MA/CSSE 473 Final Exam November 17, 2014 Name________________________

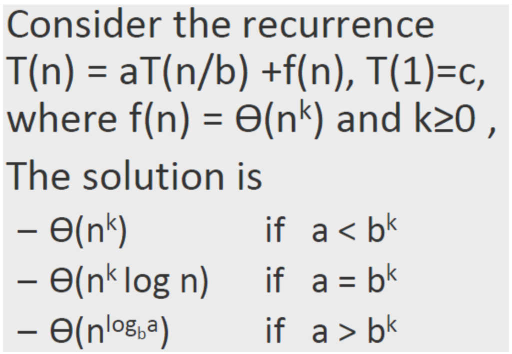
Resources allowed: Calculator, one 8.5 x 11 sheet of paper, handwritten, 2-sided.
Resources not allowed: Anything that can communicate or has headphones/earphones.

Caution! It is possible to get so caught up in getting all of the points for one problem and spend so much time on it that you do not get to the other problems. Don't do that! I will be generous with partial credit if you have the main ideas.

For Instructor use:

Description	Problem	Possible	Earned
Nested sums	1	10	
RSA	2	15	
Graph representations	3	9	
Kruskal Prim Warshall	4	25	
Fermat	5	6	
Closest Points	6	10	
Moldy Chocolate	7	12	
Huffman code	8	15	
fixheap	9	6	
Optimal BST	10	18	
Minimal Spanning Forest	11	10	
Heap decrease	12	4	
BONUS	bonus		8
	Total	140	

Unless I specify otherwise, for every problem you should show enough of your work to convince me that you know how to do the problem and did not just guess the answer. But you should keep your answers brief.



Consider the recurrence
$T(n) = aT(n/b) + f(n)$, $T(1)=c$,
where $f(n) = \Theta(n^k)$ and $k \geq 0$,
The solution is
– $\Theta(n^k)$ if $a < b^k$
– $\Theta(n^k \log n)$ if $a = b^k$
– $\Theta(n^{\log_b a})$ if $a > b^k$

1. (10) For each of the following three code snippets, give the big-Theta running-time as a function of n. Point values: 3, 2, 5

```
for (int i = 1; i ≤ n ; i++)
       for (int j = 1 ; j ≤ n*n; j++)
              for (int k=1; k ≤ j; k++)
                     total++;
```

Answer: Θ(   )

```
for (int i=1; i< n; i = i * 2)
       total++;
```

Answer: Θ(   )

```
for (int i = 1; i ≤ n; i++)
       for (int j=1; j ≤ i*i; j++)
          if (j%i == 0)
              for (int k=0; k < j; k++)
                     total++;
```

Answer: Θ(   )

2. (15) Bob has advertised his RSA public key as (N=91, e=31). Eve intercepts an encoded message m' that someone sends to Bob using Bob's public key, and that encoded message is: 2

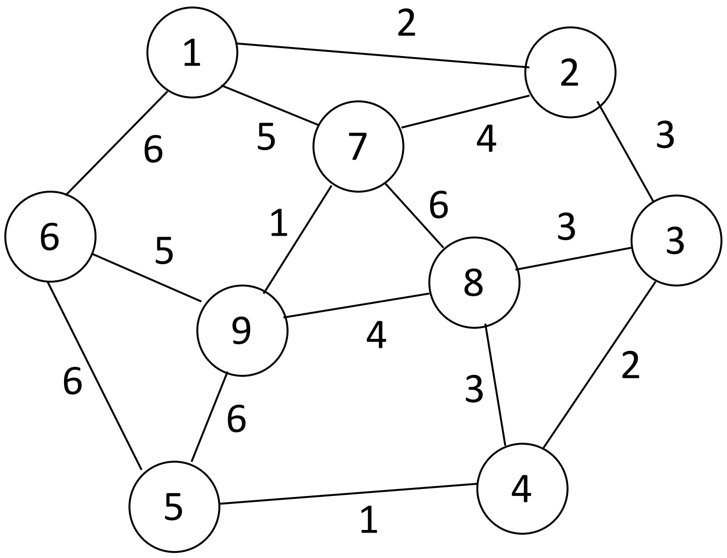
What was the original message m before it was encoded? __________
Briefly show how you get your answer; you do not have to write a lot, but convince me that you understand RSA.

3. (9) We have seen three main representations of the *graph* abstract data type. Name them, and for each representation, name an algorithm that we studied in the last few weeks of this course that is most efficiently implemented by using that representation.

	representation	algorithm
1		
2		
3		

4. (*25*) Consider the following graph.

