Final

Note

This is a closed-book exam. There are ten problems in total, each accounting for 10 points.

Problems

1. Rearrange the following array into a (max) heap using the bottom-up approach.

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

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

2. Consider the following variation of the towers of Hanoi puzzle. There are four pegs A, B, C, and D with n disks of different sizes stacked in decreasing order on peg A. The objective is to transfer all the disks on peg A to peg B, moving one disk at a time (from one peg to one of the other three) and never having a larger disk stacked upon a smaller one. Design an algorithm to solve the puzzle. Your algorithm should make as few moves as possible.

3. Given two strings *aabca* and *acaba*, compute the minimal cost matrix $C[0..5, 0..5]$ for changing the first string character by character to the second one. Show the detail of your calculation for the entry $C[5, 5]$.

4. Prove that if the costs of all edges in a given connected graph are distinct, then the graph has an unique minimum-cost spanning tree.

5. (a) Modify Floyd's *All_Pairs_Shortest_Paths* algorithm so that it terminates immediately upon detecting the existence of a negative-weight cycle. (4 points)
   (b) Prove that your modification is correct. (6 points)

6. Let $G = (V, E)$ be a directed graph, and let T be a DFS tree of G. Prove that the intersection of the edges of T with the edges of any strongly connected component of G form a subtree of T.

7. Describe how back edges, forward edges, and cross edges are handled in the algorithm for computing strongly connected components of a directed graph.

8. Give a binary de Bruijn sequence of 2^4 bits. Explain how you can systematically produce the sequence.

9. The independent set problem is as follows.

   An independent set in an undirected graph is a set of vertices no two of which are adjacent. The problem is to determine, given a graph G and an integer k, whether G contains an independent set with $\geq k$ vertices.

   Prove that the independent set problem is NP-complete.

10. The traveling salesman problem is as follows.

    The input includes a set of cities, the distances between all pairs of cities, and a number D. The problem is to determine whether there exists a (traveling-salesman) tour of all the cities having total length $\leq D$.

    Prove that the traveling salesman problem is NP-complete.

Appendix

- Below is an algorithm for the towers of Hanoi puzzle.

```
Algorithm Towers_Hanoi(A,B,C,n);
begin
   if n=1 then
      pop x from A and push x to B
   else
      Towers_Hanoi(A,C,B,n-1);
      pop x from A and push x to B;
      Towers_Hanoi(C,B,A,n-1);
end;
```

- Floyd's algorithm for all-pairs shortest paths:

Algorithm All_Pairs_Shortest_Paths (*weight*);
begin
    for $m := 1$ to n **do**
        for $x := 1$ to n **do**
            for $y := 1$ to n **do**
                if $weight[x,m] + weight[m,y] < weight[x,y]$ **then**
                    $weight[x,y] := weight[x,m] + weight[m,y]$
end

- A DFS-based algorithm for strongly connected components:

Algorithm Strongly_Connected_Components (G, n);
begin
  for every vertex v of G **do**
    $v.DFS\_Number := 0$;
    $v.component := 0$;
  $Current\_Component := 0$;
  $DFS\_N := n$;
  while there is a vertex v such that $v.DFS\_Number = 0$ **do**
    $SCC(v)$
end

procedure SCC (v);
begin
  $v.DFS\_Number := DFS\_N$;
  $DFS\_N := DFS\_N - 1$;
  insert v into $Stack$;
  $v.high := v.DFS\_Number$;
  for all edges (v, w) **do**
    if $w.DFS\_Number = 0$ **then**
      $SCC(w)$;
      $v.high := \max(v.high, w.high)$
    else if $w.DFS\_Number > v.DFS\_Number$
        and $w.component = 0$ **then**
      $v.high := \max(v.high, w.DFS\_Number)$
  if $v.high = v.DFS\_Number$ **then**
    $Current\_Component := Current\_Component + 1$;
    repeat
      remove x from the top fo $Stack$;
      $x.component := Current\_Component$
    until $x = v$
end

- A binary de Bruijn sequence is a cyclic sequence of 2^n bits $a_1 a_2 \cdots a_{2^n}$ such that each binary code s of size n is represented somewhere in the sequence; that is, there exists a unique index i such that $s = a_i a_{i+1} \cdots a_{i+n-1}$ (where the indices are taken modulo 2^n).

- The clique problem: given an undirected graph $G = (V, E)$ and an integer k, does G contain a clique of size $\geq k$?
  The problem is NP-complete.

- The Hamiltonian cycle problem: given a graph G, does G contain a Hamiltonian cycle? (A Hamiltonian cycle in a graph is a cycle that contains each vertex exactly once.)
  The problem is NP-complete.