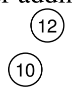
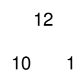
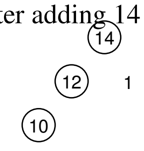
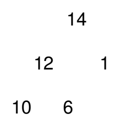
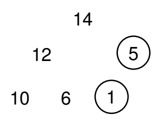
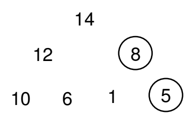
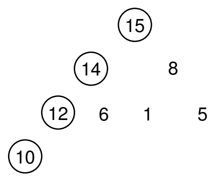
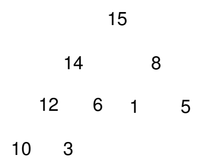
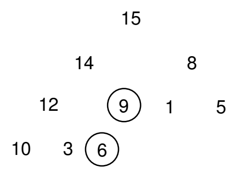
June 5, 2007 Name ______________________________

EECS 311 Data Structures
Final Exam
Don't Panic!

1. [10 pts] In the boxes below, show the result of adding the given numbers to a heap, using < as the predicate**.** Circle items that had to move at each entry.

1. After adding 10 to an empty heap.	2. After adding 12 to the previous heap. 
3. After adding 1 to the previous heap. 	4. After adding 14 to the previous heap. 
5. After adding 6 to the previous heap. 	6. After adding 5 to the previous heap. 
7. After adding 8 to the previous heap. 	8. After adding 15 to the previous heap. 
9. After adding 3 to the previous heap. 	10. After adding 9 to the previous heap. 

Average: 9.57. Median: 10.

June 5, 2007 Name ______________________________

2. [10 pts] Show how Quicksort with median-of-three would sort the array below. Be very clear about what happens in each partitioning phase, e.g., for each phase, write something like:

Partition __ to __, choose pivot __ from _______
Swap __ with __, __ with __, __ with __, ...
Resulting Array =

Be sure to include swaps done with the pivot at the beginning and end of a partitioning phase. Circle the items that are done and not involved in later phases. If a partition is 3 or fewer elements, just indicate the swaps needed, if any, to directly sort it.

0	1	2	3	4	5	6	7	8	9	10
3	**1**	**4**	**0**	**8**	**9**	**2**	**6**	**7**	**-1**	**5**

Partition 0 to 10, choose pivot 5 from 3, 9, 5
Swap 8 with -1, 9 with 2, 9 with 5
Resulting Array = 3 1 4 0 -1 2 $\boxed{5}$ 6 7 8 9

Partition 0 to 5, choose pivot 3 from 3, 4, 2
Swap 3 with 2, 4 with -1, 3 with 4
Resulting Array = 2 1 -1 0 $\boxed{3}$ 4 $\boxed{5}$ 6 7 8 9

Partition 0 to 3, choose pivot 1 from 2, 1, 0
Swap 1 with 0, 2 with -1, 2 with 1
Resulting Array = -1 0 $\boxed{1}$ 2 $\boxed{3}$ 4 $\boxed{5}$ 6 7 8 9

Partition 0 to 1 only 2 elements, no swaps needed

Partition 7 to 10, choose pivot 7 from 6, 7, 9
Swap 7 with 9, 9 with 7
Resulting Array = -1 0 $\boxed{1}$ 2 $\boxed{3}$ 4 $\boxed{5}$ 6 $\boxed{7}$ 8 9

Remaining partitions less than 3 elements, no swaps needed.

Comment [CKR1]: Median of 3 means use median value of first, middle and last locations as pivot

Comment [CKR2]: No need to move pivot first.

Comment [CKR3]: Restore pivot

Comment [CKR4]: 4, not 0, because division by 2 rounds down to get middle

Comment [CKR5]: Move pivot to end

Comment [CKR6]: Restore pivot

Comment [CKR7]: Move pivot

Comment [CKR8]: Restore pivot

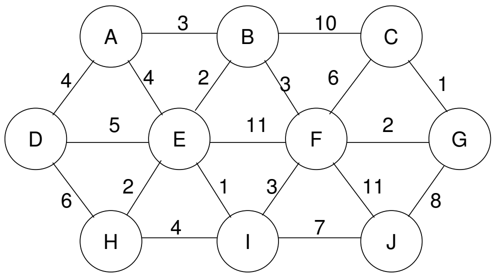
Comment [CKR9]: Even though the partition happens to be sorted, Quicksort doesn't know that, so this has to be done.

Comment [CKR10]: Move pivot to end

Comment [CKR11]: Restore pivot

Average: 8.14. Median: 9
Most common mistakes: not moving the pivot out and back, not sorting the righthand partition, swapping the pivot during the partitioning phase

3. [10 pts] Fill in the table below to show how Kruskal's algorithm would find a minimum spanning tree for the graph below. Draw a line on each edge of the graph used in the final MST. Is the MST unique? Justify your answer.



There are two possible MSTs. The other occurs if you see and accept F-I and reject B-F.

