PROVA PRATICA DI FONDAMENTI DI INFORMATICA A

COMPITO A

Si scriva un programma C che legga due serie di dati e li memorizzi in due vettori di strutture. Nel primo vettore SQUADRE (di dimensione 6) vengono memorizzate strutture (**`struct squadra`**) del tipo:
- nome squadra (stringa di lunghezza 20)
- codice squadra (intero)

Nel secondo vettore PARTITE (di dimensione 3) vengono memorizzate strutture (**`struct partita`**) del tipo:
- codice squadra 1 (intero)
- codice squadra 2 (intero)
- goal squadra 1 (intero)
- goal squadra 2 (intero)

Si scriva un programma che:

- tramite due procedure **`leggisquadra`** e **`leggipartita`** legga a terminale i dati da inserire nei due vettori;

```
void leggisquadra(int n, struct squadra S[]);
void leggipartita(int n, struct partita P[]);
```

- tramite la procedura **`vinteincasa`** stampi a terminale i nomi delle due squadre che hanno svolto una partita vinta dalla squadra 1. I nomi delle squadre sono da ricercare nel vettore SQUADRE a partire dal codice squadra recuperato dal vettore PARTITE.

```
void vinteincasa(struct partita P[], int n, struct squadra S[], int m);
```

dove **`n`** è la dimensione del vettore **`P[]`** e **`m`** la dimensione del vettore **`S[]`**

```
#include <stdio.h>
#define DIMS 6
#define DIMP 3

struct squadra{
       int codice;
       char Nome[20];};

struct partita{
       int codice1;
       int codice2;
       int goal1;
       int goal2;};

void leggisquadra(int n, struct squadra S[]);
void leggipartita(int n, struct partita P[]);
void vinteincasa(struct squadra S[], int n,  struct partita P[], int m);

main()
{int i;
 struct squadra SQUADRE[DIMS];
 struct partita PARTITE[DIMP];
 leggisquadra(DIMS,SQUADRE);
 leggipartita(DIMP,PARTITE);
 vinteincasa(SQUADRE,6,PARTITE,3);
 }

void leggisquadra(int n, struct  squadra Vet[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci Codice, Nome\n");
     scanf("%d",&Vet[i].codice);
     scanf("%s",Vet[i].Nome);
 }
}

void leggipartita(int n, struct partita Vet[]) {

int i;
for(i=0;i<n;i++)
    {printf("Inserisci codice1, codice2, goal1, goal2\n");
        scanf("%d",&Vet[i].codice1);
        scanf("%d",&Vet[i].codice2);
        scanf("%d",&Vet[i].goal1);
        scanf("%d",&Vet[i].goal2);
 }
}
```

```
void vinteincasa(struct squadra S[], int n, struct partita P[], int m){

int i,j;
for (i = 0; i < m; i++)
   {if (P[i].goal1 > P[i].goal2) /* la struct P[i] denota una partita vinta
in casa */
          for(j = 0; j < n; j++)
          {if (P[i].codice1 == S[j].codice)
               printf(" Squadra in casa: %s", S[j].Nome);
               else if (P[i].codice2 == S[j].codice)
                      printf(" Squadra ospite: %s", S[j].Nome);
          }
    }
}
```

PROVA PRATICA DI FONDAMENTI DI INFORMATICA A
31 OTTOBRE 2000 – DOCENTE MICHELA MILANO

COMPITO B

Si scriva un programma C che legga due serie di dati e li memorizzi in due vettori di strutture. Nel primo vettore SQUADRE (di dimensione 6) vengono memorizzate strutture (**`struct squadra`**) del tipo:

- nome squadra (stringa di lunghezza 20)
- codice squadra (intero)
- punti in classifica (intero)

Nel secondo vettore DOMENICA (di dimensione 3) vengono memorizzate strutture (**`struct puntipartita`**) del tipo:

- codice squadra (intero)
- aggiornamento punteggio (intero)

Il significato del campo aggiornamento punteggio sarà un intero che indica l'incremento dei punti in classifica della squadra relativamente alla partita disputata l'ultima domenica di campionato.

