**Question 1.** *(10 points)* Indicate whether each of the following statements is TRUE or FALSE. No justification required.

1.1 (2 points): Given a flow network where all the edge capacities are even integers, the algorithm will require at most $C/2$ iterations, where $C$ is the total capacity leaving the source $s$.

1.2 (2 points): Suppose that $T(n) = 2T(n-1) + O(1)$ and $T(1) = 1$. Then $T(n) = O(n^2)$.

1.3 (2 points): You analyze the recursion tree for an algorithm with recurrence $T(n) \leq aT(n/b) + cn$ and find that amount of work is decreasing at each level of the recursion tree. The running time is $O(n)$.

1.4 (2 points): Suppose $f$ is a flow of value 100 from $s$ to $t$ in a flow network $G$. The capacity of the minimum $s$-$t$ cut in $G$ is equal to 100.

1.5 (2 points): The recurrence $\text{OPT}(j) = \max_{1 \leq i \leq j} \{A[i] - B[j - i] + \text{OPT}(j - i)\}$, where $A$ and $B$ are fixed input arrays, will lead to an $O(n)$ dynamic programming algorithm.
Question 2. (10 points) Each of the two graphs below displays a flow $f$ in a graph $G$. Each edge $e$ is labeled with two values $f(e)/c(e)$ (i.e., flow / capacity). You may use the right panel in each figure to draw the residual graph $G_f$ to help answer the questions.

1. What is the value of the flow?
2. What is the residual capacity of the edge $(e,d)$ in $G_f$?
3. Indicate a minimum cut and its capacity.
4. Is there an $s \rightarrow t$ path in the residual graph? If so, indicate one.
5. Is the flow a maximum flow?
Question 3. (10 points) You are working for the UMass fake news agency, and plan to write a story about the massive growth in crowd size at commencement over the history of UMass. In reality, crowd size fluctuates from year to year: there are $n$ years of records, and the crowd size in year $i$ is $s(i)$. Your plan is to find the two years $i$ and $j$ such that $i < j$ and the difference $s(j) - s(i)$ is as large as possible, and print photographs of the crowds in those two years to demonstrate the crowd growth.

3.1 (1 points): Suppose $n = 4$ and $s(1) = 3, s(2) = 1, s(3) = 4, s(4) = 5$ What years will you select in this example?

$i = \quad j =$

Consider the following outline for an algorithm to solve the problem. Assume that $n$ is a power of 2, and, for simplicity, assume your algorithm returns only the maximum value of $s(j) - s(i)$, and not the years themselves. Let $L = \{1, \ldots, n/2\}$ and $R = \{n/2 + 1, \ldots, n\}$.

1. Recursively find the biggest increase $s(j) - s(i)$ for $i, j \in L$, and call this value $\delta_L$

2. Recursively find the biggest increase $s(j) - s(i)$ for $i, j \in R$, and call this value $\delta_R$

3. ???

3.2 (2 points): Assuming Step 3 takes linear time, write a recurrence for the running time $T(n)$ of this algorithm.

3.3 (2 points): Give the running time of the algorithm (i.e., state the solution to your recurrence).

3.4 (5 points): Describe how to complete Step 3 of the algorithm using $O(n)$ time steps.
Question 4. (10 points) In this question, we develop an alternative shortest path algorithm for a specific type of graph. In a grid graph there are $n = k^2$ nodes labelled $v_{i,j}$ where $i$ and $j$ range between 1 and $k$. For each $i < k$ there is a “down” edge of length $d_{i,j}$ from $v_{i,j}$ to $v_{i+1,j}$. For each $j < k$ there is an “across” edge of length $a_{i,j}$ from $v_{i,j}$ to $v_{i,j+1}$. For example, a possible graph for $n = 9$ is

and in this graph, for example, the across edge from $v_{2,1}$ to $v_{2,2}$ has length $a_{2,1} = 3$.

4.1 (2 points): What is the length of the shortest path from $v_{1,1}$ to $v_{3,3}$ in the above example?

4.2 (4 points): Let $OPT(i, j)$ be the length of the shortest path from $v_{i,j}$ to $v_{k,k}$. Give a formula for $OPT(i, j)$ in terms of $OPT(i + 1, j)$ and $OPT(i, j + 1)$ when $i, j \leq k$. Your recurrence should have four different cases:

Case 1. $i < k, j < k$ \hspace{1em} $OPT(i, j) =$

Case 2. $i < k, j = k$ \hspace{1em} $OPT(i, j) =$

Case 3: $i = k, j < k$ \hspace{1em} $OPT(i, j) =$

Case 4: $i = k, j = k$ \hspace{1em} $OPT(i, j) =$

4.3 (4 points): Describe an algorithm to compute $OPT(1,1)$ and state its running time. You do not need to prove correctness.
Question 5. (10 points) Recall that an independent set in a graph is a subset of nodes $S$ such that no two nodes in $S$ are connected by an edge. In this problem, we seek a maximum-weight independent set in a tree $T = (V, E)$ with root node $r$ and node weights $w(v)$ for all $v \in V$.

For any node $v$, let $T_v$ denote the subtree rooted at $v$. Define $\text{OPT}(v)$ to be the maximum-weight independent set in $T_v$. For the base case, it is clear that $\text{OPT}(u) = w(u)$ if $u$ is a leaf.

5.1 (3 points): Now, let $C(v)$ be the children of $v$. Complete the recurrence below for a non-leaf node $u$:

$$
\text{OPT}(u) = \max \begin{cases}
    w(u) + \sum_{v \in C(u)} \sum_{x \in C(v)} \text{OPT}(x) & \text{(Case 1),} \\
    & \text{(Case 2).}
\end{cases}
$$

5.2 (2 points): Two different alternatives are being considered in Case 1 and Case 2. Explain precisely what they are.

5.3 (3 points): Consider a dynamic programming algorithm to compute $\text{OPT}(r)$ for the root node $r$. Indicate what size array is needed, and describe a valid order to fill in the array entries. (You do not need to give pseudocode).

5.4 (2 points): Let $n$ be the number of nodes and let $d$ be the maximum degree of any node. Give a running time bound for the algorithm in terms of $n$ and $d$. 
5.5 (2 points): (Extra credit) Give a tight running time bound that does not depend on $d$ and explain why it is correct.