```
/*
 * Corso di Fondamenti di Informatica
 * Esercizio:
 * tema d'esame (prova in itinere)
 */

/*
NOTA SULLE LISTE SEQUENZIALI
----------------------------

Una lista sequenziale è una struttura dati (da NON confondere con una lista dinamica!) atta a contenere una successione di
dati tutti dello stesso tipo. Essa utilizza un ARRAY per memorizzare i dati.
Dal momento che la memoria occupata da un array è comunque sempre allocata per tutta la durata del programma, nel generico
istante di esecuzione del programma l'array sarà utilizzato solo in parte per dati utili; la parte rimanente conterrà dati
obsoleti oppure ignoti (perché mai inizializzati).
La dimensione dell'array, non essendo modificabile a run-time, va scelta oculatamente in modo da essere sicuramente
sufficiente ma non inutilmente grande. Inoltre occorre un meccanismo in grado di segnalare, in ogni momento, quanta parte
dell'array è occupata da dati utili. Tale meccanismo puo' essere costituito semplicemente un intero, associato all'array in
una struct, che contiene in ogni istante il numero di dati utili che l'array contiene. La struct che associa l'array e
l'intero è, appunto, la LISTA SEQUENZIALE.
Ad ogni aggiunta o rimozione di un elemento da una lista sequenziale è necessario aggiornare l'intero che ne segnala il
grado di occupazione. Inoltre per "svuotare" una lista sequenziale è sufficiente porre a zero tale intero: ciò non cancella
i dati contenuti nell'array, ma li segnala come non significativi (dunque da ignorare, e sovrascrivibili se necessario).
*/

/* Testo dell'esercizio:
Definire il tipo di dato tentativo_esame mediante una struttura contenente il nome di un esame (array di 20 caratteri) e un
esito di tipo float: l'esito coincide con il voto ottenuto (se non inferiore a 18.0) oppure, nel caso di tentativo fallito,
con -1.0. Definire il tipo di dato sequenza-tentativi mediante una lista sequenziale di strutture di tipo tentativo_esame:
la lista sequenziale deve contenere, oltre all'indicazione del numero dei tentativi, la sequenza -in ordine cronologico- dei
tentativi effettuati da uno studente di superare gli esami del suo piano di studi. Scrivere le dichiarazioni e le istruzioni
di un frammento di programma che stampa in sequenza i nomi degli esami superati con il voto positivo ottenuto, e la media
tra i soli voti positivi.
N.B Non è necessario codificare la lettura della struttura; si trascuri l'eventuale lode.
*/

#include <stdio.h>

#define MAX_LUNG 20
/* lunghezza massima ammessa per il nome di un esame */
#define MAX_TENT 100
```

```
/* massimo numero di tentativi di esame memorizzabili */
#define INSUFF -1.0
/* valore del voto che rappresenta, per convenzione, un tentativo d'esame
 * fallito */

int main()

{
  typedef struct
  {
    char NomeEsame[MAX_LUNG];
    /* nome dell'esame */
    float EsitoEsame;
    /* voto conseguito; il voto assume il valore INSUFF in caso di
     * insufficienza */
  } tentativo_esame;
  /* usata per rappresentare un tentativo d'esame */

  typedef struct
  {
    tentativo_esame DescrTent[MAX_TENT];
    /* array di descrizioni di tentativi di esame, in ordine cronologico */
    int NumTentativi;
    /* numero di tentativi di esame effettuati fino al momento attuale */
  } sequenza_tentativi;
  /* usata per rappresentare una sequenza cronologica di tentativi d'esame */

  sequenza_tentativi ArchivioTent;
  /* archivio dei tentativi di esame di un dato studente */
  int Cont;
  /* contatore usato per scandire gli elementi dell'archivio */
  float SommaParziale;
  /* somma dei soli voti positivi conseguiti */
  int NumeroVotiPositivi;
  /* numero dei voti positivi conseguiti */

  ArchivioTent.NumTentativi = 0;
  /* svuota l'archivio */

  /* ...lettura del contenuto di SeqTent... (NON RICHIESTA) */
```

```
  /*** inizio esame dell'archivio e stampa dei risultati ***/

  SommaParziale = 0;
  NumeroVotiPositivi = 0;

  for (Cont = 0; Cont < ArchivioTent.NumTentativi; ++Cont)
  {
    if ( ArchivioTent.DescrTent[Cont].EsitoEsame != INSUFF )
    /* se esito positivo */
          {
            SommaParziale = SommaParziale + ArchivioTent.DescrTent[Cont].EsitoEsame;

            ++NumeroVotiPositivi;

            printf("\nEsame di %s: superato con voto %f.",
             ArchivioTent.DescrTent[Cont].NomeEsame,
             ArchivioTent.DescrTent[Cont].EsitoEsame);
          }
  }

  if ( NumeroVotiPositivi > 0 )
  /* evita divisioni per zero */
  {
    printf("\n\nMedia dei voti positivi: %f",
          SommaParziale/NumeroVotiPositivi);
  }

  return(0);
}
```