University of Utah
School of Computing

CS 4150 **Midterm Exam** **March 3, 2016**

This is a closed-book exam, but you may refer to one sheet of notes. You have 80 minutes to complete the exam. Answer the questions in the space provided. The exam consists of 25 questions spread over 10 pages. Each question is worth 4 points; the entire exam will be graded out of 100 points. Give a concise and legible answer to each of the questions.

Read the questions carefully. If you do not understand what a question is asking, please ask for a clarification. Check the board periodically, as clarifications may be written there.

Do not discuss this exam with anyone who has not yet taken it. If you do, you will fail the course and you will be referred to the Student Behavior Committee.

Name: uNID:

Page 2	
Page 3	
Page 4	
Page 5	
Page 6	
Page 7	
Page 8	
Page 9	
Page 10	
Total	

Master Theorem

If $T(n) = aT(n/b) + O(n^d)$ for constants $a > 0$, $b > 1$, $d \geq 0$, then

$$T(n) = \begin{cases} O(n^d) & \text{if } d > \log_b a \\ O(n^d \log n) & \text{if } d = \log_b a \\ O(n^{\log_b a}) & \text{if } d < \log_b a \end{cases}$$

1. Quickselect, binary search, selection sort, and mergesort are implemented in different programming languages. Each implementation is then carefully timed to determine its average case running time on arrays of different lengths. A different computer is used to time each implementation.

The results of the timing experiments are summarized below. All times are expressed in the same (unspecified) units.

Length of array	Algorithm W	Algorithm X	Algorithm Y	Algorithm Z
4000	100	500	10	1000
8000	217	1000	40	1100
16000	467	2000	160	1200
32000	1000	4000	640	1300
64000	2136	8000	2560	1400

Identify the algorithms by writing W, X, Y, and Z in the appropriate boxes below.

Algorithm	Identity
Quickselect	
Binary search	
Selection sort	
Mergesort	

2. Indicate whether each of the following is a best practice for making timing measurements by writing "yes" or "no" in its box.

Advice	Yes or no?
Don't measure small intervals	
Account for timing overhead	
Avoid cold starts	
Time nothing that uses the heap	

3. Consider the following method:

```
public static int f (int n) {
  int total = 0;
  for (int i = 0; i < n; i += 2) {
    for (int j = 0; j < i; j++) {
      total++;
    }
  }
  return total;
}
```

What does `f(2*k)` return? Assume that $k > 0$. Give your answer in terms of k. *Give a simplified expression that does not contain a summation.*

Answer	

4. For each pair of functions below, write

- $\Theta(g)$ if f is $\Theta(g)$

Otherwise, write

- $O(g)$ if f is $O(g)$ or
- $\Omega(g)$ if f is $\Omega(g)$

f	g	Answer
$2n^2 - 10n$	$n^2 + 100$	
$n \log n$	$2n$	
n^2	n^3	
$n \log n$	$2n \log (n^2)$	

5. Each of the following algorithms has a $\Theta(n)$ worst case. Give the average case complexity of each.

Algorithm	Average case
Lookup in an n-element hash table	
Lookup in an n-element binary search tree	
Partitioning an n-element array around a pivot	
Adding to the end of an n-element dynamic array	

6. [6 points] Consider the following four functions:

- $f_1(n) = \log(n^3)$
- $f_2(n) = n \log n$
- $f_3(n) = n^2$
- $f_4(n) = 2^{\log_2 n}$

Put these functions in order of increasing asymptotic complexity by writing f_1, f_2, f_3, and f_4 in the appropriate boxes below.

