Cognome: **Nome**: **# Matricola**: **Riga**: **Col**:

Algoritmi e Strutture Dati - Parte 1 - 17/06/2019

Esercizio -1 Iscriversi allo scritto entro la scadenza. In caso di inadempienza, -1 al voto finale.

Esercizio 0 Scrivere correttamente nome, cognome, numero di matricola, riga e colonna su tutti i fogli consegnati. Consegnare foglio A4 e foglio protocollo di bella. In caso di inadempienza, -1 al voto finale.

Esercizio A1 – Punti ≥ 6

Trovare i limiti superiore e inferiori più stretti possibili per la seguente equazione di ricorrenza:

$$T(n) = \begin{cases} 2T(\lfloor n/3 \rfloor) + 3T(\lfloor n/2 \rfloor) + n^2 & n \geq 3 \\ 1 & n < 3 \end{cases}$$

Esercizio A2 – Punti ≥ 13

Sia T un albero binario, in cui ogni nodo u è associato ad un peso $u.weight$ intero.

Ovviamente, ogni coppia di foglie f_1, f_2 è connessa da un cammino semplice che passa attraverso il primo antenato in comune. Se $f_1 = f_2$, il cammino è composto dalla foglia stessa.

Il costo di un cammino è dato dalla somma dei pesi dei nodi che lo compongono (estremi inclusi).

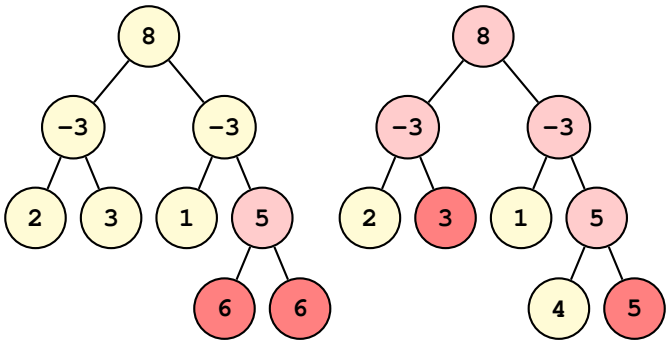
Scrivere un algoritmo

int maxPath(TREE T)

che preso in input un albero T restituisca il costo massimo fra tutti i cammini semplici che connettono due foglie qualsiasi dell'albero.

Discutere informalmente la correttezza dell'algoritmo e calcolare la sua complessità computazionale.

Negli esempi seguenti, i cammini semplici massimali sono rappresentati in rosa, con le foglie leggermente più scure. A sinistra, l'algoritmo deve restituire 17; a destra, deve restituire 15.



Esercizio A3 – Punti ≥ 13

La NASA ha a disposizione una mappa altimetrica M di una certa zona di Marte, suddivisa in n righe per n colonne. Ogni valore $M[r][c]$ rappresenta l'altitudine della cella (r, c) rispetto ad un livello prefissato.

La NASA vuole creare dei mari, riempiendo di acqua ogni cella (r, c) tale che $M[r][c] < 0$. Due celle con altitudine inferiore a zero appartengono allo stesso mare se sono adiacenti in orizzontale o verticale (non in diagonale).

La *linea costiera* di un mare è data da tutte le celle (r, c) tali che $M[r][c] \geq 0$ e (r, c) è adiacente in orizzontale o verticale (non diagonale) ad una cella di quel mare.

Il vostro compito è scrivere un algoritmo

int coastLen(**int**[][] M, **int** n, **int** r, **int** c)

che data una mappa di Marte M, la sua dimensione n, e una coppia di coordinate (r, c) tale che $M[r][c] < 0$, restituisca il numero di celle contenute nella linea costiera del mare che si verrà a creare attorno alla posizione (r, c).

Discutere informalmente la correttezza dell'algoritmo e calcolare la sua complessità computazionale.

