COMPSCI 242-001 Data Structures and Algorithms (Spring 2004)
Exam #2 (100 points), 4/06/2003 (Tuesday)

• Q1(20 points): Sort in Linear Time

(a)(5 points) In the Counting-Sort algorithm, we first build an array \( C \) where \( C[i] \) contains the number of elements in the input less than or equal to \( i \). For a given input array \( A : < 4, 1, 5, 4, 7, 3, 5, 7, 4 > \), what is the content in \( C \) after executing line 1 to line 8 of the Counting-Sort of the book?

(b)(5 points) Given an input array \( A : < 865, 128, 536, 982, 12, 430, 189 > \), please use Radix-Sort to sort them. Show your answer step by step.
(c)(5 points) Given an input with $n$ numbers, Counting-Sort may not be able to sort those $n$ numbers in linear time, but sometime it does. What kind of input can let Counting-Sort runs in linear time?

(d)(5 points) Given an input with $n$ numbers, Radix-Sort may not be able to sort those $n$ numbers in linear time, but sometime it does. What kind of input can let Radix-Sort runs in linear time?
• **Q2 (25 points): Stacks, Queues and Linked List**

(a) (5 points) Which one of the following implementations of a linked list is the best for inserting an element to the end of the list?
   
   (a) singly circular list
   (b) singly non-circular list
   (c) doubly circular list
   (d) doubly non-circular list

(b) (10 points) Describe how to use two stacks to implement a queue so that `enqueue` runs in $O(1)$ and `dequeue` runs in $O(n)$. Suppose that the stacks have no size limit.
(c)(10 points) Please write the pseudocode for List-Insert-After(head, x, y). This procedure is to insert a new node y after an existing node x in a doubly circular list.

- **Q3(25 points): Hashing**

  (a)(10 points) Suppose we would like to insert a sequence of numbers < 60, 29, 4, 27, 36 > into a hash table with size 8 using open addressing with the primary hash function \( h_1(k) = k \mod m \). Please insert the numbers into the table below using linear probing, using quadratic probing with \( c_1 = c_2 = 1/2 \), and using double hashing with the secondary hash function \( h_2(k) = 1 + (k \mod 7) \)

<table>
<thead>
<tr>
<th>index</th>
<th>linear</th>
<th>quadratic</th>
<th>double</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
(b) (5 points) In the above example, which open-addressing technique(s) will fully utilize the table.

(c) (10 points) A hash table with size 20 stores 9 elements. These 9 elements are stored in $T[0]$, $T[4]$, $T[5]$, $T[6]$, $T[10]$, $T[11]$, $T[12]$, $T[18]$, $T[19]$. Suppose that all the other entries contain no “deleted” flag. An entry has a “deleted” flag means that this entry stored an element before, but the element has already been deleted. If we would like to search an element with a key $k$ and assume the linear probing technique is used, what is the expected number of probes for an unsuccessful search?
• Q4(10 points): Binary Search Trees

For a given input array $A : < 6, 9, 3, 2, 1, 7, 8, 4, 5 >$

(a)(5 points) What is the resulting binary search tree after inserting the numbers in the list to an initially empty tree?

(b)(5 points) From the tree you have built in question (a), what is the resulting tree after deleting the value 6?
• Q5(20 points): AVL Trees

(a)(10 points) For the sequence of numbers \{9, 8, ..., 1\}, suppose we would like to construct an AVL tree by successively inserting those numbers one at a time, starting with an empty tree. Please draw the sequence of AVL trees after inserting each of the 9 numbers.
(b) (10 points) Suppose we have an AVL tree as below. If we would like to delete all the numbers from the tree in the sequence \( \{1, 2, \ldots, 9\} \), please draw the resulting AVL trees after deleting each of the 9 numbers.

```
        4
      /  \
     2    6
    /   /  \
   1   3   5
    /  \
   8
```

```