ESERCIZIO su liste di interi

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.DAT

90 100 98 110 88 87

intervallo

88 101

stampa

88 90 98 100

È possibile utilizzare ***librerie C*** (ad esempio per stringhe) e si devono utilizzare le librerie sulle liste presentate a lezione.
Qualunque ***libreria utente*** utilizzata va riportata nello svolgimento.

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
element.c	funzioni di utilità dipendenti dalla rappresentazione di element_type
element.h	header file associato ad el.c (contiene la dichiarazione di element)
main.c	contiene il programma principale (funzione main())

```
/* PROGRAMMA PRINCIPALE - file main.c */
#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 su LISTE GENERICHE

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

È possibile utilizzare le ***librerie standard C sulle stringhe e le librerie sulle liste*** esaminate durante il corso.

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 Tipo" Compito Finale - Laboratorio di Informatica L-A

Un file di testo ARCHIVIO.TXT contiene i dati (***primo autore***, ***titolo***, ***numero*** di copie ***possedute***, ***numero*** di copie in ***prestito***) relativi ai differenti volumi conservati presso una biblioteca. Più precisamente, ogni riga del file contiene nell'ordine, separati da uno spazio bianco:

- ***autore*** (non più di 20 caratteri senza spazi intermedi);
- ***titolo*** (non più di 50 caratteri senza spazi intermedi);
- ***numero_possedute*** (da leggersi come intero);
- ***numero_prestito*** (da leggersi come intero).

Si realizzi un programma C che:

1. Legga il contenuto di ARCHIVIO.TXT e costruisca in memoria centrale un ***vettore V*** di strutture corrispondenti (si supponga che il file ARCHIVIO.TXT non possa contenere più di 30 righe). Si stampi a video il contenuto del vettore.

2. A partire da V, costruisca una ***lista L di interi*** contenente per ciascun volume il ***numero di copie disponibili*** nella biblioteca, ovvero la differenza fra il numero di copie possedute e il numero di copie in prestito. Si stampi a video il contenuto della lista L.

3. Utilizzando L per ottenere la somma delle copie disponibili e V per la somma delle copie possedute, calcoli ***il rapporto fra volumi disponibili e volumi posseduti***.

Oppure

3bis. Utilizzando la lista di interi L, stampi il numero di riga di ARCHIVIO.TXT relativo al volume con più copie disponibili. In caso di più volumi con pari numero di copie disponibili, qualunque riga relativa a questi ultimi è considerata una risposta corretta.

Ad esempio:

Contenuto di ARCHIVIO.TXT:

```
Salinger IlGiovaneHolden 10 8
Wallace InfiniteJest 12 3
Carver Cattedrale 12 12
Baricco Seta 6 0
Hornby ComeDiventare 9 9
Sartre LaNausea 3 1
Robbins NaturaMorta 7 7
```

Stampa di L:

```
[2, 9, 0, 6, 0, 2, 0]
```

È possibile utilizzare ***librerie C*** (ad esempio per stringhe) e si devono utilizzare le librerie sulle liste presentate a lezione. Qualunque ***libreria utente*** addizionale eventualmente utilizzata va riportata nello svolgimento e consegnata.

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
element.h	contiene la dichiarazione di element
mainLibri.c	contiene il programma principale

```
/* PROGRAMMA PRINCIPALE - file mainLibri.c */
#include <stdio.h>
#include <stdlib.h>
#include "list.h"
#define MAX 20

typedef struct{
   char autore[20];
   char titolo[50];
   int possedute;
   int prestito;
} volume;

main() {
  volume e; list L,L1; FILE *f;
  volume V[MAX]; int elementi=0,i,pos,max;
  int somma_possedute, somma_disponibili;
  L=emptylist();


/* DOMANDA 1 */
  f = fopen("ARCHIVIO.TXT", "r");
  if (f==NULL) {
    printf("Impossibile aprire file di ingresso");
    exit(1); } /* se non riesce a creare il file
              visualizza messaggio di errore ed esce */
  while (fscanf(f, "%s%s%d%d\n", e.autore, e.titolo,
       &e.possedute, &e.prestito)>0)
```

```
            V[elementi++] = e;
  fclose(f);
for (i=0; i<elementi; i++)
   printf("Volume %d: %s\t%s\t%d\t%d\n",i,V[i].autore,
       V[i].titolo,V[i].possedute,V[i].prestito);


/* DOMANDA 2 */
  for (i=0; i<elementi; i++)
       L = cons(V[i].possedute-V[i].prestito,L);
  showlist(L);
/* in che ordine viene stampata la lista??? */


/* DOMANDA 3 */
  for (i=0; i<elementi; i++)
       somma_possedute += V[i].possedute;
  L1=L;
  while (!empty(L1))
       { somma_disponibili += head(L1);
         L1=tail(L1);
       }
  printf("Rapporto disponibili/possedute = %f\n",
       (float)somma_disponibili/somma_possedute);


/* DOMANDA 3bis */
  i=0; max=-1;
  while (!empty(L))
       { if (head(L)>max) {
             max = head(L); pos=i; }
         L=tail(L); i++;
       }
  printf("Volume con più copie disponibili: %d",
          elementi-pos-1);
}
```

Ultima Esercitazione - Laboratorio di Informatica

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.

È possibile utilizzare librerie C (ad esempio per stringhe) e si devono utilizzare le librerie sulle liste presentate a lezione. Qualunque *libreria utente* utilizzata deve essere consegnata.

Soluzione

```
#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);
}
```