Si scriva un programma che:

- tramite due procedure **`leggisquadra`** e **`leggipunti`** legga a terminale i dati da inserire nei due vettori;

```
void leggisquadra(int n, struct squadra S[]);
void leggipunti(int n, struct puntipartita P[]);
```

- tramite la procedura **`aggiornaclassifica`** modifichi i punti in classifica di ciascuna squadra relativamente a tutte le partite giocate. In particolare, per ogni elemento nel vettore DOMENICA (contenente codice squadra, aggiornamento punteggio) la procedura aggiorna il punteggio in classifica della corrispondente squadra nel vettore SQUADRE sommando ai punti in classifica l'aggiornamento punteggio. Infine, stampa il nome della squadra e i punti aggiornati.

```
void aggiornaclassifica(struct puntipartita P[], int n, struct squadra S[],
int m);
```

dove **`n`** è la dimensione del vettore **`P[]`** e **`m`** la dimensione del vettore **`S[]`**

```
#include <stdio.h>
#define DIMS 6
#define DIMP 3

struct squadra{
       char nome[20];
       int codice;
       int punticlassifica;};

struct puntipartita{
       int codice;
       int aggiornaclassifica;};

void leggisquadra(int n, struct squadra S[]);
void leggipunti(int n, struct puntipartita P[]);
void  aggiornaclassifica(struct  squadra  S[],int  n,struct  puntipartita
P[],int m);

main()
{int i;
 struct squadra SQUADRE[DIMS];
 struct puntipartita DOMENICA[DIMP];
 leggisquadra(DIMS,SQUADRE);
 leggipunti(DIMP,DOMENICA);
 aggiornaclassifica(SQUADRE,6,DOMENICA,3);
 }

void leggisquadra(int n, struct squadra S[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci Codice, Nome, Punti in classifica: \n");
     scanf("%d",&S[i].codice);
     scanf("%s",S[i].nome);
      scanf("%d",&S[i].punticlassifica);
 }
}

void leggipunti(int n, struct puntipartita P[])
{
int i;
for(i=0;i<n;i++)
    {printf("Inserisci codice, punti guadagnati nell'ultima domenica\n");
        scanf("%d",&P[i].codice);
        scanf("%d",&P[i].aggiornaclassifica);
}
}

void  aggiornaclassifica(struct  squadra  S[],int  n,struct  puntipartita
P[],int m)
{
int i,j;
for (i = 0; i < m; i++)
 for(j = 0; j < n; j++)
```

```
   if (P[i].codice == S[j].codice)
    {S[j].punticlassifica = S[j].punticlassifica + P[i].aggiornaclassifica;
     printf("La squadra: %s ha punti %d", S[j].nome, S[j].punticlassifica);
    }
}
```

PROVA PRATICA DI FONDAMENTI DI INFORMATICA A
31 OTTOBRE 2000 – DOCENTE MICHELA MILANO

COMPITO C

Si scriva un programma C che legga due serie di dati e li memorizzi in due vettori di strutture. Nel primo vettore SQUADRE (di dimensione 4) vengono memorizzate strutture (**`struct squadra`**) del tipo:
- nome squadra (stringa di lunghezza 20)
- codice squadra (intero)
- goal fatti (intero)
- goal subiti (intero)

Il vettore SQUADRE contiene le informazioni sul nome e codice della squadra e i goal fatti e subiti dall'inizio del campionato.

Nel secondo vettore ULTIMI (di dimensione 4) vengono memorizzate strutture (**`struct goalpartita`**) del tipo:
- codice squadra (intero)
- goal fatti (intero)
- goal subiti (intero)

Il vettore ULTIMI contiene un record per ogni squadra che ha disputato una partita nell'ultima domenica di campionato riportando i goal fatti e subiti.

