

Domanda 2Punteggio max.:
2,00Contrassegna
domanda

Sia data una matrice M di dimensione $r \times c$ contenente elementi interi.

Scrivere una funzione che generi una matrice M' di dimensione $r \times c$ derivata da M in cui ogni elemento $[i][j]$ assume il valore della somma cumulata di tutti gli elementi sulla medesima diagonale e antidiagonale, considerando solo elementi il cui indice di colonna sia maggiore o uguale all'elemento preso in considerazione. Il contributo dell'elemento $[i][j]$ originale è contato una singola volta.

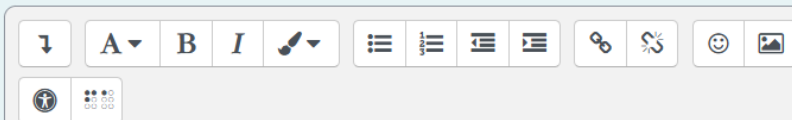
La matrice M' sia allocata dentro alla funzione.

Completare opportunamente il prototipo in modo che la nuova matrice sia disponibile al chiamante.

```
void f(int **M, int r, int c, ...);
```

Esempio:

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \Rightarrow M' = \begin{pmatrix} 15 & 8 & 3 \\ 14 & 17 & 6 \\ 15 & 14 & 9 \end{pmatrix}$$



Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

```
void f(int **M, int r, int c, ..) {  
  
}
```

Domanda 3Punteggio max.:
4,00Contrassegna
domanda

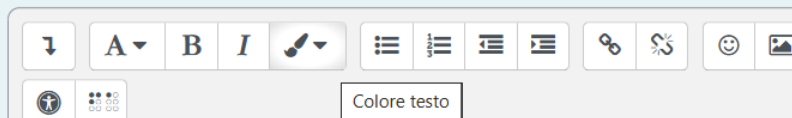
Si fornisca la definizione delle strutture dati `LIST` e `NODE`, come ADT di I categoria e quasi ADT rispettivamente, per rappresentare una lista singolo linkata di interi, senza sentinelle. Suddividere il codice in modo opportuno tra file `.h` e `.c`

Si scriva una funzione `void f(LIST l)` che ricevuta in input una lista di interi (rappresentata facendo riferimento ai tipi definiti in precedenza), compatti i contenuti della lista cancellando tutti i nodi il cui indice, ossia la cui posizione originale, sia divisibile per 3. La posizione di ogni nodo NON è salvata nel nodo stesso. Si assuma che la testa sia il nodo di posizione/indice 0.

Non è ammesso l'uso di funzioni di libreria.

Esempio:

$1 \rightarrow -2 \rightarrow 3 \rightarrow 8 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 9$ diventa $-2 \rightarrow 3 \rightarrow 5 \rightarrow 6 \rightarrow 9$



Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

```
LIST.h  
//LIST.h  
//scrivere qui il codice...  
  
LIST.c  
//LIST.c  
//scrivere qui il codice...
```

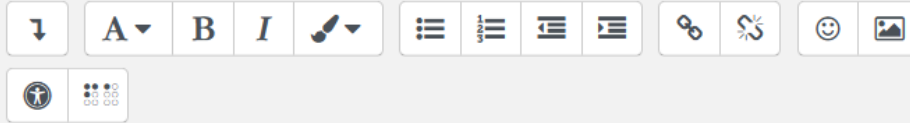
Domanda 4

Punteggio max.:
6,00

Contrassegna
domanda

Dato un vettore v di interi positivi, di dimensione nota d , identificare se possibile un modo di suddividerne il contenuto in x sottoinsiemi, con x parametro del problema, tale per cui la somma di tutti gli elementi di ognuno dei sottoinsiemi sia la stessa. Ogni elemento del vettore deve essere assegnato a un singolo sottoinsieme.

La ricerca, in caso di successo, deve terminare alla prima soluzione valida trovata.



Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

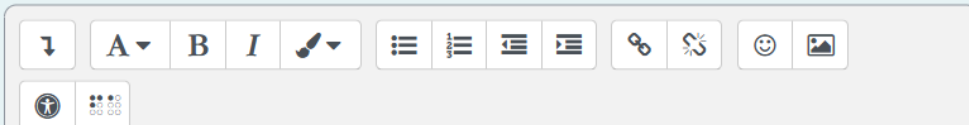
//scrivere qui il codice...

