

Laurea in “Informatica”

Corso di “Algoritmi e Strutture di Dati”

14 Febbraio 2012

1. Tempo disponibile 180 minuti. È ammesso ritirarsi entro 90 minuti.
2. Sono ammessi al più 3 scritti consegnati per l'A.A. 2010/11 (Giugno 2011-Febbraio 2012)
3. Non è possibile consultare appunti, libri o persone, né uscire dall'aula.
4. Le soluzioni degli esercizi devono:
 - spiegare a parole l'algoritmo usato (anche con l'aiuto di eventuali disegni)
 - fornire e commentare lo pseudocodice (dettagliando a parole il significato delle variabili)
 - giustificare la correttezza e la complessità (con tutti i passaggi matematici necessari)
5. Un esercizio può ammettere più soluzioni: a soluzioni computazionalmente più efficienti e/o concettualmente più semplici sono assegnati punteggi maggiori

1. Si calcoli la complessità $T(n)$ della seguente procedura ricorsiva, e si dica se tale complessità è polinomiale o esponenziale nella *dimensione* dell'input.

```
integer mystery(integer [ ] A, integer n)
    integer k ← ⌊n/2⌋
    for integer i ← n downto n - 14 do
        | k ← 2
    if n < 33 then
        | return 3
    else
        | return k · mystery(A, ⌊n/2⌋) + mystery(A, ⌊n/k⌋) + ⌊n/k⌋ · mystery(A, k)
```

2. Si scriva lo pseudo-codice di un algoritmo *divide-et-impera* che, preso un vettore A di n interi, ne calcoli in tempo *ottimo* la sommatoria dividendo l'input, ad ogni chiamata ricorsiva, in *tre parti bilanciate*. Si impostino e si risolvano le relazioni di ricorrenza della complessità.

3. Si scriva lo pseudo-codice di un algoritmo che, dati un albero binario T di n nodi ed un intero k , aggiunge in tempo *ottimo* un figlio sinistro ad ogni foglia di livello k , assumendo T realizzato con puntatori.

4. Siano dati un grafo *non orientato* $G = (N, A)$, dove ogni arco è colorato di *rosso* o di *blu*, e due nodi r ed s di N . Si vuole verificare se esiste un cammino da r ad s formato soltanto da archi rossi. Scrivere lo pseudo-codice di un algoritmo che richieda tempo *ottimo*.

5. Si scriva lo pseudo-codice dell'algoritmo di Dijkstra visto a lezione per calcolare i cammini minimi in un grafo orientato pesato da un nodo r a tutti i rimanenti nodi, e se ne discuta la complessità. Si esegua poi (a mano) l'algoritmo sul grafo $G = (N, A)$, con $N = \{1, 2, 3, 4\}$, $A = \{(1,2), (1,3), (2,3), (2,4), (2,5)\}$, $r = 1$ e pesi $w(1,2)=1$, $w(1,3)=2$, $w(2,3)=1$, $w(2,4)=3$, $w(3,4)=2$, mostrando ad ogni passo il contenuto delle strutture di dati usate.

6. Dato un insieme A di n interi distinti, si vuole decidere se esiste una *partizione* in due sottoinsiemi S e T tali che S abbia cardinalità uguale a 2 ed il prodotto degli elementi in S sia uguale al doppio della somma degli elementi in T . Scrivere lo pseudo-codice di un algoritmo *non deterministico* che richieda tempo *polinomiale*.