(a) (7) Show (in order) the first 7 edges that Kruskal's algorithm adds to the MST that it generates. If there are multiple choices at some point, choose the edge whose lowest-numbered vertex is smallest. If two choices have the same lowest-numbered vertex, choose the edge whose other vertex number is lower. List each edge as a pair of vertex numbers. For example, an edge that may or may not actually not be in the MST is (8, 9), whose weight is 4.



1. ___________	5. ____________
2. ___________	6. ____________
3. ___________	7. _____________
4. ___________	

(b) (3) If we use Prim's algorithm to find an MST, starting at vertex 1, what does the minheap of the fringe vertices contain before any edges are added? (Whenever multiple vertices are added to the heap, add the lowest-numbered vertex first). Do not include any vertices whose current weight is infinite. You can just draw the tree representation of the heap; I am not looking for the contents of the *into*, *outof*, and *key* arrays. For each item in the heap, show both the vertex number and its weight (show the vertex # in parentheses).

(c) (2) Continuing with Prim from part b, what is the first edge to be added? Show the endpoints. _________

(d) (3) After that edge is added, show the state of the minheap of fringe vertices. You do not need to include any vertices whose current weight is infinite. Again, you can just draw the tree, showing the vertex number and weight for each vertex.

(e) (4) What are the next four edges to be added? __________ __________ __________ __________

(f) (6) If we perform Warshall's algorithm (dynamic programming algorithm for calculating transitive closure) on the appropriate representation of this graph (ignoring the edge weights), what is the smallest value of k for which $R_{2\,9}^{(k)}$ is different from $R_{2\,9}^{(k-1)}$?

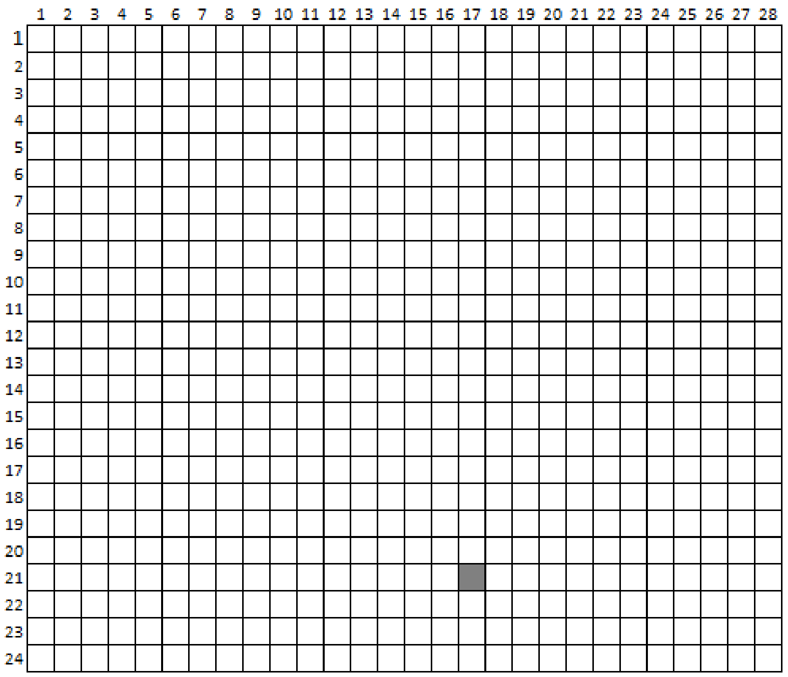
k = ______

5. (6) State Fermat's "little theorem":

6. (10) Give a high-level description of how the recursive Closest Points algorithm that we studied finds the two closest points among a set of N points in the plane in O(N log N) worst case time. Your description should include enough detail to make it clear how it "barely" manages to avoid being $\Theta(N^2)$. But keep it reasonably brief (brief enough to fit in the given space, please.

7. (12) Recall the moldy chocolate problem: two players take turns breaking off some number of rows or columns of a chocolate bar that has one moldy square. The player who ends up with only the moldy square loses. In the situation below, there are 28 rows and 24 columns; the moldy square is in row 21, column 17. Fill in the number and circle the choice that describes the unique move that guarantees a win for the first player if that player continues to make correct moves.
Clarification: There are 16 columns to the left, 11 to the right, 20 rows above, 3 below the moldy square.

Remove (number) ______ (circle one of the following) row(s) above the moldy square.
row(s) below
column(s) left of
column(s) right of



8. (15) How many total bits are in a Huffman-encoded message for *abracadabradabra* (There are seven a's, three b's, three r's, one c, and two d's)? You should not include the number of bits in the code table, just in the message itself. Of course, you should show how you get your answer.

