Progettazione di Algoritmi (9 CFU)
15/7/2024

Cognome e Nome:
Numero di Matricola:

Spazio riservato alla correzione

1	2	3	4	5	Totale
/20	/22	/18	/20	20	/100

Attenzione: non sara` valutato lo pseudocodice in cui i blocchi non sono evidenziati da una chiara indentazione. Non e` sufficiente racchiudere i blocchi tra parentesi o tra coppie di parole come ad esempio begin/end.

1. **Analisi degli algoritmi e notazione asintotica**
   a) Indicare quali delle seguenti affermazioni sono vere e quali sono false.
      1. $n^{3\log n} = O(3^n + n\log n)$
      2. $\log n + n^{1/2} = \Omega(n^{1/3}\log n)$
      3. $4^{\log n} = O(n)$, la base del log e` 2
      4. $n^5+10n+8=O(n^4)$
      5. $\log(\log^n n)=O((\log n)(\log n))$

   b) Dimostrare che la seguente affermazione e` vera giustificando la risposta. Occorre fornire le costanti c ed n_0 (valori numerici).
   $5n^3 + n=O(n^3)$

c) Si dimostri che se $0<f(n)=O(h(n))$ e $0<g(n)= O(p(n))$ e **a** e` una costante positiva allora **a**$f(n)+g(n)= O(h(n)+p(n))$. Occorre utilizzare solo la definizione di O e nessuna altra proprieta`.

d) Si analizzi il tempo di esecuzione nel caso pessimo del seguente segmento di codice fornendo una stima asintotica **quanto migliore è possibile** per esso. **Si giustifichi in modo chiaro la risposta.**

```
FOR(i=1; i<m; i=i*2){
    FOR(j=1; j<2^n; j=j*2){
        Print(j);
    }
}
```

2. Grafi

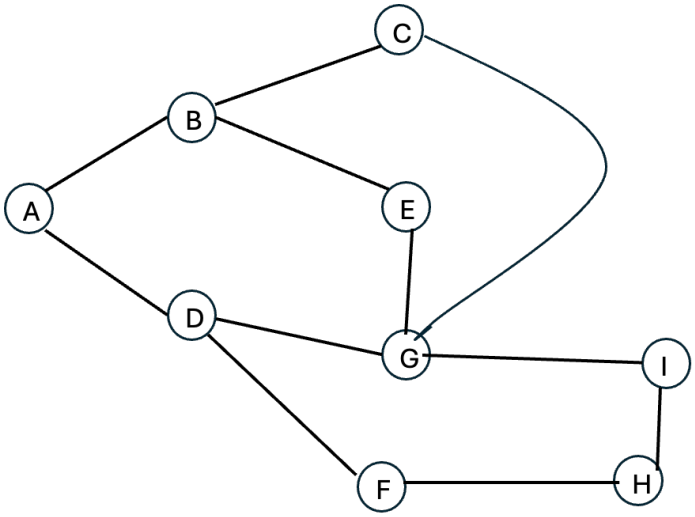
a) Si scriva lo pseudocodice di un algoritmo **ricorsivo** che prende in input un grafo G direzionato aciclico e restituisce l'ordinamento topologico di G. Scrivere l'algoritmo in modo che abbia tempo O(n+m). Un algoritmo con tempo di esecuzione maggiore sara` valutato con un punteggio piu` basso. **Scrivete un'unica versione dell'algoritmo (due o piu` versioni saranno valutate 0 punti).**

b)

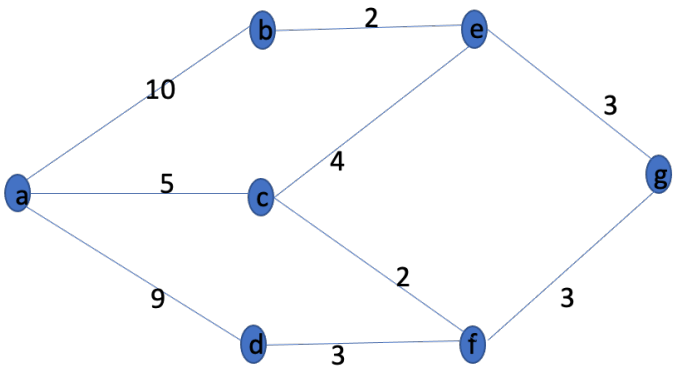
i. Si definisca in modo chiaro che cosa e` un grafo bipartito

ii. Si indichi sul seguente disegno di che colore viene colorato ciascun nodo del grafo dall'algoritmo che verifica se il grafo è bipartito.

