Practice questions

Problems provided as practice questions for the midterm.

1. Sort the following terms from *slowest* growing to *fastest* growing. If two terms are equivalent asymptotically, draw a circle around them in your ordering.

$$(\log n)^2 + n \quad 3^{2n} \quad n^{1/4} \quad n^{\log_3 7} \quad 2^{3n} \quad 1000(\log n)^2 \quad 2^{\log_2 n} \quad n \log_5 n \quad 5^{\log_3 n}$$

2. (2pts each) In each of the following cases, indicate if $f = O(g)$, $f = \Omega(g)$, $f = \Theta(g)$.

	$f(n)$	$g(n)$	O	Ω	θ
(a.)	$n^2 \log n$	$1500n^2 + 10n$			
(b.)	$1.2 + 1.2^2 + \cdots + 1.2^n$	1.2^{n+2}			
(c.)	$\log_2 n$	$\log_8 n$			
(d.)	3^n	n^{10}			
(e.)	$n^{\log_4 5}$	$n^{\log_2 5}$			

3. Maximum independent set: for a sequence of n numbers $x_1, ..., x_n$, find the set that has the largest sum and does not include consecutive pairs in $O(n)$ time. For example, the answer to the input $6, 4, 8, 2, 3, 5$ is 6+8+5. Try to solve this problem without any hint. But if you are indeed stuck, look at the hint[1].

4. You are given a strong of n characters $s[1, ..., n]$, which is a corrupted text in which punctuation has vanished (e.g., "itwasthebestoftime..."). You need to reconstruct the document using a dictionary, which is available in the form of a boolean function $dict(\cdot)$: for any string w, $dict(w)$ returns $true$ if w is a valid word, otherwise it returns $false$. Give a dynamic programming algorithm that determines whether the string $s[\cdot]$ can be reconstructed as a sequence of valid words. The run time should be at most $O(n^2)$, assuming a $dict$ call takes unit time. Try to solve this problem without any hint. If you are truly stuck, consider the hint[2].

5. Given two strings $x = x_1x_2 \cdots x_n$ and $y = y_1y_2 \cdots y_m$, we wish to find the length of their longest common substring, that is, the largest k for which there are indices i and j with $x_ix_{i+1} \cdots x_{i+k-1} = y_jy_{j+1} \cdots y_{j+k-1}$. Show how to do this in time $O(mn)$. Try to solve this problem without any hint. If you are truly stuck, consider the hint[3]

6. We use Huffman's algorithm to obtain an encoding of alphabet $\{a, b, c\}$, with frequencies f_a, f_b, f_c. In each of the following cases, either give an example of frequencies $\{f_a, f_b, f_c\}$ that would yield the specified code, or explain why the code cannot possibly be obtained (no matter what the frequencies are).

   a Code: $\{0, 10, 11\}$

[1] Let $L(i)$ be the maximum sum achievable considering sequence $x_1, ..., x_i$

[2] Let $L(i)$ be the answer to the question, can $s[1, .., i]$ be reconstructed as a sequence of valid words? Now if we know $L(1), L(2), ..., L(i-1)$, how can we use them to figure out $L(i)$?

[3] Let $L(i, j)$ be the longest common substring between $x_1x_2 \cdots x_i$ and $y_1y_2 \cdots y_j$ terminating at i and j.

b Code: $\{0, 1, 00\}$

c Code: $\{10, 01, 00\}$

7. Under a Huffman encoding of n symbols with frequencies $f_1, f_2, \ldots, f_n$, what is the longest codeword could possibly be? Give an example set of frequencies that would produce this case.

8. A server has n customers waiting to be served. The service time required by each customer is t_i minutes for customer i. So if, for example, the customers are served in the order of increasing t_i, then the i-th customer has to wait for $\sum_{j=1}^{i} t_j$ minutes. We wish to minimize the total wait time

$$T = \sum_{i=1}^{n} (\text{time spent waiting by customer } i)$$

. Prove the greedy algorithm that serves the customer in increasing order of t_i gives the optimal solution.