02MNO Algoritmi e Programmazione 01JKE APA I / 01JKF APA II

Appello del 23/02/2015 - Prova di teoria (12 punti)

1. (**2 punti**)
Si risolva la seguente equazione alle ricorrenze mediante il metodo dello sviluppo (unfolding):

$$T(n) = 4T(n/3) + n \qquad n \geq 2$$
$$T(1) = 1$$

2. (1 punto)
Si ordini in maniera ascendente mediante counting-sort il seguente vettore di interi:

5 2 7 0 8 3 6 0 10 1 2 1 10 0 4

Si indichino le strutture dati usate nei passi intermedi.

3. (2 punti)
Sia data la sequenza di interi, supposta memorizzata in un vettore:

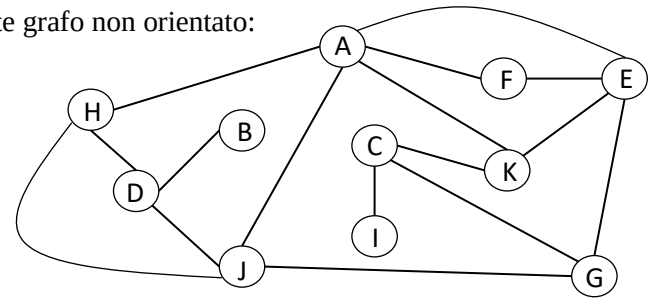
22 39 19 21 31 43 32 23 25 29 35 40

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 2 partizioni del vettore originale e le due partizioni delle partizioni trovate al punto precedente.

4. (**2 punti**)
Si effettuino, secondo l'ordine specificato, le seguenti inserzioni in foglia su un Interval BST supposto inizialmente vuoto:

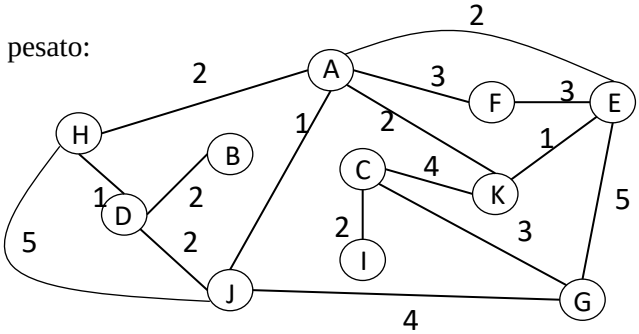
[3,10] [4, 6] [1,3] [16,21] [7,11] [2,8] [12,19] [5,15] [9,10]

5. (1 punto)
Si determinino i punti di articolazione del seguente grafo non orientato:



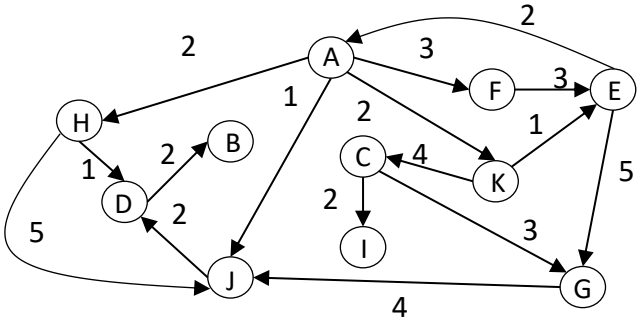
Si consideri **A** come vertice di partenza e, qualora necessario, si trattino i vertici secondo l'ordine alfabetico.

6. (2 punti)
Sia dato il seguente grafo non orientato pesato:



se ne determini un minimum spanning tree applicando l'algoritmo di Kruskal, disegnando l'albero e ritornando come risultato il valore del peso minimo. Si esplicitino i passi intermedi.

7. (2 punti)
Sul seguente grafo orientato e pesato, si determinino i valori di tutti i cammini minimi che collegano il vertice **A** con ogni altro vertice mediante l'algoritmo di Dijkstra. Si assuma, qualora necessario, un ordine alfabetico per i vertici e gli archi.



02MNO Algoritmi e Programmazione 01JKE APA I / 01JKF APA II

Appello del 23/02/2015 - Prova di programmazione (18 punti)

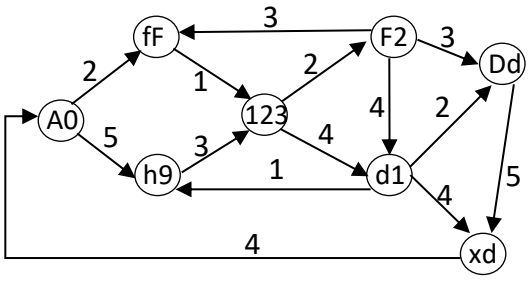
Un grafo orientato pesato è memorizzato su un primo file mediante l'elenco dei suoi archi, con il seguente formato:

idV_1 val idV_2

che indica che il vertice con identificatore idV1 è connesso con un arco orientato di peso val (valore intero e positivo) al vertice idV2.
Poiché il numero di vertici del grafo non è noto a priori, si suggerisce di calcolarlo tramite lettura preliminare del file e/o caricamento in una tabella di simboli. Ogni vertice è individuato mediante un identificatore alfanumerico di lunghezza massima uguale a 20 caratteri. Si può assumere che non esistano archi duplicati. Non è lecito assumere nessuna forma di ordinamento degli archi.

