Multiple-choice Problems:

Problem 1. Which of the following is true of functions $f(n) = 100n + \log n$ and $g(n) = n + (\log n)^2$?

a) $f(n) = O(g(n))$ b) $f(n) = \Omega(g(n))$ c) $f(n) = \Theta(g(n))$

d) All of the above e) None of the above

Problem 2. Which of the following is true of functions $f(n) = n2^n$ and $g(n) = 3^n$?

a) $f(n) = O(g(n))$ b) $f(n) = \Omega(g(n))$ c) $f(n) = \Theta(g(n))$

d) All of the above e) None of the above

Problem 3. Which of the following statements are true of collision resolution with linear probing for universal hash function h with *constant* load factor $\alpha = \frac{n}{m}$, where *n* is the number of items to hash, and *m* is the size of the hash table?

a) Operations *Insert* and *Search* can take average $O(1)$ time.

b) The *Delete* operation may not take average $O(1)$ time.

c) For all keys x and y, where $x \neq y$, $Pr[h(x) = h(y)] \leq {}^1/_m$

d) All of (a),(b) and (c) e) None of (a), (b) and (c)

Problem 4. Using which of the following algorithm you can check whether a given graph is connected or not?

a) DFS b) BFS c) Dijkstra
d) (a) and (b) e) All of the above

Problem 5. For dense graphs with $\mathbf{\Omega}(n^2)$ weighted edges what is the best running time for an all-pairs shortest-paths algorithm that you have seen in the class?

a) $O(n^2 \log n)$ b) $O(n^3)$ c) $O(n)$ d) $O(n^3 \log n)$ e) $O(n^2)$

Problem 6. Let G be a Directed Acyclic Graph (DAG) with 22 vertices. What is the maximum possible number of vertices with both in-degree and out-degree one in G?

a) 1 b) 2 c) 20
d) 21 e) 22

Problem 7. Consider the graph below which is represented by its adjacency matrix *A*. Here *A[i,j]* is the weight of the edge from vertex *i* to vertex *j*. If we run Floyd-Warshall algorithm on this graph what would be the sum of the lengths of the shortest paths of the pairs (1,5), (4,5), (5,1), and (3,1)?

$$\begin{bmatrix} 0 & 5 & 9 & 4 & 1 & 6 \\ 5 & 0 & 5 & 1 & 8 & 6 \\ 9 & 5 & 0 & 7 & 5 & 3 \\ 4 & 1 & 7 & 0 & 9 & 9 \\ 1 & 8 & 5 & 9 & 0 & 4 \\ 6 & 6 & 3 & 9 & 4 & 0 \end{bmatrix}$$

a) 10 b) 11 c) 12 d) 13 e) 14

Problem 8. What is the length of the Minimum Spanning Tree (MST) in the graph of Problem 7?

a) 8 b) 9 c) 10 d) 11 e) 13

Regular Problems (you should always PROVE THE CORRECTNESS of your solutions):

Problem 9. Given a sequence of (not necessary positive) integers $x_1, x_2, \dots, x_n$ find a subsequence $x_i, x_{i+1}, \dots, x_j$ (of consecutive elements) such that the sum of the numbers in it is **even** *and* **maximum** over all subsequences of consecutive elements with an even sum of elements. For example if the sequence is $5, -1, 4, -10, 3, 3, 3$, the maximum even consecutive subsequence is $5, -1, 4$. The running time of your algorithm should be in $O(n)$.

We say a subsequence of consecutive elements is "even" if the sum of elements in it is even, otherwise we say it is "odd". Define E_i as the maximum sum of even subsequences ending at i. Similarly, define O_i as the maximum sum of odd subsequences ending at i. Clearly the answer to the problem is $\max_i E_i$. We only need to show how we can compute E_is and O_is in $O(n)$. One can easily show that the following recursive formula holds, which would directly give us the desired algorithm: E_1 and O_1 are easily computable. For $i \geq 2$:

- If x_i is even => $E_i = \max\{0, E_{i-1} + x_i\}$ and $O_i = O_{i-1} + x_i$
- If x_i is odd => $E_i = \max\{0, O_{i-1} + x_i\}$ and $O_i = E_{i-1} + x_i$

Problem 10. Given 1 US dollar, the set of currencies, and exchange rates between each pair of currencies, design an algorithm which decides whether we can change our 1 US dollar to more than 1 US dollar by just exchanging our money. Note that if we have two currencies u and v with exchange rate *r* from *u* to *v*, it means that *x* units of money in *u* are worth *r x* units of money in *v* for any $x \in \mathbb{R}$. (Hint: use the problem of finding a negative cycle in a graph).

We make a weighted directed graph G such that G has a negative cycle iff we can change our 1 US dollar to more than 1 US dollar. We put a vertex u in G for each currency u. For every exchange ratio r from currency u to currency v, we put a directed edge of weight $-\lg(r)$ from u to v and we put a directed edge of weight $-\lg(\frac{1}{r})$ from v to u.

Now consider a cycle in exchanging the currencies. If the cycle is $x_1, x_2, \dots, x_k, x_1$ and we have 1 unit of x_1, then we will have $r_1 \times r_2 \times \dots \times r_k$ units of x_1 by exchanging our money from x_1 to x_2, then from x_2 to x_3 and so on until we change our money back to x_1 from x_k (r_i is the exchange ratio from x_i to x_{i+1}).

On the other hand, the length of this cycle in graph G, is $-\lg r_1 - \lg r_2 - \dots - \lg r_k = -\lg(r_1 \times r_2 \times \dots \times r_k)$. Finally, we know that $r_1 \times r_2 \times \dots \times r_k > 1$ if and only if $-\lg(r_1 \times r_2 \times \dots \times r_k) < 0$ and thus we can change our money from 1 dollar to more than one dollar iff G has a negative cycle.

Problem 11. An **independent set** in an undirected graph G is a set S of vertices such that between any two vertices of S there is no edge in G. Given an undirected graph G and an integer k, the **independent set problem** asks whether there is an independent set of size at least k in graph G. Prove the clique problem is polynomially reducible to the independent set problem.

Given instance of CLIQUE $G = (V, E)$, construct graph $\bar{G} = (V, \bar{E})$. There is a clique of size $\geq k$ in G iff there is an I.S. of size $\geq k$ in $\bar{G}$.

Problem 12. Explain Dijkstra's algorithm in details and prove its correctness.

Look at Section 7.4 of the book.