Final Exam

CS 311 Data Structures
Fall Quarter 2002

9:00am - 11:00am, Friday, December 13, 2002

THIS EXAM IS *CLOSED BOOK, CLOSED NOTES* .

Print your name neatly in the space provided below; print your name at the upper right corner of every page.

Name:

This booklet should have 6 sheets. If it does not, report this to the instructor.

Work efficiently. Do not get stuck too long on any one problem.
Try to answer every question. Partial credit will be given for incomplete answers.
Show your work.

Problem	Max. Points	Your Score
1	11	
2	19	
3	10	
4	10	
5	14	
6	11	
Total	75	

1. $\boxed{\text{MULTIPLE CHOICE (11 points)}}$

There is only one correct answer to each question.

(a) **In a randomized skip list what is the probability that 15 successive insertions all create level-1 nodes?**

A. $(1/2)^{15}$
B. $(1/4)^{15}$
C. $(1/8)^{15}$
D. 0
E. 1

(b) **Which of the following statements is true about a maximum-based priority queue's binary tree representation?**

A. The tree can be any binary tree and the maximum is at the root.
B. The tree can be any binary tree and the maximum is at a leaf.
C. The tree is complete and the maximum is at a leaf.
D. The tree is complete and the maximum is at the root.
E. The maximum could be anywhere in the tree.

(c) **Which of the following data structures are defined in the STL?**

```
(I)   Singly Linked List
(II)  Doubly Linked List
(III) Binary Search Tree
(IV)  Priority Queue
```

A. (I) and (II) only
B. (I), (II) and (III) only
C. (II) and (IV) only
D. (II), (III) and (IV) only
E. (II), (III) and (IV) only

(d) **Which one of the following statements is true for a B-tree?**

A. All entries in a node are greater than or equal to the entries in the node's children.
B. All leaves are at the same depth.
C. All nodes contain the same number of keys.
D. All non-leaf nodes have the same number of children.
E. The root always has more than two children.

(e) **Which operation can NEVER occur when deleting from a B-tree?**

A. The tree becomes shorter
B. A node is split
C. Two nodes are merged
D. A node has fewer keys than before

E. Any one of the above operations may occur during deletion.

(f) **Consider the following min-heap:**

2	4	6	8	10	12	14	16	18	20

Which of the following is the right child of 6?

A. 10
B. 12
C. 14
D. 16
E. 20

(g) **You have a priority queue Q implemented with a max-heap. What will Q look like after the following sequence of operations?**

```
Q.Insert(15)
Q.Insert(10)
Q.Insert(20)
Q.Insert(5)
Q.Insert(25)
Q.ExtractMax()
```

A.

20	15	10	5

B.

20	10	15	5

C.

15	10	20	5

D.

10	8	9	13

E.

5	10	15	20

(h) **Where in a max-heap is the smallest element?**

A. At the root
B. It is the leftmost leaf
C. It is the rightmost leaf
D. It could be in any leaf
E. It could be in any internal node

(i) **Which of the following algorithms *could* be used to find a shortest path between two vertices in a graph?**

```
(I)   Depth-First Search
(II)  Breadth-First Search
(III) Prim's algorithm
(IV)  Kruskal's algorithm
```

A. (I) and (II) only
B. (III) and (IV) only
C. (I), (II), (III) and (IV)
D. (II), (III) and (IV) only
E. (II) only

2. SHORT ANSWERS (19 points)

In the questions that ask you to select the best data structure for a specific situation give only ONE data structure as your answer.

(a) If we remove the color from a red-black tree, will it become an AVL tree? If yes, explain why. If no, give an example to demonstrate this.

(b) When we insert a new node in a red-black tree we color the node red. Why?

(c) Give one major advantage and one major disadvantage that a skip list has when compared to a singly-linked list.

(d) We would like to reverse a sequence 1, 2, 3, 4 by inserting those values into a data structure (in that order) and then removing each one. What data structure is best suited to do this reversal?

(e) We wish to perform a search for a value stored in a data structure that has 100 elements. If the search is guaranteed to not have to look at all the elements, what is the data structure?

(f) We wish to maintain patient records in a hospital. The records of a patient who is in hospital are extremely active, being consulted and updated continually. When the patient leaves, the records become much less active, but may still be needed occasionally (i.e. they should not be deleted). If the patient is readmitted, then the records become active again. In addition, since this may be an emergency, the records should not have been stored in backup, but they should be quickly available. What is the most suitable data structure to store all the patient records?

(g) A brave computer hacker named Neo has obtained information about the many artificial intelligence agents that live inside The Matrix (a virtual reality world used to imprison the culturally unaware). He now needs to track them down one-by-one and erase them. In order to do this he needs to carry the information on all the agents in his laptop, so that whenever he comes across an agent he can retrieve the agent's info based on the name of the agent and determine how to best erase that agent. Since Neo doesn't expect he will have much time for such an information lookup before the agent notices him, he wants the *expected-case lookup time to be very fast*. If Neo's laptop is quite fast and has all the memory he might need (i.e. there are no resource limitations), what data structure should Neo use to store the agent information records in order to be able to expect that they can be looked up by name as quickly as possible?
Briefly justify your answer.

