Suggested Solutions to Midterm Problems

1. Prove by induction that the regions formed by a planar graph all of whose vertices have even degrees can be colored with two colors such that no two adjacent regions have the same color.

   Solution. The proof is by *strong* induction on the number m of edges in the graph.

   Base case: When $m = 0$ (i.e., the graph has one or more isolated vertices), there is only one region which can be colored by any of the two colors.

   Induction step: Consider a planar graph G with $m = k$ ($k \geq 1$) edges. The induction hypothesis says that "a planar graph with $< k$ edges can be *properly* colored with two colored (such that no two adjacent regions have the same color)".

   G must contain a simple cycle (a cycle that passes through a node at most once). Remove the cycle from G to obtain a graph G' that has $< k$ edges and all of whose edges have even degrees. By the induction hypothesis, G' can be properly colored with two colors. The removed cycle, when put back, divides G' into two areas: one inside the cycle and the other outside the cycle. The cycle also divides some of the regions of G' into smaller regions (it is possible that a region be divided into more than two smaller regions). Flip the colors of the regions inside the cycle and we get a proper coloring for graph G (why this is so follows from an argument similar to that for the example of regions divided by lines in general position that we discussed in class).
   □

2. Prove by induction that a ring of even size can be colored with two colors and a ring of odd size with three colors such that no two adjacent nodes have the same color.

   Solution. The proof is by *strong* induction on the number n of nodes.

   Base case: A ring of two nodes can be colored with two colors and a ring of three nodes with three colors.

   Induction step: Consider a ring of n nodes. There are two cases: one when n is odd and the other when n is even.

   If n is odd, then we remove an arbitrary node from the ring to obtain a ring of $n - 1$ (which is even) nodes. By the induction hypothesis, the smaller ring can be properly colored with two colors. The removed node can be colored with the third color.

   If n is even, then we remove two arbitrary but consecutive nodes to obtain a ring of $n - 2$ (which is even) nodes. The smaller ring, again by the induction hypothesis, can be properly colored with two colors. The removed two nodes can be colored differently with the two colors and inserted back into the ring in such a way that no two adjacent nodes have the same color.

   □

3. Construct a gray code of length $\lceil \log_2 14 \rceil$ $(= 4)$ for 14 objects. Show how the gray code is constructed from gray codes of smaller lengths.

Solution. Let $(c_1, c_2, \ldots, c_n)^R$ denote the list $c_n, c_{n-1}, \ldots, c_1$.

Code of length 1 for 2 objects: $0, 1$.

Code of length 2 for 2 objects: $00, 01$.

Code of length 2 for 3 objects: $00, 01, 11$ (which is open).

Code #1 of length 3 for 3 objects: $000, 001, 011$.

Code #2 of length 3 for 3 objects: $100, 101, 111$.

Code of length 3 for 6 objects: $000, 001, 011, (100, 101, 111)^R$.

Code of length 3 for 7 objects: $000, 001, 011, 111, 101, 100, 110$ (which is open).

Code #1 of length 4 for 7 objects: $0000, 0001, 0011, 0111, 0101, 0100, 0110$.

Code #2 of length 4 for 7 objects: $1000, 1001, 1011, 1111, 1101, 1100, 1110$.

Code of length 4 for 14 objects:

$0000, 0001, 0011, 0111, 0101, 0100, 0110, (1000, 1001, 1011, 1111, 1101, 1100, 1110)^R$. □

4. Let $b(n)$ denote the number of distinct binary trees with n nodes; for example, $b(1) = 1$, $b(2) = 2$, and $b(3) = 5$. We stipulate that $b(0) = 1$. Write a recurrence relation that defines $b(n)$, for $n \geq 0$.

Solution. A binary tree of $n \geq 1$ nodes can have a left subtree with i nodes and a right tree with $n - i + 1$ nodes, for each feasible i. Therefore,

$$b(n) = \sum_{i=0}^{n-1} b(i)b(n-i+1)$$

□

5. Show all intermediate and the final AVL trees formed by inserting the numbers from 9 down to 0.

Solution. See the appended sheet. □

6. For each of the following pairs of functions, say whether $f(n) = O(g(n))$ and/or $f(n) = \Omega(g(n))$. Justify your answers.

	$f(n)$	$g(n)$
(a)	$\frac{n^2}{\log n}$	$n(\log n)^2$
(b)	$n2^n$	3^n

Solution. (a) $\frac{n^2}{\log n} = \Omega(n(\log n)^2)$, but $\frac{n^2}{\log n} \neq O(n(\log n)^2)$. It suffices to show that $n(\log n)^2 = o(\frac{n^2}{\log n})$, i.e., $\lim_{n\to\infty} \frac{n(\log n)^2}{\frac{n^2}{\log n}} = 0$.

$\lim_{n\to\infty} \frac{n(\log n)^2}{\frac{n^2}{\log n}} = \lim_{n\to\infty} \frac{(\log n)^3}{n} = \lim_{n\to\infty} \frac{((\log n)^3)'}{(n)'} = \lim_{n\to\infty} \frac{3(\log n)^2 \frac{1}{n}}{1} = \lim_{n\to\infty} \frac{3(\log n)^2}{n} = \cdots = \lim_{n\to\infty} \frac{6}{n} = 0$.

(b) $n2^n = O(3^n)$, but $n2^n \neq \Omega(3^n)$. It suffices to show that $n2^n = o(3^n)$, i.e., $\lim_{n\to\infty} \frac{n2^n}{3^n} = 0$. The proof is similar. □

7. (a) What is the result of merging the following two skylines: (1, **9**, 3, **12**, 9, **0**, 12, **6**, 18, **14**,22) and (3, **7**, 13, **4**, 16, **12**, 21, **8**, 25).
(b) Give a detailed algorithm (in pseudo code) for merging two skylines.

