f==NULL) {
printf("Impossibile aprire file di ingresso");
exit(1); } /* se non riesce a creare il file
visualizza un messaggio di errore ed esce */
while (fscanf(f, "%s%s%d\n", x.autore, x.titolo,
&x.durata)>0) {
v[*pindice] = x; (*pindice)++; }
fclose(f);
}

void mostraVettoreBrani(brano elenco[], int dim){
int i;
for (i=0; i<dim; i++)
printf("\nautore:%s\ntitolo:%s\ndurata:%ds\n",
elenco[i].autore, elenco[i].titolo, elenco[i].durata);
}

/* ---------- domanda 3 -------- */
double media(list listab) {
double ris = 0.0;
int n = 0; list listaux;
listaux = copy(listab);
/* si opera su lista ausiliaria, lasciando intatta listab */
while (!empty(listaux)) {
ris = ris + head(listaux);
listaux = tail(listaux);
n++;
}
return ris/n;
}

main(){
brano vettoreb[NUMEROBRANI];
list listab = emptylist();
int dim = 0, i;
FILE *f;
char autore[DIMAUTORE];
double durata_media;

/* --------------- prova domanda 1 --------------------*/
leggi_brani("CANZONI.TXT", vettoreb, &dim);
mostraVettoreBrani(vettoreb, dim);

/* --------------- domanda 2 --------------------*/
printf("\nInserire l'autore: ");
scanf("%s",autore);
for (i=0; i<dim; i++)
if (strcmp(vettoreb[i].autore,autore)==0)
listab = cons(vettoreb[i].durata,listab);
showlist(listab);

/* ------------------ domanda 3 ----------------------*/
durata_media = media(listab);
printf("\nLa durata media dei brani e': %lf\n",
```

```
durata_media);
}
```