Midterm 2

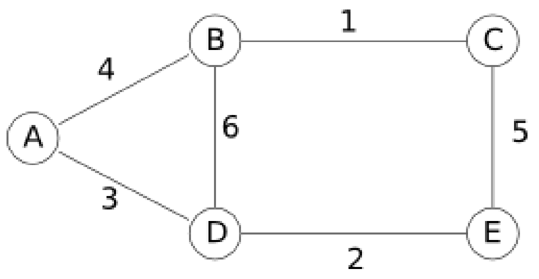
Name / SID:

TA / Section:

Answer all questions. Read them carefully first. Be precise and concise. The number of points indicate approximately the amount of time (in minutes) each problem is worth spending. Write in the space provided, and use the back of the page for scratch. **Good luck!**

1	
2	
3	
4	
5	
Total	

Problem 1 (*16 points total - 4 points each*)



In the graph shown above:

(a) In what order are the vertices deleted from the priority queue in Dijkstra's algorithm for shortest paths? (Starting at node A)

(b) In Prim's algorithm for the minimum spanning tree? (Starting at node A)

(c) In what order are the edges added in Kruskal's algorithm?

(d) Show the union-find trees at the end of Kruskal's algorithm (in case of a tie in rank, make the alphabetically first node the root).

Problem 2 (*16 points total - 2 points each*)

True or false? Provide a **very brief** explanation (1 to 3 sentences).

1. In Huffman's algorithm the character with the highest probability (all probabilities are unique) is guaranteed to be one of the leaves that is closest to the root (i.e it has the least depth among all leaves).

2. In the greedy algorithm for Horn clauses, we start with all variables set to true.

3. Even without path compression (but still using union-by-rank), any operation in the union-find data structure takes at most $O(\log n)$ time.

4. In the standard implementation of the union find data structure (with union-by-rank and path compression), any operation in the union-find data structure takes at most $O(\log^* n)$ time.

5. The simplex algorithm for linear programming is a polynomial-time algorithm.

6. At the conclusion of the max-flow algorithm, at least one edge coming out of the source S is full to capacity.

7. Consider a minimum capacity s-t cut, which partitions the vertices of a graph into two sets S and T, with $s \in S$ and $t \in T$. Then any maximum s-t flow saturates/fills to capacity all edges crossing from S to T.

8. Consider a minimum capacity s-t cut, which partitions the vertices of a graph into two sets S and T, with $s \in S$ and $t \in T$. Then increasing the capacity of any edge crossing from S to T increases the maximum s-t flow.

Problem 3 (*14 points total*)

(a) (*7 points*) We have a factory that produces two types of xylophones (a musical instrument), named x_1 and x_2. When we sell them, we earn a net profit of \$3 for each unit of x_1 and \$2 for each unit of x_2. We produce these items using two machines, A and B. To produce one unit of x_1, we require 5 minutes on machine A and 3 minutes on machine B. To produce one unit of x_2, we require 6 minutes on machine B. Due to union labor rules, the machine operators can only run machine A for 10 minutes per day and machine B for 12 minutes per day. Since xylophones are awesome, assume that we can sell out all units of x_1 and x_2 produced, and that we can produce and sell fractional amounts of x_1 and x_2. Write a linear program that finds the maximum profit that can be obtained per day. Find the optimal number of units of x_1 and x_2 to produce per day and the maximum profit it obtains per day.

(b) (*7 points*) Write the dual of this linear program. What is the value of its optimal solution?

Problem 4 (*15 points total)*

Given an array of integers $a[1], a[2], \ldots, a[n]$ (positive and negative) and an integer k (e.g. array $a[1, \ldots, 6] = 20, 100, -5, 80, 10, 15$ and $k = 3$), we wish to find a subset of exactly k of these numbers, which has the largest possible total weight, but such that no two adjacent numbers are taken.

(a) *(4 point)* Describe a simple greedy algorithm that finds a correct solution on the above example. Prove that it always works or show a counter example where it fails.

(b) We can solve this problem by (you guessed it!) dynamic programming. Fill in the blanks!

- *(3 point)* For each $i = 0, 1, \ldots, n$, and $j = 1, \ldots, k$ define a subproblem (describe in words) `maxsum`$[i, j]$ to be:

- *(1 point)* Initialize the base case(s) of the subproblem with the following code (it may help to solve the next part first):

- *(5 point)* Solve each subproblem iteratively with the following code:

- *(1 point)* In order to recover the the best subset, what other data structure would you use, and what data would it store?

- *(1 point)* What is the running time of the algorithm?

Problem 5 (*14 points total)*

(a) (*7 points)* Suppose that you have found a minimum spanning tree T of a weighted graph $G = (V, E)$; now you are told that the weight of an edge has been *decreased*. Describe a $O(|V| + |E|)$ algorithm to find a minimum spanning tree in this new graph, given T . (Hint: It should be not be done by recomputing a minimum spanning tree from scratch.)

(b) (*7 points)* Solve the above problem again for the case in which one edge weight is *increased.*