Esempio: contenuto del primo file e grafo corrispondente:

```
fF 1 123
A0 2 fF
A0 5 h9
h9 3 123
123 2 F2
123 4 d1
F2 3 Dd
F2 4 d1
d1 2 Dd
d1 4 xd
d1 1 h9
Dd 5 xd
xd 4 A0
F2 3 fF
```



Su un secondo file sono memorizzati 2 cammini **semplici** con il seguente formato: sulla prima riga compare la lunghezza del primo cammino, seguita dai vertici che lo compongono separati da spazi o caratteri a-capo. Analogamente per il secondo cammino.

Esempio: contenuto del secondo file

```
4
A0 fF 123 d1 xd
5
A0 h9 123 F2 d1 Dd
```

Si scriva un programma C in grado di:

- ricevere i nomi dei due file sulla riga di comando
- leggere il grafo dal primo file e memorizzarlo in un'opportuna struttura dati
- leggere i 2 cammini semplici dal secondo file, identificare i vertici in comune (intersezione dei 2 cammini) e visualizzare ciascuno dei 2 cammini decomposto in sottocammini determinati dai vertici in comune
- data una coppia di vertici sorgente/destinazione letta da tastiera, dati 2 interi k e p letti da tastiera ($k \leq p$), stampare il **cammino ottimo non necessariamente semplici** tra i vertici sorgente e destinazione che soddisfi i seguenti vincoli:
  - è massima la somma dei pesi degli archi del cammino
  - sono riattraversati al più k vertici
  - il numero complessivo di riattraversamenti è al massimo p
  - una volta raggiunto il nodo di destinazione, il cammino è da considerarsi terminato.

In riferimento all'esempio, i 2 cammini hanno in comune i vertice `A0`, `123 e d1`, quindi si dovrà visualizzare:

```
i vertici in comune sono:
A0
123
d1
```

```
il cammino 1 si decompone in 3 sottocammini:
sottocammino 1.1: A0, fF, 123
sottocammino 1.2: 123, d1
sottocammino 1.3: d1, xd

il cammino 2 si decompone in 3 sottocammini:
sottocammino 2.1: A0, h9, 123
sottocammino 2.2: 123, F2, d1
sottocammino 2.3: d1, Dd
```

Ad esempio, indicando `A0` e `fF` come sorgente e destinazione, rispettivamente, con k=1 e p=1, il cammino massimo vale 27 e corrisponde a: `A0 h9 123 F2 d1 Dd xd A0 fF` (si può riattraversare solo un nodo e una sola volta).
Mantenendo i medesimi nodi come sorgente e destinazione, ma utilizzando k=6 e p=7, si ottiene un cammino di valore 50, corrispondente a: `A0 h9 123 F2 d1 Dd xd A0 h9 123 d1 Dd xd A0 fF`

02MNO Algoritmi e Programmazione 01JKE APA I / 01JKF APA II

Appello del 23/02/2015 - Prova di programmazione (12 punti)

1. (**2 punti**)

Si realizzi una funzione C `invertSequence` con il seguente prototipo:

```
void invertSequence (int *v1, int n, int *v2);
```

La funzione riceve come parametri 2 vettori di interi `v1` e `v2` e la loro dimensione `n` e memorizza in `v2` lo stesso insieme di numeri presente in `v1` ma con tutte le sottosequenze ascendenti in `v1` trasformate in sottosequenze discendenti in `v2`.

Ad esempio, se inizialmente `v1` contiene `1 2 3 4 5 0 12 13 14 2`, `v2` al termine dovrà contenere `5 4 3 2 1 14 13 12 0 2`.

2. (**4 punti**)

Sia dato un albero binario i cui nodi sono definiti dalla seguente struttura C:

```
struct node {
  int  key;
  struct node *left;
  struct node *right;
};
```

Si realizzi una funzione C

```
void printPaths (struct node *root, int h, ...);
```

che, dato un albero di altezza `h`, visualizzi tutti i cammini radice-foglie elencandone le chiavi. Si noti che ogni cammino deve essere visualizzato completamente da radice a foglia, quindi il cammino percorso in ogni momento deve essere passato tra chiamate ricorsive. Se necessario, è possibile aggiungere parametri alla funzione `printPath`.

3. (**6 punti**)

Una password è una stringa di 5 elementi:

- i primi 3 sono lettere dell'alfabeto inglese maiuscolo e gli ultimi 2 sono cifre della base 10
- la stessa lettera o cifra può comparire al più k volte, dove k è un intero letto da tastiera

Si scriva una funzione ricorsiva in C in grado di generare tutte le password secondo le regole di cui sopra e di scriverle su un file di uscita, il cui nome è passato come parametro nella riga di comandi.

Esempio: con k=2 alcune delle password generate sono: AAB11, ZDZ09, ABC34

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

- ***indicare nell'elaborato e nella relazione (oltre a nome, cognome e numero di matricola) anche il nome del corso per cui si sta sostenendo l'esame (AP, APA I+II).***
- ***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 delle librerie utilizzate devono essere allegati all'elaborato. 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 giovedì 26/02/2015, alle ore 23:59, mediante caricamento su Portale. Le istruzioni per il caricamento sono pubblicate sul Portale nella sezione Materiale). QUALORA IL CODICE SPEDITO CON LA RELAZIONE NON COMPILI CORRETTAMENTE, VERRÀ APPLICATA UNA PENALIZZAZIONE. Si ricorda che la valutazione del compito viene fatta senza discussione o esame orale, <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.***

SCADENZE PER LAUREANDI Viste le scadenze per superare gli esami e presentare domanda di laurea, per I SOLI LAUREANDI le tempistiche del II appello sono le seguenti: - martedì 24 febbraio ore 23.59: termine per la consegna della relazione mediante caricamento su Portale. L'archivio deve avere nome sXXXXXX_230215_LAUREANDO - venerdì 27 febbraio ore 9.00 aula 23: esami orali.