03AAX Algoritmi e Strutture Dati / 03MNO Algoritmi e Programmazione

Appello del 09/02/2022 - Prova di teoria (12 punti)

1. (2 punti)

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

9 2 4 33 3 69 13 19 7 41 0 5 6 8

- La si trasformi in un heap, ipotizzando di usare un vettore come struttura dati. Si riportino graficamente i passi significativi della costruzione dell'heap ed il risultato finale. Si ipotizzi che, alla fine, nella radice dell'heap sia memorizzato il valore massimo
- Si eseguano su tale heap i primi 2 passi dell'algoritmo di heapsort.

NB: la sequenza è già memorizzata nel vettore e rappresenta una configurazione intermedia per cui la proprietà di heap non è ancora soddisfatta.

2. (2.5 punti)

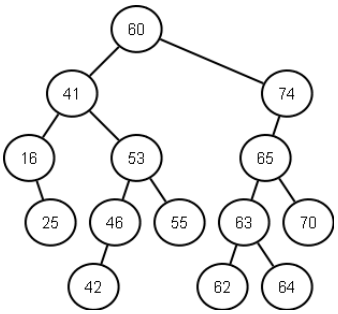
Data la catena di matrici (A_1, A_2, A_3, A_4) di dimensioni (4x6), (6x2), (2x8) e (8x3) rispettivamente, si determini mediante un algoritmo di programmazione dinamica la parentesizzazione ottima del prodotto di matrici che minimizza il numero di moltiplicazioni.

3. (1 punto)

Si converta la seguente espressione da forma prefissa a forma infissa: / * - A B / C D / E * - F G + H I

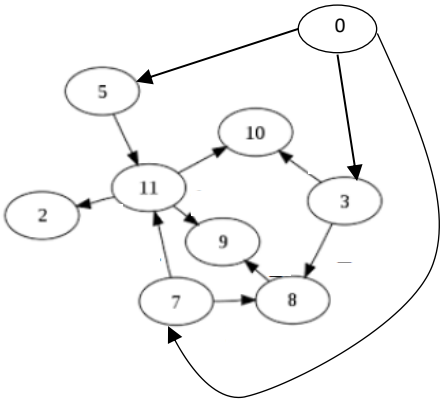
4. (2 punti)

Si partizioni il seguente BST attorno alla sesta chiave più piccola. In ogni nodo compare la chiave intera, altre eventuali informazioni non sono riportate.



5. (2.5 punti)

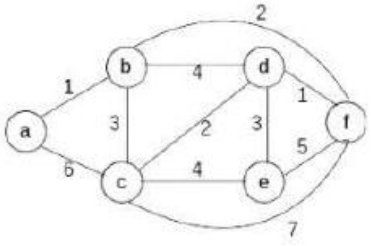
Dato il seguente grafo orientato:



si applichi l'algoritmo di Kosaraju per il calcolo delle componenti fortemente connesse, considerando **11** come vertice di partenza e, qualora necessario, trattando i vertici secondo l'ordine numerico.

6. (2 punti)

Dato il seguente grafo non orientato, connesso e pesato, se ne determini un minimum spanning tree applicando l'algoritmo di Prim a partire dal vertice **a**, disegnando l'albero e ritornando come risultato il valore del peso minimo. Si esplicitino i passi intermedi.

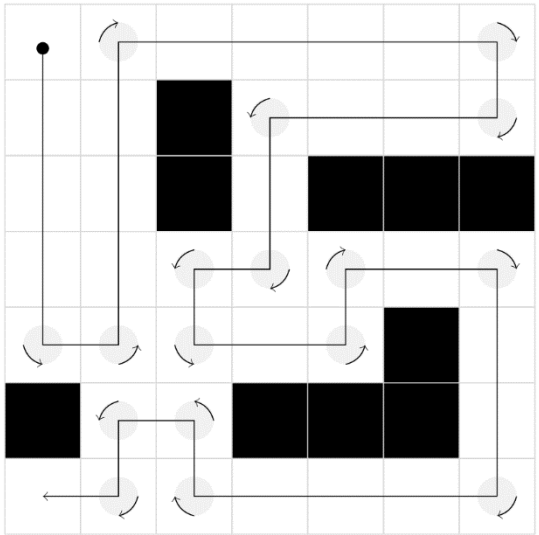


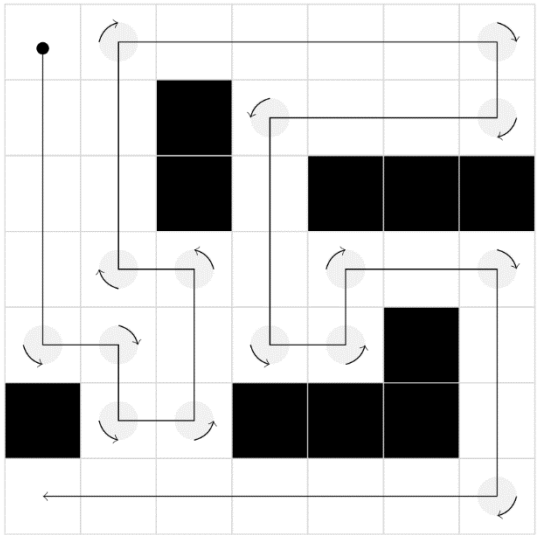
03AAX Algoritmi e Strutture Dati
03MNO Algoritmi e Programmazione
Appello del 09/02/2022 - Prova di programmazione (18 punti)