Si scriva un programma che:

- tramite due procedure **`leggisquadra`** e **`leggigoal`** legga a terminale i dati da inserire nei due vettori;

```
void leggisquadra(int n, struct squadra S[]);
void leggigoal(int n, struct goalpartita P[]);
```

- tramite la procedura **`aggiornagoal`** modifichi i goal fatti e subiti da ciascuna squadra. In particolare, per ogni elemento nel vettore ULTIMI la procedura aggiorna il goal fatti e subiti della corrispondente squadra nel vettore SQUADRE sommando ai goal fatti e subiti dall'inizio del campionato quelli fatti e subiti nell'ultima domenica di campionato. Infine, stampa il nome della squadra e i goal fatti e subiti aggiornati.

```
void aggiornagoal(struct goalpartita P[],int n,struct squadra S[], int m);
```

dove **`n`** è la dimensione del vettore **`P[]`** e **`m`** la dimensione del vettore **`S[]`**

```
#include <stdio.h>
#define DIMS 4
#define DIMP 4

struct squadra{
       char nome[20];
       int codice;
       int goalfatti;
       int goalsubiti;};

struct goalpartita{
       int codice;
       int goalfatti;
       int goalsubiti;};

void leggisquadra(int n, struct squadra S[]);
void leggigoal(int n, struct goalpartita P[]);
void aggiornagoal(struct squadra S[],int n,struct goalpartita P[],int m);

main()
{int i;
 struct squadra SQUADRE[DIMS];
 struct goalpartita ULTIMI[DIMP];
 leggisquadra(DIMS,SQUADRE);
 leggigoal(DIMP,ULTIMI);
 aggiornagoal(SQUADRE,4,ULTIMI,4);
 }

void leggisquadra(int n, struct  squadra Vet[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci Codice, Nome, Goal Fatti, Goal subiti: \n");
     scanf("%d",&Vet[i].codice);
     scanf("%s",Vet[i].nome);
      scanf("%d",&Vet[i].goalfatti);
      scanf("%d",&Vet[i].goalsubiti);
 }
}

void leggigoal(int n, struct goalpartita Vet[]) {

int i;
for(i=0;i<n;i++)
    {printf("Inserisci codice, goal fatti e subiti ultima domenica\n");
     scanf("%d",&Vet[i].codice);
     scanf("%d",&Vet[i].goalfatti);
      scanf("%d",&Vet[i].goalsubiti);

}
}

void aggiornagoal(struct squadra S[],int n, struct goalpartita P[],int m)
{
```

```
int i,j;
for (i = 0; i < m; i++)
 for(j = 0; j < n; j++)
   {if (P[i].codice == S[j].codice)
    {S[j].goalfatti = S[j].goalfatti + P[i].goalfatti;
     S[j].goalsubiti = S[j].goalsubiti + P[i].goalsubiti;
     printf("La squadra: %s ha fatto %d goal e ha subito %d goal",
S[j].nome, S[j].goalfatti, S[j].goalsubiti);
   }
}
}
```

PROVA PRATICA DI FONDAMENTI DI INFORMATICA A
31 OTTOBRE 2000 – DOCENTE MICHELA MILANO

COMPITO D

Si scriva un programma C che legga due serie di dati e li memorizzi in due vettori di strutture. Nel primo vettore SQUADRE (di dimensione 4) vengono memorizzate strutture (**`struct squadra`**) del tipo:
- nome squadra (stringa di lunghezza 20)
- codice squadra (intero)
- spesa complessiva per giocatori (intero)

Il vettore SQUADRE contiene le informazioni sul nome e codice della squadra e la spesa affrontata dalla squadra per l'acquisto dei giocatori.

Nel secondo vettore ACQUISTI (di dimensione 3) vengono memorizzate strutture (**`struct giocatore`**) del tipo:
- codice squadra (intero)
- nome giocatore (stringa di lunghezza 20)
- costo (intero)

Il vettore ACQUISTI contiene un record per ogni giocatore acquistato da una squadra ed il suo prezzo.