Domanda 5

Non valutata

Contrassegna
domanda

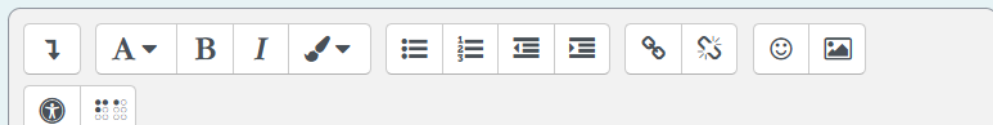
Si giustifichi la scelta del modello combinatorio adottato.

**Domanda 6**

Non valutata

Contrassegna
domanda

Si descrivano i criteri di pruning adottati o il motivo della loro assenza.



La prossima domanda rappresenta l'inizio della prova da 18 punti.

Informazione

Contrassegna domanda

Descrizione del problema

Una matrice quadrata di dimensione $N \times N$ rappresenta una griglia di gioco per uno "sliding puzzle".

La griglia contiene una serie di tessere semoventi e alcuni buchi.

La regola di movimento è la seguente: il giocatore ad ogni passo può scegliere una direzione (su/giù/destra/sinistra) e tutte le tessere libere di muoversi in quella direzione (c'è un buco nella cella destinazione) si muovono di conseguenza in contemporanea di UNA posizione. A ogni passo deve muoversi almeno una tessera, altrimenti la mossa non è valida (non cambierebbe lo stato del problema). Le tessere non possono muoversi uscendo dai bordi della griglia.

Tutte le tessere (ad eccezione di due, descritte a seguire) sono caratterizzate da avere due lati collegati da un canale.

Due tessere speciali, sorgente e destinazione, hanno un singolo lato di connessione. Scopo del gioco è muovere le tessere in modo da che esista un canale contiguo che connetta sorgente e destinazione.

Nota bene, sorgente e destinazione sono formalmente interscambiabili tra loro come ruolo. Non c'è una vera direzione di flusso tra le due.

Non è necessario coinvolgere tutte le tessere nella creazione del canale complessivo, l'importante è che sorgente e destinazione siano connesse.

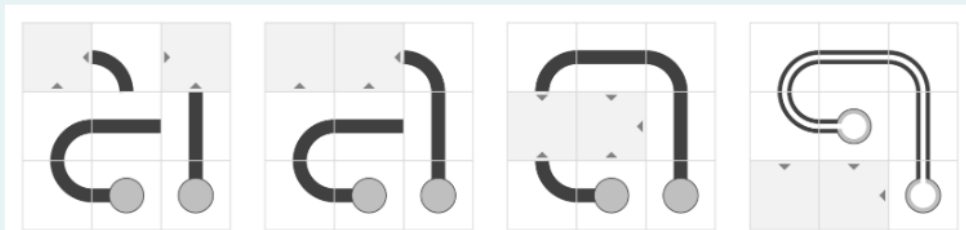
Il giocatore ha a disposizione al massimo M mosse, parametro del problema.

Esempio

Nell'esempio proposto è rappresentata una griglia 3×3 con due buchi (inizialmente, angolo alto a sinistra e alto a destra), cinque tessere standard (due lati comunicanti) e le due tessere terminali (sorgente/destinazione tonda grigia e singolo lato aperto).

Le frecce grigio scuro ai bordi di tessere confinanti con un buco evidenziano direzioni di movimento ammesse per ogni tessera, e di conseguenza tutte le tessere coinvolte da un movimento in quella direzione.

Nell'esempio proposto, la sequenza di movimenti DESTRA, SU, SU porta alla soluzione del puzzle. Si ricorda che la singola mossa muove tutte le tessere libere di spostarsi nella direzione specificata.



Richieste del problema

A seguire una sintesi delle richieste del problema. Per ogni richiesta si troverà una domanda dedicata nelle sezioni a seguire con una descrizione più dettagliata per le richieste.

Strutture dati e letture

Definire opportune strutture dati per rappresentare i dati del problema e tutte le strutture dati ausiliarie ritenute opportune per la risoluzione dei problemi di verifica e di ricerca. Definire inoltre le funzioni di lettura secondo il formato dei file descritti nelle domande a seguire.