Solution. (a) (1, **9**, 3, **12**, 9, **7**, 13, **6**, 16, **12**, 18, **14**, 22, **8**, 25).
(b) Left as a programming exercise. □

8. Apply the quicksort algorithm to the following array. Show the result after each partition operation.

8	1	5	11	16	12	2	15	7	3	13	4	10	9	14	6

Solution.

8	1	5	11	16	12	2	15	7	3	13	4	10	9	14	6
7	1	5	6	4	3	2	**8**	15	12	13	16	10	9	14	11
2	1	5	6	4	3	**7**	**8**	15	12	13	16	10	9	14	11
1	**2**	5	6	4	3	**7**	**8**	15	12	13	16	10	9	14	11
1	**2**	4	3	**5**	6	**7**	**8**	15	12	13	16	10	9	14	11
1	**2**	3	**4**	**5**	6	**7**	**8**	15	12	13	16	10	9	14	11
1	**2**	3	**4**	**5**	6	**7**	**8**	14	12	13	11	10	9	**15**	16
1	**2**	3	**4**	**5**	6	**7**	**8**	9	12	13	11	10	**14**	**15**	16
1	**2**	3	**4**	**5**	6	**7**	**8**	**9**	12	13	11	10	**14**	**15**	16
1	**2**	3	**4**	**5**	6	**7**	**8**	**9**	11	10	**12**	13	**14**	**15**	16
1	**2**	3	**4**	**5**	6	**7**	**8**	**9**	10	**11**	**12**	13	**14**	**15**	16

□

9. Rearrange the following array into a heap using the bottom-up approach.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
8	2	5	11	9	12	3	10	7	1	13	4	15	14	6

Show the result after each element is added to the part of array that already satisfies the heap property.

Solution.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
8	2	5	11	9	12	3	10	7	1	13	4	15	14	6
8	2	5	11	9	12	**14**	10	7	1	13	4	15	**3**	6
8	2	5	11	9	**15**	14	10	7	1	13	4	**12**	3	6
8	2	5	11	**13**	15	14	10	7	1	**9**	4	12	3	6
8	2	5	11	13	15	14	10	7	1	9	4	12	3	6
8	2	**15**	11	13	**12**	14	10	7	1	9	4	**5**	3	6
8	**13**	15	11	**9**	12	14	10	7	1	**2**	4	5	3	6
15	13	**14**	11	9	12	**8**	10	7	1	2	4	5	3	6

□

10. Prove that the sum of the heights of all nodes in a complete binary tree with n nodes is at most $n-1$. (A complete binary tree with n nodes is one that can be compactly represented by an

array A of size n, where the root is stored in $A[1]$ and the left and the right children of $A[i]$, $1 \leq i \leq \lfloor \frac{n}{2} \rfloor$, are stored respectively in $A[2i]$ and $A[2i+1]$.)

Solution. Let $G(n)$ denote the sum of the heights of all nodes in a complete binary tree with n nodes. For a full binary tree (a special case of complete binary trees) with n nodes, where $n = 2^{k+1} - 1$ for some $k \geq 0$, we already know that $G(n) = 2^{k+1} - (k+2) = n - (k+1) \leq n - 1$.

A complete but not full binary tree can be regarded as the result of removing some of the right most leaves from a full binary tree. Note that removing leaf nodes from a full binary tree may reduce the heights of internal nodes that are ancestors of the removed leaves by at most 1. For a complete binary tree with $n = (2^{k+1} - 1) - m$ nodes, where $k \geq 1$ and $1 \leq m \leq 2^k - 1$, we find that

$$G(n) = (2^{k+1} - (k+2)) - \sum_{i=1}^{k} \lfloor \frac{m}{2^i} \rfloor$$

The term $\sum_{i=1}^{k} \lfloor \frac{m}{2^i} \rfloor$ is the number of nodes whose heights are reduced by 1 due to the removal to the m nodes. We need to show that $2^{k+1} - (k+2) - \sum_{i=1}^{k} \lfloor \frac{m}{2^i} \rfloor \leq (2^{k+1} - 1 - m) - 1$. The inequality simplifies to $m - k \leq \sum_{i=1}^{k} \lfloor \frac{m}{2^i} \rfloor$.

It suffices to consider the cases where $m + 1$ is a power of 2. This is so, since if the inequality $m - k \leq \sum_{i=1}^{k} \lfloor \frac{m}{2^i} \rfloor$ holds for m such that $m + 1 = 2^i$, where $1 \leq i \leq k$, then it holds any m such that $2^{i-1} \leq m \leq 2^i - 1$ (detail of proof omitted). For the cases under consideration, we observe that $\sum_{i=1}^{k} \lfloor \frac{m}{2^i} \rfloor$ is the number of internal nodes of a full binary tree with $m + 1$ leaves excluding those internal nodes that are on the left most branch of the tree. Since $m + 1 \leq 2^k$, the number is at least $((m+1) - 1) - k = m - k$. □

11. Write a program (or modify the following code) to recover the solution to a knapsack problem using the *belong* flag. You should make your solution as efficient as possible. (Note: The knapsack algorithm that appeared in the original problem statement has been removed.)

Solution.

Procedure Print_Solution (S, P, n, K);
begin
    if $\neg P[n, K].exist$ **then**
        print "no solution"
    else $i := n$;
        $k := K$;
        while $k > 0$ **do**
            if $P[i, k].belong$ **then**
                print i;
                $k := k - S[i]$;
            $i := i - 1$
end

□