Answer: __________

9. (6 points) Use the bottom-up fixheap (a.k.a. buildheap) algorithm to turn the following array into a **min** heap. Show the numbers in the array after the fixheap algorithm has run.

before		7	4	9	3	1	6	2	8	5
	0	1	2	3	4	5	6	7	8	9
after										

10. (18) This problem refers to the Optimal Binary Search Tree dynamic programming algorithm from class, where we examined a triangular grid similar to the one below. **Input parameters:**

index	0	1	2	3	4	5	6	7
letter		A	E	I	O	U	W	Y
a		8	2	10	6	4	16	3
b	1	4	7	3	2	3	12	3

Values in the output table: Recall that when the subscripts are ij, it is referring to a subtree that contains internal nodes i+1, …, j and external nodes i, …, j. In each square, the R value is number of the root of the optimal subtree, the C value is the weighted path length of that subtree, and the W value is the sum of the a and b values for all of the nodes in that subtree.

(a) (8) Show the values of R_{14} ____________ W_{14}__________ C_{14}____________ (points: 2, 2, 4)

(b) (4) In the space below, show how you use the dynamic programming algorithm to get W_{14} and C_{14} with only a few additions.

(c) (6) Also in the space below the grid, draw a diagram of the optimal tree, labeling each node by its letter value and the particular R value that shows that this node goes where it does. **Example** (may or may not be an actual labeled node in the optimal tree): A (R_{01}).

R00: 0	R01: 1	R02: 1	R03: 2	R04: 3	R05: 3	R06: 3	R07: 6
W00: 1	W01: 13	W02: 22	W03: 35	W04: 43	W05: 50	W06: 78	W07: 83
C00: 0	C01: 13	C02: 35	C03: 68	C04: 89	C05: 112	C06: 186	C07: 212
	R11: 1	R12: 2	R13: 3	R14:	R15: 3	R16: 6	R17: 6
	W11: 4	W12: 13	W13: 26	W14:	W15: 41	W16: 69	W17: 74
	C11: 0	C12: 13	C13: 39	C14:	C15: 81	C16: 150	C17: 172
		R22: 2	R23: 3	R24: 3	R25: 3	R26: 6	R27: 6
		W22: 7	W23: 20	W24: 28	W25: 35	W26: 63	W27: 68
		C22: 0	C23: 20	C24: 39	C25: 62	C26: 125	C27: 147
			R33: 3	R34: 4	R35: 4	R36: 6	R37: 6
			W33: 3	W34: 11	W35: 18	W36: 46	W37: 51
			C33: 0	C34: 11	C35: 27	C36: 73	C37: 95
				R44: 4	R45: 5	R46: 6	R47: 6
				W44: 2	W45: 9	W46: 37	W47: 42
				C44: 0	C45: 9	C46: 46	C47: 68
					R55: 5	R56: 6	R57: 6
					W55: 3	W56: 31	W57: 36
					C55: 0	C56: 31	C57: 53
						R66: 6	R67: 7
						W66: 12	W67: 17
						C66: 0	C67: 17
							R77: 7
							W77: 2
							C77: 0

11. (10 For Kruskal's Algorithm, the pseudocode from the Levitin textbook is:

```
ALGORITHM Kruskal(G)
//Input: a weighted connected graph G=<V,E>
//Output: $E_T$, the set of edges composing a minimum spanning tree of G
sort E in nondecreasing Order of the edge weights $w(e_{i1}) \leq \cdots \leq w(e_{i|E|})$
$E_T \leftarrow 0$
$encountered \leftarrow 0$
$k \leftarrow 0$
while $encountered < |V| - 1$ do
        $k \leftarrow k + 1$
        if $E_T \cup \{e_{ik}\}$ is acyclic
              $E_T \leftarrow E_T \cup \{e_{ik}\}$ ;
              $encountered \leftarrow encountered + 1$
return $E_T$
```

Suppose the graph is ***not*** connected. If we would like to get a minimum spanning *forest* from this algorithm, (each tree in the forest is a minimum spanning tree for one connected component of the graph), we don't need to build the solution code from scracth; we can change the above code to achieve that. Thinking about the differences between a connected graph and a graph with multiple connected components, how should we change the above code? You may write bwlow or chage the code above.

12. (4 points) Consider our standard representation of binary heaps, and suppose we want to write the procedure:

def decreasePriority(h, val, smallerval):

which takes a minheap h,

a priority value of some item that is in the heap, (you may assume that there is only one with that priority) new lower value to change that item's priority to.

(a) What is the worst-case running time for decreasePriority? ______________________
Why?

(b) If we use the indirect heap implementation with *into* and *outof* arrays, what wll the runtime for decreasePriority be?

____