Problema di verifica

Data una proposta di sequenza di mosse, valutare che questa rispetti le regole indicate in precedenza e se sia in grado, o meno, di risolvere il puzzle.

Problema di ottimizzazione

Identificare, se possibile, una sequenza di mosse valide in grado di risolvere il puzzle.

Domanda 7

Non valutata

Contrassegna domanda

Strutture dati e acquisizione

Scrivere qui la definizione e implementazione delle strutture dati repute necessarie a modellare le informazioni del problema, e le funzioni di acquisizione dei dati stessi. In caso di organizzazione delle strutture dati su più file, indicare esplicitamente il modulo di riferimento.

Si assuma che lo schema di gioco in input sia riportato in un file di nome `grid.txt`, organizzato come segue:

- Sulla prima riga appare una coppia di interi N e T dove N è la dimensione del lato della griglia e T il numero di tessere presenti
- Seguono $N \times N$ righe ognuna caratterizzata da una quaterna di valori 0/1 a indicare i lati connessi dal canale nell'ordine nord-sud-ovest-est, a rappresentare la griglia di gioco in rappresentazione row-major
- Righe dove appaiono due zeri sono tessere normali, in cui gli 1 rappresentano i lati comunicanti
- Righe dove appare un singolo 1 sono tessere sorgente/destinazione
- Righe in cui appare una quaterna di zeri sono buchi.

Con riferimento all'esempio presentato in precedenza, la configurazione iniziale della griglia corrisponde a un file i cui contenuti sono i seguenti. Nota bene, i commenti sono a mero scopo descrittivo, e non fanno parte del file.

```
3 7
0 0 0 0 // Buco
0 1 1 0 // Tessera con connessione SUD-OVEST
0 0 0 0 // Buco
0 1 0 1 // Tessera con connessione SUD-EST
0 0 1 1 // Tessera con connessione OVEST-EST
1 1 0 0 // Tessera con connessione NORD-SUD
1 0 0 1 // Tessera con connessione NORD-EST
0 0 1 0 // Sorgente/Destinazione con connessione a OVEST
1 0 0 0 // Sorgente/Destinazione con connessione a NORD
```

↵

A ▾

B

I

✎ ▾

☰

☷

☱

☲

🔗

🔄

😊

🖼

🌐


🔍

Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

...


Domanda 8

Non valutata

 Contrassegna domanda**Problema di verifica**

Data una soluzione proposta rappresentata da una sequenza di movimenti, verificare che questa rappresenti una sequenza valida tenendo conto delle regole di movimento espresse in precedenza. Valutare inoltre se questa sequenza sia in grado di risolvere o meno il puzzle.

La soluzione proposta deve essere letta da file il cui nome e formato è a discrezione del candidato, che è tenuto a fornire anche una breve spiegazione dei contenuti del file stesso.





A


▼


B


I





























Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.


...

Domanda 9

Non valutata

 Contrassegna domanda**Problema di ricerca e ottimizzazione**

Identificare, se possibile, una sequenza di movimenti in grado di risolvere il puzzle, tenendo conto delle regole di movimento espresse in precedenza. Si ricorda che non è necessario fare uso di tutto le tessere: l'obiettivo è solo quello di creare un canale continuo tra sorgente/destinazione.





A


▼


B


I





























Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

...

PARTE TEORIA (12 PUNTI)

Domanda 1

Punteggio max.:
2,00

Contrassegna
domanda

Sia data la sequenza di interi, supposta memorizzata in un vettore A:

211, 126, 99, 46, 44, 127, 313, 91, 32, 37, 19, 53

si eseguano i primi 2 passi dell'algoritmo di quicksort per ottenere un ordinamento ascendente, indicando ogni volta il pivot scelto. NB: i passi sono da intendersi, impropriamente, come in ampiezza sull'albero della ricorsione, non in profondità. Si chiede, pertanto, che siano ritornate le due partizioni del vettore originale e le due partizioni delle partizioni trovate al punto precedente.