Si scriva un programma che:

- tramite due procedure **`leggisquadra`** e **`leggigiocatore`** legga da terminale i dati da inserire nei due vettori;

```
void leggisquadra(int n, struct squadra S[]);
void leggigiocatore(int n, struct giocatore P[]);
```

- tramite la procedura **`aggiornaspesa`** aggiorni la spesa sostenuta da ciascuna squadra. In particolare, per ogni elemento nel vettore ACQUISTI la procedura aggiorna la spesa complessiva per giocatori della corrispondente squadra nel vettore SQUADRE sommando alla spesa per giocatori complessiva quella degli ultimi acquisti. Infine, stampi il nome della squadra e la spesa sostenuta.

```
void aggiornaspesa(struct giocatore P[],int n,struct squadra S[], int m);
```

dove **`n`** è la dimensione del vettore **`P[]`** e **`m`** la dimensione del vettore **`S[]`**

```
#include <stdio.h>
#define DIMS 4
#define DIMP 4

struct squadra{
       char nome[20];
       int codice;
       int spesa;};

struct giocatore{
       int codicesquadra;
       char nomegioc[20];
       int spesa;};

void leggisquadra(int n, struct squadra S[]);
void leggigiocatore(int n, struct giocatore P[]);
void aggiornaspesa(struct squadra S[],int n,struct giocatore P[],int m);

main()
{int i;
 struct squadra SQUADRE[DIMS];
 struct giocatore ACQUISTI[DIMP];
 leggisquadra(DIMS,SQUADRE);
 leggigiocatore(DIMP,ACQUISTI);
 aggiornaspesa(SQUADRE,4,ACQUISTI,4);
 }

void leggisquadra(int n, struct  squadra Vet[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci Codice, Nome, Spesa per giocatori: \n");
     scanf("%d",&Vet[i].codice);
     scanf("%s",Vet[i].nome);
      scanf("%d",&Vet[i].spesa);
 }
}

void leggigiocatore(int n, struct giocatore Vet[]) {

int i;
for(i=0;i<n;i++)
    {printf("Inserisci codice squadra, nome giocatore, spesa:\n");
     scanf("%d",&Vet[i].codicesquadra);
     scanf("%s",Vet[i].nomegioc);
      scanf("%d",&Vet[i].spesa);

}
}

void aggiornaspesa(struct squadra S[],int n, struct giocatore P[],int m){
int i,j;
```

```
for (i = 0; i < m; i++)
 for(j = 0; j < n; j++)
   {if (P[i].codicesquadra == S[j].codice)
    {S[j].spesa= S[j].spesa + P[i].spesa;
printf("La squadra: %s ha speso %d per l'acquisto di giocatori", S[j].nome,
S[j].spesa);
   }
}
}
```

PROVA PRATICA DI FONDAMENTI DI INFORMATICA A

COMPITO E

Si scriva un programma C che legga due serie di dati e li memorizzi in due vettori di strutture.

Nel primo vettore FILM (di dimensione 6) vengono memorizzate strutture (**`struct film`**) del tipo:
- titolo (stringa di lunghezza 20 non contenente spazi bianchi)
- codice film (intero)

Nel secondo vettore PROIEZIONE (di dimensione 3) vengono memorizzate strutture (**`struct cinema`**) del tipo:
- nome cinema (stringa di lunghezza 20)
- codice film (intero)
- numero posti (intero)

Si scriva un programma che:
- tramite due procedure **`leggifilm`** e **`leggicinema`** legga da terminale i dati da inserire nei due vettori;

```
void leggifilm(int n, struct film F[]);
void leggicinema(int n, struct cinema C[]);
```