(h) Neo has decided to bring his good friend Jar Jar into the Matrix with him to track down the agents. Unfortunately, Jar Jar demands Neo leave the good laptop behind and bring Jar Jar's old laptop instead. The amount of memory this laptop has is not large enough to store more than one agent's information record. Therefore, the information records for each agent will need to be stored on the laptop's very slow hard drive instead. This being the case, what data structure should be used to store the information records now (again keeping in mind that the record needs to be obtained as quickly as possible)?
Briefly justify your answer.

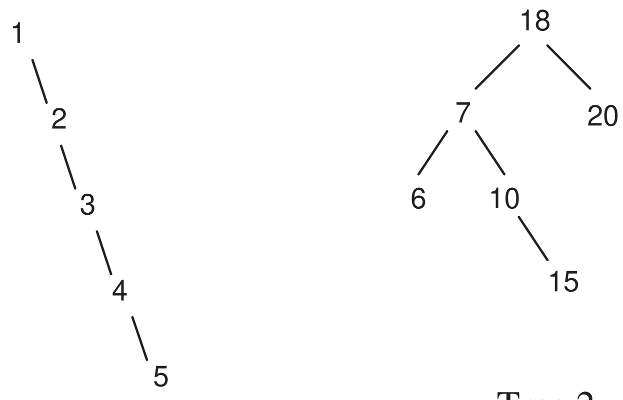
(i) You want to write a program that can translate from English to Klingon. You want to be able to type a word in English and the corresponding word in Klingon will be printed out.

i. Suppose you want your program to have the fastest possible search time in the **worst** case, regardless of how much memory you would need for that. What data structure will you use to store the English-Klingon pairs? Briefly justify your answer.

ii. Now suppose that you want to use as little memory as possible regardless of how this would affect search or insertion time. Storing the data in the hard disk is not an option. What is the best data structure for this situation? Briefly justify your answer.

3. $\boxed{\text{SPLAY TREES (10 points)}}$

Consider the following splay trees:



Tree 1 Tree 2

(a) Show the resulting tree after accessing node 5 in Tree 1.

(b) Show the resulting tree after accessing node 10 in Tree 2.

4. HASHING (10 points)

Insert the values 12, 4, 3, 22, 17, 13 in that order into the hash table below. $h(k) = k \ mod \ 9$ and $h_2(k) = (k \ mod \ 5) + 4$. The table below shows the actual values of the functions so you don't need to compute them. The hash table is an array with zero-indexing.

k	12	4	3	22	17	13
$h(k)$	3	4	3	4	8	4
$h_2(k)$	6	8	7	6	6	7

(a) Use $h()$ for hashing. Use quadratic probing to resolve collisions.

(b) Use $h()$ for hashing. Use double hashing with $h_2()$ to resolve collisions.

5. DISJOINT SETS (14 points)

(a) Consider the disjoint sets structure below which uses **path compression** and **union-by-size**. Give the resulting tree after the operations **UNION(13, 40)** and **FIND(20)**.

(b) When using trees to represent disjoint sets, path compression cannot work efficiently with union-by-height. Why is that? How is this problem solved?

(c) Recall that we can use disjoint sets to compute the connected components of an undirected graph $G = (V, E)$ where V is the set of vertices and E the set of edges. The algorithm is as follows:

```
algorithm CONNECTED_COMPONENTS(G):
for each vertex v in V
    MAKE_SET(v)
for each edge (u,v) in E
    if FIND(u) != FIND(v)
        UNION(u,v)
```

Suppose we are executing CONNECTED_COMPONENTS on an undirected graph that has k components.

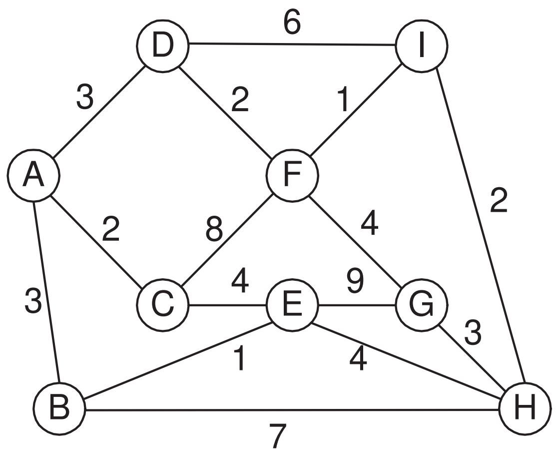
What is the **exact** number of times that the FIND operation is executed?

What is the **exact** number of times that the UNION operation is executed?

NOTE: express your answers in both parts in terms of k, $|E|$, $|V|$. You do not need to justify your answers.

6. GRAPHS (11 points)

In this problem, you will perform Prim's algorithm on the graph shown below.



(a) List the edges in the order that they are inserted in the minimum spanning tree, one step at a time. For steps 3 and 4 **briefly** explain why you chose those edges.

Step 1: .
I am starting at vertex ....

Step 2:

Step 3:

Step 4:

Step 5:

Step 6:

Step 7:

Step 8:

Step 9:

(b) On the graph shown above, **clearly** mark the edges that comprise the minimum spanning tree you found.

SCRATCH PAPER