↴

A ▾

B

I

✎ ▾

☰

☰

☰

☰

🔑

🔄

😊

🖼

👤

👥

Formato della risposta:

Elemento singolo oppure vettore come sequenza di interi separati da una virgola (,)

Pivot al primo passo:

Vettore A dopo il primo passo:

Pivot del sottovettore sinistro al secondo passo:

Pivot del sottovettore destro al secondo passo:

Vettore A dopo il secondo passo:

Domanda 2

Punteggio max.:
2,00

Contrassegna
domanda

Sia data la sequenza di chiavi intere

211, 26, 79, 46, 154, 17, 43, 229

Si riporti il contenuto di una tabella di hash inizialmente supposta vuota, in cui avvenga l'inserimento della sequenza indicata. Si usi l'open addressing con linear probing. Si determini la dimensione minima M della tabella di per avere un fattore di carico $\alpha < \frac{1}{2}$.

↴

A ▾

B

I

✎ ▾

☰

☰

☰

☰

🔑

🔄

😊

🖼

👤

👥

Formato della risposta:

Dimensione minima della tabella

M =

Collisioni chiave 43:

collisione 1 alla cella di indice:

.....

collisione n alla cella di indice:

Collisioni chiave 230:

collisione 1 alla cella di indice:

.....

collisione n alla cella di indice:

Contenuto della cella all'indice 7:

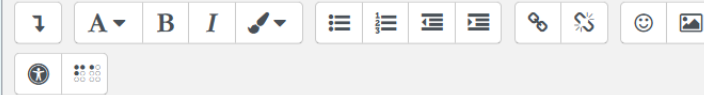
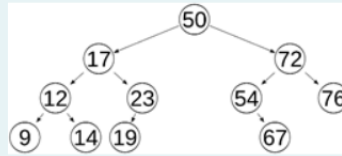
Dimensione del primary cluster più grande:

Domanda 3

Punteggio max.:
2,00

Contrassegna
domanda

Si inseriscano in **radice** nel BST di figura in sequenza le chiavi 45 e 10 poi si cancelli la chiave 50.



Dopo l'inserzione in radice di 45:

figlio sinistro di 17

figlio destro di 17

Dopo l'inserzione in radice di 10:

figlio sinistro di 10

figlio destro di 10

Dopo la cancellazione di 50:

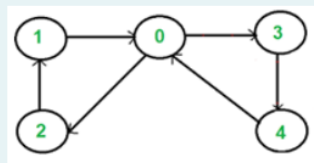
figlio sinistro di 54

Domanda 4

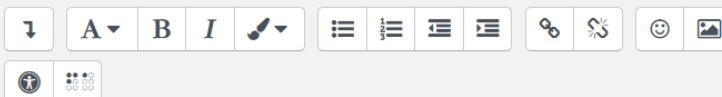
Punteggio max.:
3,50

Contrassegna
domanda

Sia dato il seguente grafo orientato:



se ne effettui una visita in profondità, considerando **0** come vertice di partenza etichettando i vertici con tempo di scoperta/tempo di fine elaborazione (**2 punti**) e gli archi con T, B, F, C (**1.5 punti**). Qualora necessario, si trattino i vertici secondo l'ordine numerico.



Tempo di scoperta / tempo di fine elaborazione

vertice 0:

vertice 1:

vertice 2:

vertice 3:

vertice 4:

Etichettatura archi

arco 0-2:

arco 0-3:

arco 1-0:

arco 2-1:

arco 3-4:

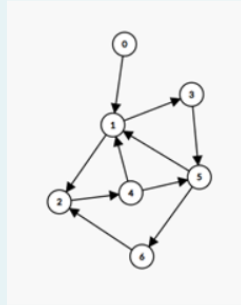
arco 4-0:

Domanda 5

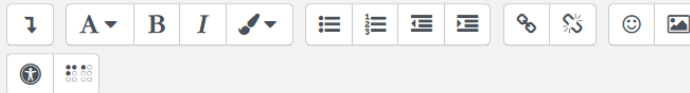
Punteggio max.:
2,50

Contrassegna
domanda

Dato il seguente grafo orientato:



si applichi l'algoritmo di Kosaraju per il calcolo delle componenti fortemente connesse, considerando **0** come vertice di partenza e, qualora necessario, trattando i vertici secondo l'ordine numerico.



tempo di scoperta / tempo di fine elaborazione del vertice 4 nella visita del grafo trasposto

vertice da cui parte la visita del grafo originale

SCC1 =

SCC2 =

....

SCCn =