- tramite la funzione **`maxcinema`** determini il cinema (**`struct cinema`**) con il maggior numero di posti

```
struct cinema maxcinema(struct cinema P[], int n);
```

dove **`n`** è la dimensione del vettore **`P[]`**

- tramite la procedura **`stampacinema`** stampi a terminale il nome del film che viene proiettato nel cinema **`c`** contenente il maggior numero di posti restituito dalla funzione **`maxcinema`**.

```
void stampacinema(struct cinema c, struct film F[], int m);
```

dove **`m`** è la dimensione del vettore **`F[]`**

```
#include <stdio.h>
#define DIMS 6
#define DIMP 3
#define TRUE 1
#define FALSE 0

struct film{
       char titolo[20];
       int codice;};

struct cinema{
       char nome[20];
       int codicefilm;
       int numeroposti;};

void leggifilm(int n, struct film F[]);
void leggicinema(int n, struct cinema C[]);
struct cinema maxcinema(struct cinema C[], int n);
void stampacinema(struct cinema c, struct film F[], int m);

main()
{int i;
 struct film FILM[DIMS];
 struct cinema PROIEZIONE[DIMP];
 struct cinema c;
 leggifilm(DIMS,FILM);
 leggicinema(DIMP,PROIEZIONE);
 c=maxcinema(PROIEZIONE,DIMP);
 stampacinema(c,FILM,DIMS);
}

void leggifilm(int n, struct  film Vet[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci Codice, Titolo\n");
     scanf("%d",&Vet[i].codice);
     scanf("%s",Vet[i].titolo);
    }
}

void leggicinema(int n, struct cinema Vet[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci nome, codicefilm, num posti\n");
        scanf("%s",Vet[i].nome);
        scanf("%d",&Vet[i].codicefilm);
        scanf("%d",&Vet[i].numeroposti);
    }
}
```

```
struct cinema maxcinema(struct cinema C[], int n)
{ struct cinema c;
  int i, maxindice = 0;
  for (i=1; i<n; i++){
    if (C[maxindice].numeroposti < C[i].numeroposti)
      maxindice = i;
  }
  return C[maxindice];
}

void stampacinema(struct cinema c, struct film F[], int m)
{int i = 0, trovato = FALSE;
 while ((i < m)&& (!trovato))
   if (F[i].codice == c.codicefilm)
     {printf("Il film nel cinema piu' grande e' %s", F[i].titolo);
      trovato = TRUE;}
   else i++;
}
```

PROVA PRATICA DI FONDAMENTI DI INFORMATICA A
31 OTTOBRE 2000 – DOCENTE MICHELA MILANO

COMPITO F

Si scriva un programma C che legga due serie di dati e li memorizzi in due vettori di strutture.

Nel primo vettore FILM (di dimensione 3) vengono memorizzate strutture (**`struct film`**) del tipo:
- titolo (stringa di lunghezza 20 non contenente spazi bianchi)
- codice film (intero)
- costo (intero)
- numero giorni di riprese

Nel secondo vettore ATTORI (di dimensione 5) vengono memorizzate strutture (**`struct attore`**) del tipo:
- nome (stringa di lunghezza 20)
- codice film (intero)
- costo al giorno (intero)

Si scriva un programma che:
- tramite due procedure **`leggifilm`** e **`leggiattore`** legga da terminale i dati da inserire nei due vettori;

```
void leggifilm(int n, struct film F[]);
void leggiattore(int n, struct attore A[]);
```

