CS1501 Fall 2018

Practice Questions for Midterm Exam

Here are some example questions that may help you to study for the Midterm Exam. Try to answer the questions fully before looking at the answers. Though these questions indicate some of the material that may be on the exam, they are by no means comprehensive. For details on the topics covered see the Midterm Exam Study Guide on CourseWeb.

Fill in the Blanks Complete the statements below with the MOST APPROPRIATE words/phrases.

a) Given N keys, each with b bits, in a digital search tree, the WORST CASE search time for a key in the tree requires ____________________ key comparisons, while the AVERAGE CASE search time requires ________________ key comparisons.

b) Delete is a problem with open-addressing hashing because ______________________________ ____.

c) If I have a linear probing hash table of size M, and a cluster of size C, the probability that a random key will be Inserted into the location immediately after the cluster is ______________.

d) The mismatched character heuristic of the Boyer-Moore algorithm has a best case run-time of ____________.

e) If an encoding scheme is prefix-free, it is certain that ___________________________________ _____________________________.

f) In the 8-Queens problem each recursive call attempts to ________________________________ _________________.

g) Given a multiway radix search trie in which 32-bit keys are compared 4 bits at a time, the maximum height of the tree is __________ and interior nodes will each have up to ______ children.

h) Consider an empty separate chaining hash table of size 100. If we hash 200 keys into this table, the average chain length will be _________ and the worst case chain length will be ________.

i) Given a pattern of length M and a text of length N, the brute-force string matching algorithm discussed in lecture will require time ______________________ in the normal (average) case and time ____________________ in the worst case.

True/False Indicate whether each of the following statements is True or False. For False answers, **explain why it is false.**

a) The brute-force algorithm to crack an n-digit PIN number has an upper-bound run-time of Theta(n!).

b) A multiway trie that considers k bits at a time has 2^k branches at each node.

c) A good hash function should utilize the entire key.

d) The KMP string matching algorithm improves over the brute-force algorithm in the normal (average) case.

e) When a file is compressed, its entropy level is decreased.

f) The Theta asymptotic bound is an exact bound (within a constant factor) – it is both an upper and a lower bound on the asymptotic performance.

g) Pruning is a technique that improves the asymptotic run-times of exhaustive search algorithms.

h) The Java String operator "+" appends one String to another in constant time (relative to the lengths of the Strings).

i) A problem with regular multiway radix search tries is that they consume large amounts of memory due to the extensive branching at each node.

j) I can avoid collisions in hashing as long as the size of my hash table, M is greater than the amount of data I am storing, N.

Short Answers and Calculations

1) You have two programs, Program A, which runs in time k_1N and Program B, which runs in time $k_2\log_2(N)$ for some constants k_1 and k_2. Assume that for a problem of size N_o, both programs take X seconds to execute. Approximately how much time would each program take to run if we double the problem size? Show your work.

2) Define what it means to have a collision in a hash table, and why we cannot usually prevent them from occurring.

3) Consider a file containing the following text data:

```
AAABBBAAB
```

Trace the LZW encoding process for the file (in the same way done in handout lzw.txt, so each "step" produces a single codeword). Assume that the extended ASCII set will use codewords 0-255. For each step in the encoding, be sure to show all of the information indicated below. Note: The ASCII value for 'A' is 65.

```
STEP # PREFIX MATCHED CODEWORD OUTPUT (STRING, CODE) ADDED TO DICTIONARY
------ -------------- --------------- ----------------------------------
```

4) Consider 2-pass Huffman compression. How would it perform on each of the following files and why? Be specific by giving approximate compression ratios in each case.

a) A file containing 1000 of each ASCII character

b) A file containing 1000000 As

5) Consider the **mismatched character heuristic** of the **Boyer-Moore** string matching algorithm. For the pattern and text strings shown below, state and **justify** how many **total character comparisons** must be done in order to match the pattern shown within the text string. Justify your answer using the **right array** for the pattern.

Text:

ABCDXABCDYABCDZABCDE

Pattern:
ABCDE

6) Justify in detail how many character comparisons are required to find a string in a DLB in the worst case. Assume that your DLB has N strings, each with a maximum of K characters, and that your alphabet has S possible characters in it.

7) Consider the crossword puzzle algorithm that you implemented in Assignment 1. Consider the board below.

	0	1	2	3
0	H	E	L	P
1	A	N	+	+
2	+	+	+	+
3	+	+	+	+

You are currently looking at square [1][2] and proceeding through the board in a row-wise fashion.

a) Assuming that you will not backtrack to square [1][1], what is the maximum number of recursive calls (to square [1][3]) that is possible from this square? Justify your answer.

b) Assuming a "typical" dictionary of English words, how likely is this maximum number to occur? Thoroughly justify your answer.

8) Consider an emtpy de la Briandais Tree, which uses the lower case letters (plus a string termination character) as its alphabet, using the implementation that we discussed in lecture. Also consider the following strings: **then them the this blue glue**. Draw the de la Briandais tree that results after inserting the strings shown in the order shown above.

9) Consider the two open addressing hash tables, with **h(x) = x mod 13**, shown below. Also, consider the following keys (in order): 23, 17, 36, 24, 26, 37, 49.

a) Assume that **linear probing** is being used for collision resolution. Show the table after the keys shown above are inserted in the order shown above.

b) Assume now that **double hashing** is being used for collision resolution, with **h_2(x) = (x mod 11) + 1**. Show the table after the keys shown above are inserted in the order shown above. For full credit for each collision indicate the h_2(x) value for the key.

Linear Probing	
Index	key
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	

Double Hashing	
Index	key
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	

Coding

1) Assume that you are using **linear probing** in a simple hash table of **Strings.** Function h(x) is defined as we discussed it in class (and you do NOT have to write it). Complete the code for the Find function below, which will return true if the item is present and false otherwise. Be sure to handle ALL possibilities.

```
String [] table; // instance variable
// Other methods not shown. Note that all table locations are initialized to null
// prior to any Inserts. Assume that no Deletes are allowed. Fill in the code.
public boolean find(String item)
{
     int index = h(item);









}
```