CMPSCI 311: Introduction to Algorithms
First Midterm Exam Solutions

February, 28, 2018.

Name: SOLUTIONS ID: _____

Instructions:

- Answer the questions directly on the exam pages.
- Show all your work for each question. Providing more detail including comments and explanations can help with assignment of partial credit.
- If you need extra space, use the back of a page.
- No books, notes, calculators or other electronic devices are allowed. Any cheating will result in a grade of 0.
- If you have questions during the exam, raise your hand.

Question	Value	Points Earned
1	10	
2	10	
3	10	
4	10	
5	10	
Total	50	

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

1.1 (*2 points*): *If G is a connected graph on n vertices, then it must have at least n edges.*
Solution: False. A tree is a connected graph, but a tree on n vertices has $n-1$ edges.

1.2 (*2 points*): $\sum_{i=1}^{n}\sum_{j=1}^{n} ij = \Theta(n^3)$.
Solution: False. $\sum_{i=1}^{n}\sum_{j=1}^{n} ij = \sum_{i=1}^{n} i \sum_{j=1}^{n} j = (\sum_{i=1}^{n} i)^2 = (n(n+1)/2)^2 = \Theta(n^4)$.

1.3 (*2 points*): *When assigning lectures to classrooms, recall that the* depth *is the maximum number of lectures that are in progress at the same time. There always exists an assignment of lectures to classrooms (such that no two lectures use the same classroom at the same time) that uses exactly depth classrooms.*
Solution: True. The greedy algorithm that orders by starting time finds a solution with at most depth classrooms.

1.4 (*2 points*): *Let $G_1 = (V, E, w)$ be a weighted graph where edge e has edge weight $w(e) > 0$ and let $G_2 = (V, E, w')$ be another graph with weights $w'(e) = w(e) + 1$ for all $e \in E$. If T is an MST in G_1, then T is also an MST in G_2.*
Solution: True. All spanning trees use $n-1$ edges. So if the weight of each edge is increased by 1, the cost for each spanning tree goes up by exactly $n-1$, the MST cannot change.

1.5 (*2 points*): *Suppose we have a stable matching instance where all colleges rank student* 1 *last in their ordering. There exists a stable matching where student* 1 *is matched with their first choice college.*
Solution: True. Suppose s_1 prefers c_1 to c_2 and s_2 prefers c_2 to c_1. Both c_1, c_2 prefer s_2 to s_1. Now if we run Gale-Shapley where the students propose, s_1 proposes to c_1 who accepts, and then s_2 proposes to c_2, who also accepts, and we are done.

Question 2. (*10 points*) Suppose we have an array A of numbers of size n and suppose that the array is already sorted in increasing order with no duplicates. Indexing into the array is an $O(1)$ operation. For each of the following problems, give a brief description of a *simple* algorithm and state the running time of your algorithm. No proof required.

2.1 (*2 points*): *Compute the maximum value in the array, i.e.,* $\max_{1 \le t \le n} A[t]$.

Running Time: **Solution:** $O(1)$

Brief Description: **Solution:** Just return $A[n]$ which is the largest since the list is sorted

2.2 (*2 points*): *Compute the mean or average value in the array, i.e.,* $\frac{1}{n}\sum_{t=1}^{n} A[t]$.

Running Time: **Solution:** $O(n)$

Brief Description: **Solution:** In one pass, add up all the elements, then divide by n.

2.3 (*3 points*): *For this question, assume that n is odd. Compute the median value in the array. The median of a set of n numbers S, where n is odd, is the element $x \in S$ for which exactly $(n-1)/2$ elements of S are strictly smaller than x.*

Running Time: **Solution:** $O(1)$

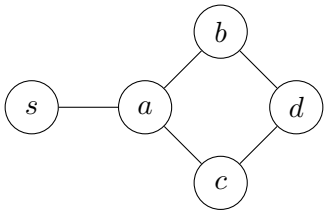
Brief Description: **Solution:** Return $A[(n+1)/2]$. Since the list is sorted, $(n-1)/2$ entries are strictly smaller