Lowest asymptotic complexity	
Next highest asymptotic complexity	
Next highest asymptotic complexity	
Highest asymptotic complexity	

7. Use the Master Theorem to derive a tight upper bound on the following two recurrences:

$$\begin{aligned} T_1(n) &= 2\,T_1(n/3) + O(n) \\ T_2(n) &= 4\,T_2(n/2) + O(n \log n) \end{aligned}$$

Recurrence	Bound
T_1	
T_2	

8. Below are four statements about the type of blended algorithm that we studied. Classify each as true or false.

Statement	True or False?
For sufficiently large problem sizes, a blended algorithm is faster than both of the algorithms from which it is formed.	
A blended algorithm is asymptotically faster than both of the algorithms from which it is formed.	
At least one of the algorithms from which a blended algorithm is formed must be a divide and conquer algorithm.	
Both of the algorithms from which a blended algorithm is formed must be divide and conquer algorithms.	

9. Consider this method g. It operates on an array of integers of length n, where $n > 0$.

```
// A must be non-empty.

public static int g (int[] A) {
  if (A.length < 5) {
    return A[0];
  }
  else {
    int total = 0;
    for (int i = 0; i < 5; i++) {
      total += g(chooseRandomly(A, A.length/5));
    }
    return total;
  }
}


// A must be non-empty.  Assume that rand is a random
// number generator (a Random object).  Returns an
// array of k randomly chosen elements from A

public static int[] chooseRandomly (int[] A, int k) {
  int[] B = new int[k];
  for (int i = 0; i < k; i++) {
    B[i] = A[rand.nextInt(A.length)];
  }
  return B;
}
```

Find a recurrence relation of the form

$$T(n) = a\,T(n/b) + O(n^d)$$

that expresses the running time $T(n)$ of g on arrays of length n. (A random integer can be chosen in constant time.) Below, give your values for a, b, and d. Also, use the Master Theorem to find a tight $O()$ bound for the recurrence.

	Answer
a	
b	
d	
Bound	

10. Below are four statements about sorting algorithms. Classify each as true or false. *We are not interested in modifications that simply check if the array is sorted and return immediately if so.*

Statement	True or False?
Basic quicksort uses $\Omega(n)$ additional space in the worst case, but can be modified to use $O(\log n)$ additional space in the worst case.	
Basic mergesort uses $\Omega(n)$ additional space in the worst case, but can be modified to use $O(\log n)$ additional space in the worst case.	
Basic quicksort uses $\Omega(n \log n)$ time in the best case, but can be modified to use $O(n)$ time in the best case.	
Basic mergesort uses $\Omega(n \log n)$ time in the best case, but can be modified to use $O(n)$ time in the best case.	

11. Adding a pair of n-by-n matrices requires $O(n^2)$ time. The straightforward algorithm for multiplying a pair of n-by-n matrices requires $O(n^3)$ time. (Pairs of 1-by-1 matrices can be added and multiplied in constant time.)

In 1969, Volker Strassen showed how to reduce the problem of multiplying one pair of n-by-n matrices to the problem of multiplying seven pairs of n/2-by-n/2 matrices and adding a few n/2-by-n/2 matrices.

Strassen used this idea to design a revolutionary divide and conquer algorithm for multiplying matrices. Find a recurrence relation of the form

$$T(n) = a\,T(n/b) + O(n^d)$$

that expresses the running time $T(n)$ of Strassen's algorithm on pairs of n-by-n matrices. Below, give your values for a, b, and d. Also, use the Master Theorem to find a tight $O()$ bound for the recurrence. If your solution involves an exponent, give its exact value. Do not simplify it to an approximate numerical value.

	Answer
a	
b	
d	
Bound	

12. Give one circumstance in which a balanced binary search tree would be preferable to a hash table for representing a set.

Answer	

13. Use O() notation to give the tightest possible upper bounds on the times required to answer each of the following questions about a directed graph G consisting of V vertices and E edges. Answer each question once assuming that the graph is represented as an adjacency matrix (two-dimensional array) and again assuming that the graph is represented as an adjacency list (array of linked lists).

Question	Matrix	List
Does G have a cycle?		
Is there a path from vertex u to vertex v?		

14. Answer the following questions about directed graphs with V vertices and E edges.

Question	Answer
Is it possible to topologically sort any such graph?	
If such a graph is a dag, how many strongly connected components does it have?	

15. Below are statements about a call to the depth-first search algorithm (dfs) on a graph G. Classify each as true or false.

