Due Date: 25.10.2004, until 13:00.
Submit to: Ömer Sinan Sarac (Bz-19)

Specifications:
• Answers to the questions should be your individual work. For similarities above a threshold, I won't accept “we elaborated/thought on the problem together” as an excuse. In that case, both parts will receive 0.
• You have to submit a computer print-out. Hand written homeworks will not be accepted.

Give answers to the following exercises. Try to be clear and short.

1) (15 Points) Use master method to find an asymptotical tight bound for the following recurrence relation. You can assume $T(n) = \Theta(1)$ for $n = 1$.

$T(n) = 2T(n/4) + n \log n$

2) (15 Points) Find an asymptotical upper bound for the given recurrence relation using both substitution and iteration methods. You can assume $T(n) = \Theta(1)$ for $n = 1$.

$T(n) = 2T(n-1) + n$

3) (30 Points) Analyse the following function $cld$ given in pseudocode:

```python
CLD(m, n)
1 If n > m then SWAP(n, m)
2 If m mod n = 0
3 then return n
4 else return CLD(n, m mod n)
```

a) Prove that the algorithm halts for all inputs.

b) Explain what the algorithm does. (One sentence)

c) Prove the function $cld(m,n)$ runs in $O(m)$ time

d) The bound in part (c) is not tight. Can you find a tighter upper bound? [Hint: Consider the call $cld(m,n)$. In the worst case it will make a recursive call $cld(n,k)$. Find an upper bound of $k$ in terms of $m$. Also note that $k = m \mod n \Rightarrow m = n.p + k$ for some $p \geq 0$.]

Prove your bound by solving the recurrence relation.

e) Define a sequence $a_0, a_1, a_2, ..., a_n$ such that $cld(a_n, a_{n+1})$ will make $n$ recursive calls. Use this sequence to verify that your bound in (d) also bounds this case asymptotically. [Hint: If you can’t find such a sequence, read chapters 2-3 of Cormen’s book very carefully. The answer is within those pages]
4) (20 Points) Two sets with $O(n)$ elements are given as linked lists. The problem is finding the intersection of two sets. You may assume the sets have no duplicate elements. Also, assume comparing two elements takes $O(1)$ time.
   a) Give a pseudo code that runs in $O(n^2)$ time
   b) You can do better. Give a pseudocode that runs in $O(n\log n)$ time. You don’t need to give details and you can call other functions as long as you know their run time efficiency.

5) (20 Points) Given the function definitions below

   ```
   FUNCA(n)
   1   If n = 0
   2      then return 0
   3   else return FUNCB(n-1,n-1) - (n-1)
   
   FUNCB(m, k)
   4   If m = 0
   5      then return FUNCA(k)
   6   else return 1 + FUNCB(m-1, k)
   ```

   a) find the output produced by the statement
      ```
      print FUNCA(n), where n ≥ 0;
      ```
   b) find the recurrence relation for its running-time function;
   c) find its run-time complexity.