Esempio: in questa mappa ci sono due mari, colorati in azzurro. La linea costiera del mare associato alla cella $(2, 4)$ è colorata in arancio e consta di 18 celle.

	1	2	3	4	5	6	7
1	1	2	3	4	5	6	1
2	3	-2	-3	-3	-1	3	3
3	4	-2	1	-2	-3	4	5
4	2	-1	2	-1	-1	7	7
5	1	2	2	-1	1	-3	-2
6	4	4	4	-1	2	-2	-3
7	4	4	4	4	3	2	2

Cognome: **Nome**: **# Matricola**: **Riga**: **Col**:

Algoritmi e Strutture Dati - Parte 2 - 17/06/2019

Esercizio -1 Iscriversi allo scritto entro la scadenza. In caso di inadempienza, -1 al voto finale.

Esercizio 0 Scrivere correttamente nome, cognome, numero di matricola, riga e colonna su tutti i fogli consegnati. Consegnare foglio A4 e foglio protocollo di bella. In caso di inadempienza, -1 al voto finale.

Esercizio B1 – Punti ≥ 8

Un'impresa edile deve gestire c cantieri in m mesi numerati da 1 a m.

- Il cantiere i-esimo inizia nel mese $start[i]$ e deve finire entro il mese $end[i]$.
- Ogni cantiere richiede $len[i]$ mesi/persona per essere completato. Per esempio, se una persona lavora per 3 mesi e un'altra per 4 mesi, sono stati impiegati 7 mesi/persona in totale.
- Ogni mese sono disponibili p_m persone.
- Per come sono strutturati i cantieri, durante ogni mese non è possibile far lavorare più di p_c persone alla volta nello stesso cantiere.

Descrivere un algoritmo che prenda in input i dati descritti sopra e restituisca vero se è possibile completare tutti i cantieri in tempo, falso altrimenti.

Discutere informalmente la correttezza dell'algoritmo e calcolare la sua complessità computazionale.

Esercizio B2 – Punti ≥ 12

Una stringa è k-palindroma se rimuovendo al massimo k caratteri si riesce ad ottenere una stringa palindroma.

Ad es., `"DISillusaSfingosognifasulli"` è k-palindroma per $k \geq 4$ (rimuovendo `"DIS"` e `"S"`).

Scrivere un algoritmo

boolean k-palindrome(ITEM[ ] S, **int** n, **int** k)

che restituisca **true** se la stringa S di n caratteri è k-palindroma, **false** altrimenti.

Discutere informalmente la correttezza dell'algoritmo e calcolare la sua complessità computazionale. In particolare, si evidenzi se l'algoritmo proposto è polinomiale, pseudopolinomiale o superpolinomiale.

Esercizio B3 – Punti ≥ 12

Coach Malone è alla prese con una classe di ragazzi particolarmente turbolenti. Dopo tre anni di scuola media, si sono create amicizie e inimicizie. Sia $G = (V, E)$ il grafo non orientato delle amicizie, dove V rappresenta l'insieme dei ragazzi ed $(greg, rowley) \in E$ significa che $greg$ e $rowley$ sono amici.

Ora, bisogna mettere i ragazzi in un'unica fila per la mensa, ovvero stabilire un ordinamento fra tutti i nodi V. Al primo posto andrà il capoclasse, u.

Sfortunatamente, se due bambini che non sono amici si trovano in fila uno dopo l'altro, inizieranno a menarsi e Coach Malone dovrà intervenire.

Coach Malone vi ha chiesto di scrivere un algoritmo:

int[ ] safeLine(GRAPH G, **int** u)

che prenda in input un grafo delle amicizie G, il capoclasse u e restituisca un qualsiasi ordinamento totale che inizi con il capoclasse e che non richieda il suo intervento, oppure **nil** se tale ordinamento non esiste.

Discutere informalmente la correttezza dell'algoritmo e calcolare la sua complessità computazionale.