Edge	Weight	Action
E-I	1	Accept
C-G	1	Accept
F-G	2	Accept
B-E	2	Accept
E-H	2	Accept
B-F	3	Accept
A-B	3	Accept
F-I	3	Reject
A-E	4	Reject
A-D	4	Accept
H-I	4	Reject
D-E	5	Reject
C-F	6	Reject
D-H	6	Reject
I-J	7	Accept
Done		

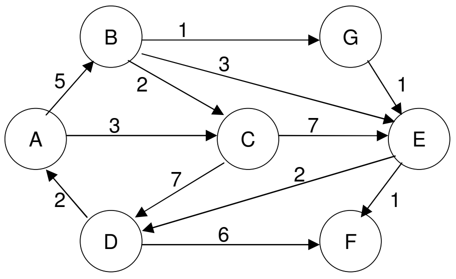
Comment [CKR12]: Because it forms an edge. This is the ONLY place where order in which the edges are checked mattered.

Comment [CKR13]: Stop as soon as 9 edges are accepted.

Average: 9.07. Median: 9

June 5, 2007 Name ______________________________

4. [10 pts] Use the table below to show how Dijkstra's algorithm would find the shortest path from A to D. In the **Distance** column put the various distance values assigned to each vertex. In the **When Finalized** column put 1 for the first vertex that is finished, 2 for the 2nd vertex finished, etc.



When Finalized	**Vertex**	**Distance**
1	A	0
3	B	5
2	C	3
7	D	~~10~~ 9
5	E	~~10~~ ~~8~~ 7
6	F	8
4	G	6

Average: 9.11. Median: 10
Most common mistakes: wrong order of finalization, not adding distances.

5. [15 pts total] a). Given an array of N elements of three different types: Cold, Warm, and Hot, design and describe clearly an O(N) in-place algorithm to put all the cold elements, on the left, followed by all the warm elements, followed by all the hot elements on the right. Your algorithm can use only a small constant amount of extra space.

```
-- Put all cold items on the left
Let i = 0.
For J from 0 to N-1
  if a[j] is cold, swap a[i] and a[j] and increment i.
-- Put all hot items on the right
Let k = N-1.
For j from k down to I
  if a[j] is hot, swap a[k] and a[j] and decrement k.
```

Comment [CKR14]: A single pass can work but you need to maintain 3 indices, and the conditional is much more complex because of the possible cases.

b) Show how your algorithm would operate on this array:

0	1	2	3	4	5	6	7	8	9	10
C	**W**	**H**	**C**	**W**	**W**	**C**	**H**	**C**	**W**	**C**

I = 0. j = 0. a[j] is C. Swap a[i] and a[j], I becomes 1.
When j is 3, swap, I becomes 2, a = C C H W W W C H C W C
When j Is 6, swap, I becomes 3, a = C C C W W W H H C W C
When j Is 8, swap, I becomes 4, a = C C C C W W H H W W C
When j Is 10, swap, I becomes 5, a = C C C C C W W H H W W
k = 10, j Is 10.
When j is 8, swap, k becomes 9, a = C C C C C W W W H W H
When j is 7, swap, k becomes 8, a = C C C C C W W W W H H

c) Give as formal an argument as you can for why your algorithm is O(N) and correct for all possible arrays, even those with only a single kind of element.

Complexity = O(N) because there are two sequential FOR loops with at most N iterations of O(1) operations.

Correctness by invariants: In first loop, invariant is "$a[n] = C$ for all $n < I$, $a[n] \neq C$ for all $I \leq n < j$." Trivially true when I and j are 0. Each iteration maintains this invariant, ergo, when $j = n$, all C's and only C's must be left of i. The same argument applies to the second loop, with the invariant "$a[n] = H$ for all $n > k$, $a[n] \neq H$ for all $j < n \leq k$."

Average: 11.46. Median 12
Most common mistakes: no correctness argument, very unclear algorithm description, sorting which is N log N, copying into another array or list, which is not O(1) extra space, too many vague terms in proof, assuming the elements are identical to C, W and H.

June 5, 2007 Name ______________________________

6. [10 pts] Use dynamic programming to find the optimal solution to the sequence alignment problem below. A match scores 2 points, a mismatch scores -1, and a gap in either sequence scores -2.

Sequence 1: G A A T T C A G T T A
Sequence 2: G G A T C G A

		G	A	A	T	T	C	A	G	T	T	A
	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22
G	-2	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18
G	-4	0	1	-1	-3	-5	-7	-9	-8	-10	-12	-14
A	-6	-2	2	3	1	-1	-3	-5	-7	-9	-11	-10
T	-8	-4	0	1	5	3	1	-1	-3	-5	-7	-9
C	-10	-6	-2	-1	3	4	5	3	1	-1	-3	-5
G	-12	-8	-4	-3	1	2	3	4	5	3	1	-1
A	-14	-10	-6	-2	-1	0	1	5	3	4	2	3

Comment [CKR15]: Putting all zeros in the first row and column means there's no penalty for an initial gap. Either way produces the same answer in this case.

Draw a line indicating a solution path in the table above, and show the actual alignment for that path below, using — to indicate any gaps. Is there more than one solution? Justify your answer.

Two solutions, depending on which green cell is picked.

```
One solution: G A A T T  C A G T T A
              G G A T -- C -- G -- -- A
One other:    G G A -- T C -- G -- -- A
```

Average: 8.54. Median: 9.
Most common mistakes: missing the second path, calculation errors (most common: missing a match)