1. **(18 punti)**
Una matrice rettangolare `R x C` rappresenta una griglia di gioco.
La griglia contiene celle bianche e celle nere. Le celle bianche sono celle su cui è possibile transitare mentre le celle nere sono inaccessibili e rappresentano degli ostacoli sulla griglia di gioco.
La cella in posizione (0,0) è sempre bianca ed è sempre il punto di partenza del cammino.
L'obiettivo del gioco è visitare tutte le celle bianche della griglia, se possibile, percorrendo un cammino semplice che esegua il minor numero possibile di cambi di direzione. La cella finale di destinazione non è definita a priori. Un cammino, purché corretto, può finire ovunque. Il movimento è possibile solo lunghe le quattro direzioni principali (nord, sud, ovest, est): non sono ammessi spostamenti in diagonale.
Si scriva un programma in C che:

- legga un primo file di testo `griglia.txt` organizzato come segue:
  - la prima riga contiene una coppia di interi `NR NC`, ossia le dimensioni della griglia
  - seguono `NR` righe di `NC` valori interi separati da spazi, ciascuno a rappresentare una cella bianca (0) o nera (1)
- legga un secondo file di testo `proposta.txt`, il cui formato è a discrezione del candidato, a rappresentare una possibile sequenza di mosse sulla griglia e determini se sia una soluzione valida secondo le regole di cui sopra. In caso affermativo conteggi il numero di cambi di direzione
- trovi una visita della griglia rispettando i vincoli di cui sopra effettuando il minor numero possibile di cambi di direzione

Esempio.
Le figure seguenti rappresentano due possibili soluzioni per la stessa griglia di gioco `7 x 7`. L'immagine a sinistra propone un cammino semplice che transita per tutte le celle bianche eseguendo 17 cambi di direzione. L'immagine a destra presenta un secondo cammino valido che usa il minimo numero di cambi di direzione: 15.





03AAX Algoritmi e Strutture Dati
03MNO Algoritmi e Programmazione
Appello del 09/02/2022 - Prova di programmazione (12 punti)

1. (**2 punti**)
Sia data una matrice `M` di dimensione `r x c` contenente elementi interi positivi o nulli.
Scrivere una funzione `f` che generi una matrice `M'` di dimensione `r' x c'` derivata da `M` mantenendo solo le righe/colonne dove entrambi gli indici sono pari. Per convenzione gli indici partono da zero (pari).
Completare opportunamente il prototipo in modo che la nuova matrice e le relative dimensioni siano disponibili al chiamante.

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

Esempio. Siano `r` = 3, `c` = 4

$$M = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 0 & 1 & 2 \end{pmatrix} \rightarrow M' = \begin{pmatrix} 1 & 3 \\ 9 & 1 \end{pmatrix}$$

2. (**4 punti**)
Si scriva una funzione (wrapper) `char *decode(H h, char *str)` che, ricevuto in input un albero binario `h`, rappresentante la codifica di Huffman associata a un certo set di caratteri, e una stringa di caratteri `str` che contiene una sequenza di 0/1, la decodifichi, sulla base della codifica memorizzata nell'albero `h`, ritornando come risultato una stringa decodificata.
Fornire inoltre la definizione del tipo `H` (come ADT di prima classe) e del tipo nodo al suo interno (come quasi ADT).
Non è ammesso l'utilizzo di funzioni di libreria.

3. **(6 punti**)
Una matrice `M` quadrata di dimensione `N x N` rappresenta le relazioni di amicizia tra `N` persone. Ogni cella contenente il valore 1 indica che la coppia *(i,j)* è una coppia di amici. In caso contrario il valore memorizzato è 0. La relazione di amicizia è simmetrica (se *i* considera *j* suo amico, vale anche l'opposto).

- Scrivere una funzione ricorsiva in grado di individuare il più grande gruppo di persone tale per cui ogni persona è amica con almeno altre k persone (diverse dalla persona stessa) del gruppo.
- Indicare esplicitamente il modello combinatorio utilizzato e giustificarne la scelta.
- Descrivere e giustificare i criteri di pruning utilizzati o la loro eventuale assenza.

Esempio. Sia `N` = 4, k=2

$$M = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix} \rightarrow g = \{p_0, p_1, p_3\}$$

PER ENTRAMBE LE PROVE DI PROGRAMMAZIONE (18 o 12 punti):

- ***indicare nell'elaborato e nella relazione nome, cognome e numero di matricola.***
- ***se non indicato diversamente, è consentito utilizzare chiamate a funzioni <u>standard</u>, quali ordinamento per vettori, funzioni su FIFO, LIFO, liste, BST, tabelle di hash, grafi e altre strutture dati, considerate come librerie esterne.***
- ***gli header file devono essere allegati all'elaborato (il loro contenuto riportato nell'elaborato stesso). Le funzioni richiamate, inoltre, dovranno essere incluse nella versione del programma allegata alla relazione. I modelli delle funzioni ricorsive non sono considerati funzioni standard.***
- ***consegna delle relazioni (per entrambe le tipologie di prova di programmazione): entro il 12/02/2022, alle ore 12:00, mediante caricamento su Portale. Le istruzioni per il caricamento sono pubblicate sul Portale nella sezione Materiale). QUALORA IL CODICE CARICATO CON LA RELAZIONE NON COMPILI CORRETTAMENTE, VERRÀ APPLICATA UNA PENALIZZAZIONE. Si ricorda che la valutazione del compito viene fatta, senza la presenza del candidato, <u>sulla base dell'elaborato svolto in aula</u>. Non verranno corretti i compiti di cui non sarà stata inviata la relazione nei tempi stabiliti.***