1. To solve the single source shortest paths problem on weighted directed graphs with integral weights, one can first transform every directed edge (u, v) with weight w(u, v) = k > 1 into a directed path <u, x1, x3, . . . , xk−1 , v> with length k, and then use BFS to find the shortest paths. What are the advantages or disadvantages of this method? Discuss relationships with algorithms studied in the class. (Is this always possible to solve any instance of the problem using this method? Does it give a faster/slower algorithm? Why? Discuss various possible scenarios.)

   Ans: If you transform the graph using the transformation described in the problem (i.e., replacing each edge with weight w by w edges of unit weight), then the resultant graph will have size proportional to the sum of the weights of all the edges. Thus running BFS on the resultant graph will have a running time proportional to its new size. So, this transformation is useful (i.e., gives an efficient algorithm) if the sum of the weights of all the edges is smaller than the time needed to run the single source shortest paths algorithm on the original problem.

2.
   a. Simplify the expression: $O(n) + \Omega(n) + \Theta(n)$.
      Ans: $\Omega(n)$

   b. Sort the following terms in the increasing order of their asymptotic growth:

      $n^{(\log 6)}$, $6^{(\log n)}$, $(n \log n)^6$, $(6 \log 6)^n$, $(n \log 6)^6$, $(6 \log n)^6$

      Ans: $(6 \log n)^6$, $n^{(\log 6)}$, $6^{(\log n)}$, $(n \log 6)^6$, $(n \log n)^6$, $(6 \log 6)^n$

3. Given a set of n positive integers whose sum is S <= n^c for some small constant c>1, describe an efficient algorithm to check if there exists a partition of the set into two subsets such the sum of all the integers in each subset is the same. Analyze the running time of your algorithm.

   Ans: Use the pseudopolynomial time algorithm for the subsetsum problem discussed in class.

4. (a) Suppose someone finds a polynomial time algorithm for factoring. What would be the consequences for the relationship between P, NP and co-NP?

   Ans: No specific consequences with respect to these classes.

   (b) Independently (i.e., without having a polynomial time algorithms for factoring), what would be the consequences if someone showed a polynomial time reduction from subsetsum to factoring?

   Ans: NP = co-NP

5. What is a priority queue? What operations does it support? If we use a priority queue to sort an input sequence of n elements, what is the runtime (express it in terms of the complexities of the priority queue operations).

Ans: (first part from the slides). The input can be sorted by inserting the n elements into a min priority queue, and then performing n deletemin operations. The runtime is equal to n times the time for performing one insert and one deletemin operation.

6. Given the string "SHESELLSSEASHELLSONSEASHORE", encode it using a variable-length prefix code such that the total length of the encoding is minimized (assuming that the alphabet contains only the characters that appear in the string).

   Ans: Compute the frequencies of the characters and construct a Huffman tree to obtain variable length codes for the characters, and use these codes to encode the text.

7. Let A be an array consisting of n distinct integers. An inversion in A is a pair (i,j) such i<j and A[i]>A[j]. Describe the algorithm (that is discussed in the class) to compute the number of inversions in A. Show its runtime analysis briefly.

   Ans: (from Chapter 4)

8. There are n professors at NTNU, each having some joint research projects with m professors (from various universities) in Denmark – each professor can be part of multiple joint projects, and each joint project has exactly one Norwegian and one Danish professor associated with it. For a workshop discussing Norway-Denmark research cooperation, we would like to choose the minimum number of professors from either of these countries, such that all the joint projects are represented at the workshop (i.e., for each joint project, at least one of the professors associated with it is chosen).
   Is it possible to solve this problem in polynomial time? Justify your answer.
   [Note: You don't need to design an algorithm. Once you identify the right problem, you can lookup to check if there is an efficient algorithm to solve it.]

   Ans: If we add a vertex corresponding to each professor, and an edge between two vertices if the corresponding professors have a joint project, then we get a bipartite graph. Now choosing the minimum number of professors representing all the projects corresponds to finding a minimum vertex cover for this (bipartite) graph.
   We saw in Chapter 10 how to compute a minimum vertex cover in a bipartite graph in polynomial time.