Algoritmi I (A.A. 2007-2008) Docente: Irene Finocchi

Appello dell'11 Febbraio 2008

L'appello è riservato agli studenti immatricolati nell'A.A. 2005-2006 o in anni precedenti.

Gli studenti esonerati devono svolgere solo gli esercizi 3 e 4. Nel caso in cui abbiano già consegnato la seconda parte del compito nell'appello del 28 gennaio (non ottenendo la sufficienza), il primo esonero non è più valido e l'esame deve essere sostenuto per intero.

Esercizio 1. Analizzare l'albero della ricorsione dell'algoritmo `fibonacci2` e dare una stima del numero di linee di codice mandate in esecuzione per calcolare F_n. Assumere di sapere che, in un albero binario in cui ogni nodo interno ha esattamente due figli, il numero di nodi interni è sempre uguale al numero di foglie diminuito di uno.

Esercizio 2. Sia A un array di n elementi. Denotiamo con $A[i; j]$ il sottoarray di A dalla posizione i alla posizione j (incluse). Considerare il seguente algoritmo:

1. trova il minimo m di $A[1; n]$ e scambialo con $A[1]$;
2. trova il massimo M di $A[1; n]$ e scambialo con $A[n]$;
3. se $m = M$ esci;
4. trova il più grande valore $x \in A[2; n-1]$ che sia $\leq (m + M)/2$;
5. esegui `partition` su $A[2; n-1]$ usando x come perno;
6. ordina ricorsivamente gli elementi di $A[2; n-1]$ che sono $\leq x$;
7. ordina ricorsivamente gli elementi di $A[2; n-1]$ che sono $> x$.

Qual è il tempo di esecuzione nel caso peggiore? Come è strutturata una istanza di caso peggiore?

Esercizio 3. Sia T un albero AVL contenente n nodi. Siano a e b, con $a \leq b$, due chiavi contenute in T. Progettare un algoritmo (il più possibile efficiente) che, dati T, a e b, stampi tutte le chiavi di T di valore $\geq a$ e $\leq b$. Discutere correttezza e tempo di esecuzione dell'algoritmo proposto. Potete assumere per semplicità che T non contenga chiavi duplicate.
[*Suggerimento: un algoritmo efficiente dovrebbe richiedere tempo $O(K + \log n)$, dove K è il numero di chiavi stampate.*]

Esercizio 4. Sia G un grafo e si considerino i seguenti due algoritmi per visitare G:

```
// Algoritmo A1
marca tutti i vertici di G come inesplorati
for i=1 to n
     if (i e' inesplorato) then visita(i)
```

```
// Algoritmo A2
marca tutti i vertici di G come inesplorati
for i=n downto 1
     if (i e' inesplorato) then visita(i)
```

La procedura `visita` è quella descritta nel libro di testo con la seguente modifica: i vertici sono marcati come `aperti`/`chiusi` utilizzando lo stesso vettore di marcatura usato dagli algoritmi $\mathcal{A}_1$ e $\mathcal{A}_2$, senza introdurre un vettore di marcatura locale. Sia $\mathcal{A}_1$ che $\mathcal{A}_2$ generano una foresta di copertura per G, ovvero una foresta che contiene tutti i vertici di G. Dire se le seguenti affermazioni sono vere o false, *motivando la risposta* e discutendo separatamente il caso in cui G è *non orientato* e quello in cui è *orientato*.

1. Indipendentemente dalla strategia di visita utilizzata (DFS/BFS), le foreste generate da $\mathcal{A}_1$ e da $\mathcal{A}_2$ hanno lo stesso numero di alberi.

2. Supporre di utilizzare una visita DFS. Le foreste generate da $\mathcal{A}_1$ e da $\mathcal{A}_2$ hanno lo stesso numero di archi nell'albero (tree edges).

3. Sia $i < j$. Se il vertice j è raggiungibile da i in G, esiste un albero T della foresta tale che j è raggiungibile da i in T.

NOME	COGNOME

Soluzione Esercizio 1:

Appello dell'11 Febbraio 2008

Appello dell'11 Febbraio 2008

Nome	Cognome

Soluzione Esercizio 2:

APPELLO DELL'11 FEBBRAIO 2008

Appello dell'11 Febbraio 2008

Nome	Cognome

Soluzione Esercizio 3:

Appello dell'11 Febbraio 2008

Appello dell'11 Febbraio 2008

Nome	Cognome

Soluzione Esercizio 4:

APPELLO DELL'11 FEBBRAIO 2008