2.4 (*3 points*): *Compute the squared pairwise differences* $\sum_{i=1}^{n}\sum_{j=1}^{n}(A[i]-A[j])^2$. *(Hint: simplify this expression.)*

Running Time: **Solution:** $O(n)$

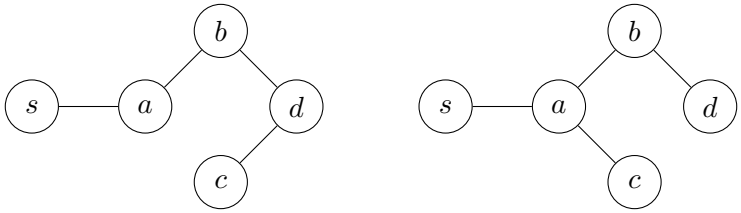
Brief Description: **Solution:** $\sum_{i=1}^{n}\sum_{j=1}^{n}(A[i]-A[j])^2 = 2\left(n\sum_{i=1}^{n}A[i]^2 - (\sum_{i=1}^{n}A[i])^2\right)$. Both of these two terms can be computed in one pass.

Question 3. (*10 points*) In the first two parts of this question we consider the following graph G:



3.1 (*4 points*): *Draw the trees we obtain by running DFS and BFS starting from s.*

Solution: *The left tree is DFS and the right tree is BFS. (Other solutions are also possible)*



3.2 (*2 points*): *Is the graph bipartite?*
Solution: Yes. You can see this since in the BFS tree the only non-tree edges is between consecutive layers. Or because there are no odd-length cycles.

The next two questions relate to an arbitrary undirected graph H.

3.3 (*2 points*): *Suppose H is a tree and we run DFS and BFS starting from the same node s. Are the resulting trees the same? Justify your answer.*
Solution: Yes. There cannot be any non-tree edges (since the original graph is a tree!). Both algorithms must use all the edges, so they produce the same tree.

3.4 (*2 points*): *Conversely, suppose in some general connected graph H we run DFS and BFS starting from the same node s and we find that the trees are the same. Does this necessarily mean that H is a tree? Justify your answer.*
Solution: Yes. If H is not a tree, it must contain a cycle. Then DFS will ensure that all of the nodes in the cycle are on the same path from the root, while BFS will split the cycle into at least two branches.
Another argument is that there must be one non-tree edge, but in DFS the non-tree edge must be between an ancestor and a descendant, but in BFS the non-tree edge must be between two nodes in consecutive layers. If the non-tree edge is the same in both, then the tree itself must be different, since the endpoints must be in different locations in the tree. On the other hand if the non-tree edge is different, then clearly the tree is different.

Question 4. (*10 points*) Consider the following *fractional packing* problem. There are n liquids where liquid i has weight w_i and value v_i. We would like to take fractional amounts of each liquid to maximize the value, without exceeding a weight budget W. Specifically, a solution specifies fractions $f_1, \ldots, f_n$ where $0 \leq f_i \leq 1$ and a solution is feasible if $\sum_{i=1}^{n} f_i w_i \leq W$. We would like to find a feasible solution that maximizes the value $\sum_{i=1}^{n} f_i v_i$.

4.1 (*2 points*): *Suppose we have two liquids with $w_1 = 1, w_2 = 2, v_1 = 5, v_2 = 8$ with budget $W = 2$. What are the optimal fractions f_1, f_2?*
Solution: $f_1 = 1, f_2 = 0.5$. This satisfies the weight budget since $w_1 f_1 + w_2 f_2 = 1 + 1 = 2$. And the value is $v_1 f_1 + v_2 f_2 = 5 + 4 = 9$.

A natural greedy algorithm involves sorting in some way, and then taking as much of each liquid as possible (in order) until you meet the weight budget W. Specifically if when processing liquid i, A is the weight of the current partial solution, we set $f_i \leftarrow \min\{1, A/w_i\}$ and we update $A \leftarrow A - f_i w_i$.