Statement	True or False?
If G is undirected, dfs will call explore once per connected component	
If G is directed, dfs will call explore once per strongly connected component	
If G is undirected, dfs will run in time proportional to the size of G's representation	
If G is directed, dfs will run in time proportional to the size of G's representation	

16. Let G be a directed acyclic graph on which a depth-first search has been performed. Classify the following statements as true or false.

Statement	True or False?
The vertex with the smallest pre time must be a source	
The vertex with the largest pre time must be a source	
The vertex with the smallest post time must be a sink	
The vertex with the largest post time must be a sink	

17. How many depth-first searches does the strongly connected component algorithm that we studied perform?

Answer	

Let G be a graph whose meta-graph consists of more than one meta-vertex. Under what circumstance will it be possible to add a single edge to G that converts it into a graph that has a single strongly connected component?

Answer	

18. Let G be a graph with V vertices and E edges. Define "sparse" and "dense" using asymptotic complexity notation.

Term	Definition
G is sparse	
G is dense	

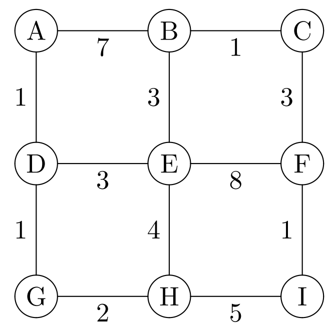
19. Let G be a weighted, directed, acyclic graph with V vertices and E edges. At most how many update operations will be applied to G by the shortest paths algorithm for dags?

Answer	

20. Let G be a weighted, directed graph with V vertices and E edges. In the best case, how many update operations will be applied to G by the Bellman-Ford algorithm? (Assume that the algorithm employs the early termination optimization discussed in class.)

Answer	

The next two questions concern this undirected, weighted graph Y.



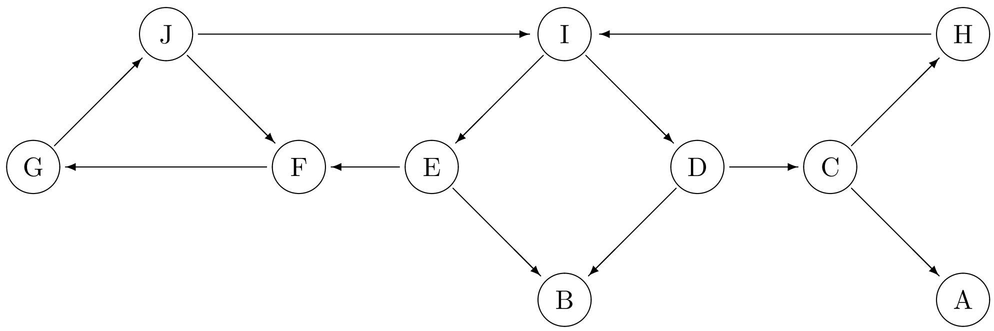
21. Suppose a breadth-first search is performed on Y, starting at vertex A (ignore the weights). What will be the number of edges in the path with the most edges?

Answer	

22. Suppose the Bellman-Ford algorithm is used on Y to find a shortest paths tree rooted at vertex A. What will be the number of edges in the path with the most edges?

Answer	

The directed graph X diagrammed below is the subject of the next three questions.



23. Suppose that you do a depth-first search of X. Suppose also that whenever there is an arbitrary choice of vertices to visit (in either `dfs` or `explore`), you always pick the one that comes first in the alphabet. What will be the pre number of E, the post number of H, the pre number of I, and the post number of J?

Pre number of E	
Post number of H	
Pre number of I	
Post number of J	

24. Answer the following questions about X.

Question	Answer
How many strongly-connected components (SCCs) does X have?	
How many sink SCCs does X's meta-graph have?	

25. Answer the following questions about X.

Question	Answer
What two edges must be removed from X to convert it into a dag whose only two sources are G and H?	
What two edges must be removed from X to convert it into a dag whose only source is J?	