- tramite la procedura **`aggiornacosto`** aggiorni il costo del film sommando a questo il costo di ciascun attore che partecipa al film. Il costo dell'attore viene calcolato come costo al giorno per il numero di giorni delle riprese.

```
void aggiornacosto(struct film F[], int n, struct attore A[], int m);
```

dove **`n`** è la dimensione del vettore **`F[]`** e **`m`** è la dimensione del vettore **`A[]`**

```
#include <stdio.h>

struct film{
       char titolo[20];
       int codice;
       int costo;
       int giorni;};

struct attore{
       char nome[20];
       int codicefilm;
       int costogiorno;};

void leggifilm(int n, struct film F[]);
void leggiattore(int n, struct attore A[]);
void aggiornacosto(struct film F[], int n, struct attore A[], int m);

main()
{int i;
 struct film FILM[3];
 struct attore ATTORI[5];
 leggifilm(3,FILM);
 leggiattore(5,ATTORI);
 aggiornacosto(FILM,3,ATTORI,5);
 }

void leggifilm(int n, struct film Vet[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci Codice, Titolo, Costo, numero giorni \n");
     scanf("%d",&Vet[i].codice);
     scanf("%s",Vet[i].titolo);
     scanf("%d",Vet[i].costo);
     scanf("%d",&Vet[i].giorni);
 }
}

void leggiattore(int n, struct attore Vet[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci nome, codicefilm, costo al giorno\n");
        scanf("%s",Vet[i].nome);
        scanf("%d",&Vet[i].codicefilm);
        scanf("%d",&Vet[i].costogiorno);
 }
}
```

```
void aggiornacosto(struct film F[], int n, struct attore A[], int m)
{int i,j;
 for (i = 0; i < n; i++)
  for (j = 0; j < m; j++)
         if (A[i].codicefilm == F[j].codice)
            {F[j].costo = F[j].costo + A[i].costogiorno*F[j].giorni;
printf("costo aggiornato del film %s: %d", F[j].titolo, F[j].costo);
                 }
}
```

PROVA PRATICA DI FONDAMENTI DI INFORMATICA A
31 OTTOBRE 2000 – DOCENTE MICHELA MILANO

COMPITO G

Si scriva un programma C che legga due serie di dati e li memorizzi in due vettori di strutture.

Nel primo vettore FILM (di dimensione 4) vengono memorizzate strutture (**`struct film`**) del tipo:
- titolo (stringa di lunghezza 20 non contenente spazi bianchi)
- codice film (intero)
- costo (intero)
- giorni (intero)

Nel secondo vettore REGISTI (di dimensione 4) vengono memorizzate strutture (**`struct regista`**) del tipo:
- nome (stringa di lunghezza 20)
- codice film (intero)
- costo al giorno (intero)

Si scriva un programma che:
- tramite due procedure **`leggifilm`** e **`leggiregista`** legga a terminale i dati da inserire nei due vettori;

```
void leggifilm(int n, struct film F[]);
void leggiregista(int n, struct regista A[]);
```

- tramite la procedura **`aggiornacosto`** aggiorni il costo del film sommando a questo il costo del regista del film. Il costo del regista viene calcolato come costo al giorno per il numero di giorni di riprese.

```
void aggiornacosto(struct film F[], int n, struct regista R[], int m);
```

dove **`n`** è la dimensione del vettore **`F[]`** e **`m`** è la dimensione del vettore **`R[]`**

```c
#include <stdio.h>

struct film{
       char titolo[20];
       int codice;
       int costo;
       int giorni;};

struct regista{
  char nome[20];
  int codicefilm;
  int costogiorno;};

void leggifilm(int n, struct film F[]);
void leggiregista(int n, struct regista R[]);
void aggiornacosto(struct film F[], int n, struct regista R[], int m);

main()
{int i;
 struct film FILM[4];
 struct regista REGISTI[4];
 leggifilm(4,FILM);
 leggiregista(4, REGISTI);
 aggiornacosto(FILM,4, REGISTI,4);
 }

void leggifilm(int n, struct film Vet[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci Codice, Titolo, numero giorni, costo \n");
     scanf("%d",&Vet[i].codice);
     scanf("%s",Vet[i].titolo);
     scanf("%d",&Vet[i].giorni);
     scanf("%d",&Vet[i].costo);
 }
}

void leggiregista(int n, struct regista Vet[]) {
int i;
for(i=0;i<n;i++)
    {printf("Inserisci nome regista, codicefilm, costo al giorno\n");
        scanf("%s",Vet[i].nome);
        scanf("%d",&Vet[i].codicefilm);
        scanf("%d",&Vet[i].costogiorno);
 }
}
```

```
void aggiornacosto(struct film F[], int n, struct regista R[], int m){
int i,j;
for (i = 0; i < n; i++)
  for (j = 0; j < m; i++)
     if (R[i].codicefilm == F[j].codice)
      {F[j].costo = F[j].costo + R[i].costogiorno*F[j].giorni;
       printf("costo aggiornato del film %s: %d", F[j].titolo, F[j].costo);
      }
}
```