iii. Si dica se il grafo è bipartito o meno e, nel caso in cui non lo sia, si fornisca una prova del fatto che il grafo non e` bipartito spiegando come trovarla. La prova deve poter convincere una persona che non conosce l'algoritmo e non sa come sono colorati i nodi.



b) Si mostri l'esecuzione dell'algoritmo di Prim sul seguente grafo in modo che la radice dell'albero sia a. **Per ogni passo si mostri il contenuto della coda a priorita` indicando anche la chiave associata a ciascun nodo e l'albero costruito fino a quel passo. Si dica se il minimo spanning tree e` unico giustificando la risposta data.**



Progettazione di Algoritmi (9 CFU)
15/7/2024

3. **Algoritmi greedy**

a) Si consideri il problema del partizionamento di intervalli.
   - **i.** Si fornisca un'istanza di **6 attivita`** del problema per la quale **la soluzione ottima e` uguale a 4** e l'algoritmo greedy ottimo incrementa il numero di risorse utilizzate negli istanti **1, 6, 8 e 10** e associa ciascuna delle risorse 1 e 4 a due attivita` e ciascuna delle altre risorse ad un'unica attivita` .
   - **ii.** Per ciascuna risorsa dire a quali attivita` essa viene assegnata.

Si indichino con precisione i valori numerici in input e si motivino le risposte ai punti I e II.

b) Si dimostri che l'algoritmo greedy per Interval Scheduling produce la soluzione ottima usando il seguente fatto: *per ogni p, il p-esimo job selezionato dall'algoritmo greedy termina non piu` tardi del job con il p-esimo tempo di fine piu` piccolo tra quelli selezionati dall'algoritmo ottimo.* Non occorre dimostrare il suddetto fatto.

c) Si scriva lo pseudocodice dell'algoritmo greedy che computa lo scheduling ottimo per il problema della minimizzazione dei ritardi con una piccola modifica. **L'algoritmo restituisce un'array R tale che R[i] contiene il ritardo dell'i-esima attivita` nell'ordinamento.** Inoltre, si dica se, a partire dai valori di R, sia possibile calcolare il valore della soluzione ottima e se si` in che modo.

4. **Programmazione dinamica**
   a) Si consideri il problema di subset sum.
      I. Si dica in cosa consiste il problema: in cosa consiste l'input e qual e` l'obiettivo del problema.
      II. Si spieghi cosa rappresenta OPT(i,w) e cosa rappresentano i suoi parametri.
      III. Si fornisca la relazione di ricorrenza che esprime il valore della soluzione ottima giustificando in modo chiaro la risposta.

b) Si fornisca la tabella dei valori computati dall'algoritmo di programmazione dinamica che calcola il valore della soluzione ottima per il problema del minimum coin change problem quando l'istanza in input e` v_1=1, v_2=2, v_3=4, v_4=5, V=8. Alla fine, **si fornisca la soluzione ottima** e si **indichino con un cerchio le entrate sui cui indici vengono effettuate le chiamate ricorsive dall'algoritmo che stampa la soluzione ottima**.

c) Si scriva lo pseudocodice dell'algoritmo ricorsivo che **stampa la soluzione ottima** per il problema dello zaino. Si analizzi il tempo di esecuzione dell'algoritmo nel caso pessimo giustificando in modo chiaro la risposta.

5. **Divide et Impera**
    a. Si scriva lo pseudocodice di MergeSort (comprensivo dell'algoritmo di fusione) **e si spieghi poi in modo chiaro perche' l'algoritmo ordina correttamente l'array ricevuto in input.**

Progettazione di Algoritmi (9 CFU)
15/7/2024

b. Si fornisca la relazione di ricorrenza che esprime il limite superiore al tempo di esecuzione dell'algoritmo al punto a. **Si giustifichi in modo chiaro la risposta.**

c. Si **scriva lo pseudocodice di** un algoritmo che prende in input un array A di numeri a due a due distinti (non necessariamente ordinati) e restituisce un array B che contiene gli elementi di A ordinati in ordine crescente. Per ottenere l'array ordinato B, l'algoritmo non puo` effettuare direttamente confronti tra gli elementi di A ma puo` invocare iterativamente QuickSelect **con input opportuni**. Si analizzi il tempo di esecuzione dell'algoritmo fornito nel caso pessimo. N.B.: non importa se l'algoritmo fornito e` meno efficiente degli algoritmi di ordinamento studiati.

Foglio per la minuta

Foglio per la minuta

Foglio per la minuta

Foglio per la minuta

Foglio per la minuta

Progettazione di Algoritmi (9 CFU)
15/7/2024