4.2 (*2 points*): *It turns out that ordering the liquids in decreasing order of value v_i, does not always produce an optimal solution. Clearly describe an example where this ordering fails.*
Solution: This greedy strategy fails on the example above. If we run the greedy algorithm on that example, we will set $f_2 = 1$ and use up our weight budget, but the value will be $v_2 f_2 = 8$ which is suboptimal.

4.3 (*2 points*): *It turns out that ordering the liquids in increasing order of weight w_i, does not always produce an optimal solution. Clearly describe an example where this ordering fails.*
Solution: Modify the example above, by setting $v_1 = 3$. Then ordering by increasing weight will produce $f_1 = 1, f_2 = 0.5$ but the value is $v_1 f_1 + v_2 f_2 = 3 + 4 = 7$. However in this modified example if we take $f_2 = 1$ we get value of 8.

4.4 (*4 points*): *Processing the liquids in decreasing order of v_i/w_i always produces an optimal solution. Prove that this is the case by using an exchange argument to show how to improve any other solution.*

Solution: *Assume that the items are sorted by v_i/w_i (so liquid 1 maximizes this ratio). Let $g_1, \ldots, g_n$ be another solution. If there exists $i < j$ such that $g_i < 1, g_j > 0$ then we can shift a fixed amount of weight from g_j to g_i. Specifically let $f_i = \max\{g_i + w_j g_j / w_i, 1\}$ and set $f_j = 0$ if $g_i + w_j g_j / w_i \leq 1$ and otherwise set it to $f_j = g_j - (1 - g_i) w_i / w_j$. Leave the remaining solution unchanged. This leaves the weight unchanged since (Here I am just considering the case where $f_j = 0$ but the other case is very similar)*

$$\sum_{k=1}^{n} w_k f_k = w_i(g_i + w_j g_j / w_i) + \sum_{k \neq i,j} w_k g_k = \sum_{k=1}^{n} w_k g_k.$$

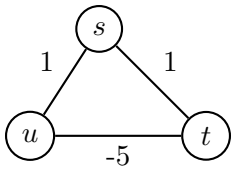
On the other hand, we have improved the value since

$$\sum_{k=1}^{n} v_k f_k = v_i(g_i + w_j g_j / w_i) + \sum_{k \neq i,j} v_k g_k = v_i w_j g_j / w_i + \sum_{k \neq j} v_k g_k \geq \sum_{k=1}^{n} v_k g_k.$$

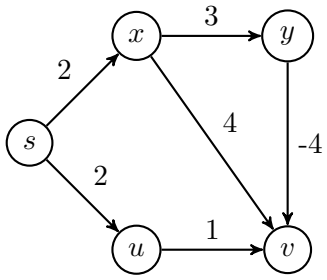
Question 5. (*10 points*)

5.1 (*2 points*): *Draw a weighted undirected graph with just one negative weight such that some shortests paths have length* $-\infty$.

Solution:



Consider the directed graph



5.2 (*4 points*): *We can also run Dijkstra's algorithm in a directed graph, considering only out-going edges. In this graph, if we run Dijkstra's algorithm what are the distances that we find?*

$d(x) =$ ***Solution*** 2 $\quad d(y) =$ ***Solution*** 5 $\quad d(u) =$ ***Solution*** 2 $\quad d(v) =$ ***Solution*** 3

5.3 (*4 points*): *In the proof of correctness for Dijkstra's algorithm, we argued that when we reach a node* v *for the first time, the path we used must be the shortest path. Specifically, if there were some other path* P *to* v*, it must leave the explored set on some edge* (x, y)*, so if* P' *is the subpath from* s *to* x*, we have*

$$\ell(P) \geq \ell(P') + w(x, y) \geq d(x) + w(x, y) \geq d(y) \geq d(v).$$

When the edge weights can be negative, one of these inequalities no longer holds. Which one is it and describe an example where this inequality does not hold.

Solution: The first inequality is incorrect, since a path P can have shorter weight than its subpath P'. An example of this is in the above graph, where the path (s, x, y, v) has length 1, but the subpath (s, x, y) has length 5.