PROVA PRATICA DI FONDAMENTI DI INFORMATICA A
31 OTTOBRE 2000 – DOCENTE MICHELA MILANO

COMPITO H

Si scriva un programma C che legga due serie di dati e li memorizzi in due vettori di strutture.

Nel primo vettore FILM (di dimensione 4) vengono memorizzate strutture (**`struct film`**) del tipo:
- titolo (stringa di lunghezza 20 non contenente spazi bianchi)
- codice film (intero)
- incasso complessivo (intero)

Nel secondo vettore ULTIMOMESE (di dimensione 4) vengono memorizzate strutture (**`struct ultimo`**) del tipo:
- codice film (intero)
- incasso ultimo mese (intero)

che rappresentano gli incassi effettuati dai film nell'ultimo mese.

Si scriva un programma che:
- tramite due procedure **`leggifilm`** e **`leggiultimo`** legga da terminale i dati da inserire nei due vettori;

```
void leggifilm(int n, struct film F[]);
void leggiultimo(int n, struct ultimo U[]);
```

- tramite la procedura **`aggiornaincasso`** aggiorni l'incasso complessivo del film sommandogli l'incasso ottenuto nell'ultimo mese. In particolare, per ogni elemento del vettore ULTIMOMESE, acceda al vettore FILM e aggiorni il campo incasso del film corrispondente.

```
void aggiornaincasso(struct film F[], int n, struct ultimo U[], int m);
```

dove **`n`** è la dimensione del vettore **`F[]`** e **`m`** è la dimensione del vettore **`U[]`**

```
#include <stdio.h>

struct film{
       char titolo[20];
       int codice;
       int incasso; };

struct ultimo{
       int codicefilm;
       int incassoultimo;};

void leggifilm(int n, struct film F[]);
void leggiultimo(int n, struct ultimo U[]);
void aggiornaincasso(struct film F[], int n, struct ultimo U[], int m);

main()
{int i;
 struct film FILM[4];
 struct ultimo ULTIMOMESE[4];
 leggifilm(4,FILM);
 leggiultimo(4, ULTIMOMESE);
 aggiornaincasso(FILM,4, ULTIMOMESE,4);
 }

void leggifilm(int n, struct  film Vet[])
{int i;
 for(i=0;i<n;i++)
    {printf("Inserisci Codice, Titolo, Incasso \n");
     scanf("%d",&Vet[i].codice);
     scanf("%s",Vet[i].titolo);
      scanf("%d",&Vet[i].incasso);
    }
}

void leggiultimo(int n, struct ultimo Vet[])
{ int i;
for(i=0;i<n;i++)
    { printf("Codicefilm, incasso ultimo mese:\n");
      scanf("%d",&Vet[i].codicefilm);
         scanf("%d",&Vet[i].incassoultimo);
    }
}

void aggiornaincasso(struct film F[], int n, struct ultimo U[], int m)
{ int i,j;
for (i = 0; i < n; i++)
  for (j = 0; j < m; j++)
         if (U[i].codicefilm == F[j].codice)
            {F[j].incasso = F[j].incasso + U[i].incassoultimo;
              printf("incasso  aggiornato  del  film  %s:  %d",  F[j].titolo,
F[j].incasso);
            }
}
```