```
/*
 * Corso di Fondamenti di Informatica
 * Esercizio:
 * tema d'esame
 */

/* Si consideri un sottoprogramma che, ricevuto in ingresso il puntatore a FILE associato ad un file di testo
precedentemente aperto dal programma chiamante, crea una lista dinamica che riporta lâ€™insieme dei valori interi contenuti
nel file, e il numero di volte in cui ciascuno di essi compare, e restituisce la testa della lista. Il file di testo
contiene una successione di numeri interi, separati l'uno dall'altro da caratteri '\n'.
Ad esempio, se il file contiene i numeri

  5, 0, -3, 0, 5, 2, 0, 121, -18

la lista restituita sarÃ

  L -> (5,2) -> (0,3) -> (-3, 1) -> (2, 1) -> (121, 1) -> (-18, 1)

*/


#include <stdio.h>
#include <stdlib.h>
#define NOMEFILE "./file-es11_03/input.txt"

typedef struct EL
{
  int Valore;
  int NumOccorrenze;
  struct EL *Prossimo;
} ElemLista;

typedef enum {false, true} boolean;

ElemLista* ContaOccorrenze(FILE *Stream);
/* si veda il testo dell'esercizio; il parametro rappresenta il file
 * da esaminare */

void StampaOccorrenze(ElemLista *TestaLista);
/*** FUNZIONE NON RICHIESTA ALL'ESAME ***/
/* stampa a schermo il contenuto della lista contenente i dati sulle
```

```
 * occorrenze dei valori prodotta dalla funzione ContaOccorrenze */

int main()
/*** FUNZIONE NON RICHIESTA ALL'ESAME ***/
{
  ElemLista *Testa;
  FILE* FileInput;

  FileInput = fopen(NOMEFILE, "r");

  if ( NULL == FileInput )
  {
    printf("Errore di apertura file di ingresso!");
  }
  else
  {
    printf("\nFile di ingresso aperto correttamente.\n");

    Testa = ContaOccorrenze(FileInput);
    StampaOccorrenze(Testa);

    if ( 0 != fclose(FileInput) )
    {
      printf("Errore di chiusura file di ingresso!");
    }
  }

  return 0;
}

ElemLista* ContaOccorrenze(FILE *Stream)
{
  ElemLista *TestaLista;
  ElemLista *NuovoElem, *UltimoElem;
  /* puntano rispettivamente ad un nuovo elemento da inserire nella lista e
   * (quando la lista non e' vuota) all'ultimo elemento della lista */
  int ValoreLetto;
  /* ultimo numero letto dal file di input */
  ElemLista *Cursore;
  /* usato per esplorare la lista */
  boolean GiaPresente;
```

```
/* vale true se il valore in esame e' gia' presente nella lista, false
 * altrimenti */
int RisultatoLettura;
/* esito di un'operazione di lettura da file */

TestaLista = NULL;

do
{
  RisultatoLettura = fscanf(Stream, "%d", &ValoreLetto);

  if ( 1 == RisultatoLettura )
  /* se e' effettivamente stato letto un numero */
  {
    /* valutazione del numero di occorrenze del valore appena letto e
     * determinazione dell'ultimo elemento della lista */
    Cursore = TestaLista;
    GiaPresente = false;

    while ( (NULL != Cursore) && (false == GiaPresente) )
    /* esamina la lista alla ricerca di un elemento il cui campo
       Valore corrisponde a ValoreLetto; se lo trova, incrementa il campo
       NumOccorrenze di tale elemento (in questo caso non occorre
       aggiungere un nuovo elemento alla lista) */
    /* NOTA: la seconda parte della condizione di esecuzione del ciclo
       while interrompe l'esame della lista quando l'elemento cercato
       viene trovato. Cio' implica che, casi in cui l'elemento e'
       effettivamente presente, all'uscita dal ciclo while UltimoElem NON
       punti necessariamente all'ultimo elemento della lista.
       Cio' non rappresenta un problema, dal momento che tale puntatore
       verra' effettivamente utilizzato (per l'inserimento in coda di un
       nuovo elemento) soltanto nel caso in cui la ricerca nella lista
       dell'elemento contenente ValoreLetto non e' andata a buon fine */
    {
      UltimoElem = Cursore;

      if (Cursore->Valore == ValoreLetto)
      {
        ++Cursore->NumOccorrenze;
        GiaPresente = true;
      }
```

```
        Cursore = Cursore->Prossimo;
      }

      if (false == GiaPresente)
      {
        /* creazione di un nuovo elemento e suo inserimento in coda alla
         * lista */
        NuovoElem = malloc(sizeof(ElemLista));
        NuovoElem->Valore = ValoreLetto;
        NuovoElem->NumOccorrenze = 1;
        NuovoElem->Prossimo = NULL;

        if (NULL == TestaLista)
        {
          TestaLista = NuovoElem;
        }
        else
        {
          UltimoElem->Prossimo = NuovoElem;
        }
      }
    }
  } while ( 1 == RisultatoLettura );
  /* il ciclo termina se viene raggiunta la fine del file o una riga vuota
   * (nel primo caso fscanf restituisce EOF, nel secondo restituisce 0) */

  return(TestaLista);
}


void StampaOccorrenze(ElemLista *TestaLista)
/*** FUNZIONE NON RICHIESTA ALL'ESAME ***/
{
  ElemLista *Cursore;
  /* usato per esplorare la lista */

  printf("\n\nDati sulle occorrenze:\n\n");

  Cursore = TestaLista;
```

```
  while (NULL != Cursore)
  {
    printf("Valore: %d\tNumero occorrenze: %d\n", Cursore->Valore,
                                                  Cursore->NumOccorrenze);
    Cursore = Cursore->Prossimo;
  }
}
```