Informatica 3

Prof. Giovanni Agosta
Prof. Alessandro Campi
Prof. Maristella Matera

Prova scritta – Appello del 04/02/2011[1]

COGNOME e NOME:..... Matricola:.....
SEZIONE: Agosta ○ Campi ○ Matera ○

[1]Tempo: 2 ore. Possono essere consultati libri di testo e slide del corso. È consentito scrivere a matita. Scrivere il proprio nome sugli eventuali fogli aggiuntivi. *Rispondere punto per punto alle domande!*

Esercizio 1 (8 punti)

Scrivere cosa stampa a video il seguente programma Python quando come parametro da linea di comando viene passata la propria matricola, giustificando la risposta.

```
#!/usr/bin/python
from sys import argv

matr = argv[1]
x = [ int(i) for i in matr ]
z = [ i**2 for i in x if i<5 ]

print len(z), reduce(lambda x,y: x+y, z[1:-1])

def foo(h):
def bar(g):
return str(g)+str(h)
return bar

print foo(min(z))(max(z))

print [ i-j for i in z for j in z if i==j ]

print 0 if 0 in z else 1

try :
if 0 in z or 7 not in z :
raise Exception, 'This is exception 0'
if 1 in z or 5 not in z :
raise Exception, 'This is exception 1'
print 'No exception!'
except Exception, e:
print e

l=zip(z, range(len(z)))
l.sort()
print l[0][0], l[-1][1]

print len([ i for i,j in l if i==j ])
```

Esercizio 2 (8 punti)

Dato un albero binario, definiamo altezza minimale di un nodo v la minima distanza di v da una delle foglie del suo sottoalbero, definiamo invece altezza massimale di un nodo v la massima distanza di v da una delle foglie del suo sottoalbero. Supponiamo di avere un albero T che contiene in ogni nodo anche un numero intero m. Diciamo che l'albero è k-equilibrato se tutti i nodi (eccetto al più k) contengono un valore m che è compreso tra l'altezza minimale e l'altezza massimale del nodo stesso. Definire un algoritmo che riceve in input l'albero T e un intero k e restituisce vero se l'albero T è k-equilibrato, falso altrimenti.

Esercizio 3 (5 punti)

Determinare la codifica di Huffman della seguente frase: *it would be nice to know an optimum way of encoding*

Esercizio 4 (5 punti)

Inserire le seguenti stringhe in un 2-3-4 Tree (ovvero un BTree con 1-3 chiavi e 2-4 figli per nodo) inizialmente vuoto: *cod, bee, ape, cow, rat, fox, bat, cat, dog, gnu, emu, boa, ant.*

Esercizio 5 (7 punti)

1. Determinare i bound di complessità asintotica più stretti per le seguenti ricorrenze:

   (a) $T(n) = T(n-1) + T(n-3) + \Theta(1)$

   (b) $T(n) = 4T(n/3) + n^5$

   (c) $T(n) = T(n-5) + n^2 + 5n$

   (d) $T(n) = 2T(n/4) + n^2 log n$

2. Determinare la complessità computazionale del seguente frammento di codice C in funzione della lunghezza della stringa A.

```
int foo(char A[], int n, int m){
   int i, a=0;
 if (n>=m) return 0;
   for(i=n; i<m; i++)
     a+=A[i];
   return a + foo(A, n*2, m/2);
}

int bar(char A[]){
   return foo(A,1,strlen(A));
}
```