1. Clearly indicate the following structures in the weighted graph pictured below. Some of these subproblems have more than one correct answer.

(a) A depth-first spanning tree rooted at $s$
(b) A breadth-first spanning tree rooted at $s$
(c) A shortest-path tree rooted at $s$
(d) A minimum spanning tree
(e) A minimum $(s,t)$-cut

2. A multistack consists of an infinite series of stacks $S_0, S_1, S_2, \ldots$, where the $i$th stack $S_i$ can hold up to $3^i$ elements. Whenever a user attempts to push an element onto any full stack $S_i$, we first pop all the elements off $S_i$ and push them onto stack $S_{i+1}$ to make room. (Thus, if $S_{i+1}$ is already full, we first recursively move all its members to $S_{i+2}$.) Moving a single element from one stack to the next takes $O(1)$ time.

(a) In the worst case, how long does it take to push one more element onto a multistack containing $n$ elements?
(b) Prove that the amortized cost of a push operation is $O(\log n)$, where $n$ is the maximum number of elements in the multistack.

3. Describe and analyze an algorithm to determine, given an undirected graph $G = (V,E)$ and three vertices $u, v, w \in V$ as input, whether $G$ contains a simple path from $u$ to $w$ that passes through $v$. You do not need to prove your algorithm is correct.
4. Suppose we are given an \( n \)-digit integer \( X \). Repeatedly remove one digit from either end of \( X \) (your choice) until no digits are left. The \textit{square-depth} of \( X \) is the maximum number of perfect squares that you can see during this process. For example, the number 32492 has square-depth 3, by the following sequence of removals:

\[
32492 \rightarrow \underline{32492} \rightarrow \underline{324} \rightarrow \underline{324} \rightarrow \underline{324} \rightarrow \underline{324} \rightarrow \underline{324} \rightarrow 4.
\]

Describe and analyze an algorithm to compute the square-depth of a given integer \( X \), represented as an array \( X[1..n] \) of \( n \) decimal digits. Assume you have access to a subroutine \texttt{IsSquare} that determines whether a given \( k \)-digit number (represented by an array of digits) is a perfect square \textit{in} \( O(k^2) \) time.

5. Suppose we are given two sorted arrays \( A[1..n] \) and \( B[1..n] \) containing \( 2n \) distinct numbers. Describe and analyze an algorithm that finds the \( n \)th smallest element in the union \( A \cup B \) in \( O(\log n) \) time.

6. Recall the following problem from Homework 2:

- \texttt{3WayPartition}: Given a set \( X \) of positive integers, determine whether there are three disjoint subsets \( A, B, C \subseteq X \) such that \( A \cup B \cup C = X \) and

\[
\sum_{a \in A} a = \sum_{b \in B} b = \sum_{c \in C} c.
\]

(a) Prove that \texttt{3WayPartition} is NP-hard. \textit{[Hint: Don't try to reduce from 3SAT or 3COLOR; in this rare instance, the 3 is just a coincidence.]}

(b) In Homework 2, you described an algorithm to solve \texttt{3WayPartition} in \( O(nS^2) \) time, where \( S \) is the sum of all elements of \( X \). Why doesn't this algorithm imply that \( P=NP \)?

7. Describe and analyze efficient algorithms to solve the following problems:

(a) Given an array of \( n \) integers, does it contain two elements \( a, b \) such that \( a + b = 0 \)?

(b) Given an array of \( n \) integers, does it contain three elements \( a, b, c \) such that \( a + b + c = 0 \)?