$\alpha$	1	4 7	r		4 •	
<b>\fi</b> 11	$\alpha$	าโ	Info	rm	วราก	n
ισι	uu		ши	,, ,,,,,	auv	11

Student Intol mation					
Name:	Student ID:				
Due Date: 01 Nov 11:59pm.					
Submit answers on eDimension in pdf f	ormat. Submission without student information will NO				
be marked! Any questions regarding the tact information on eDimension).	e homework can be directed to the TA through email (con				

## **Exercise 1 Heap Sort**

- 1. What are the minimum and maximum numbers of elements in a heap of height h?
- 2. What is the running time of HEAPSORT on an array A of length n that is already sorted in increasing order? What about decreasing order?
- 3. Where in a min-heap might the biggest element reside, assuming that all elements are distinct?

# **Exercise 2 Binary Search Tree**

Suppose we have int values between 1 and 1000 in a BST and search for 525. Which of the following **cannot** be the sequence of keys examined, and **why**?

- (a) 4 300 800 500 823 525
- (b) 300 325 700 699 650 510 520 525
- (c) 900 873 850 300 412 600 570 550 400 525
- (d) 700 670 600 300 350 379 400 570 550 510 525

#### **Exercise 3 AVL Tree**

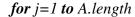
- 1. Insert the following sequence of elements into an AVL tree (not ordinary BST), starting with an empty tree: 22; 33; 25; 38; 43; 28; 29; 31. Show steps by drawing.
- 2. Delete 28 from the AVL tree in question1. Show steps by drawing.
- 3. Delete 43 from the AVL in question1 (note that node 28 is added back to the AVL tree). Show steps by drawing.

## **Exercise 4 Sorting**

```
COUNTING-SORT(A, B, k)
   let C[0..k] be a new array
   for i = 0 to k
 3
        C[i] = 0
 4 for j = 1 to A. length
        C[A[j]] = C[A[j]] + 1
 6 // C[i] now contains the number of elements equal to i.
 7 for i = 1 to k
 8
        C[i] = C[i] + C[i-1]
 9
   //C[i] now contains the number of elements less than or equal to i.
10 for j = A. length downto 1
11
        B[C[A[j]]] = A[j]
12
        C[A[j]] = C[A[j]] - 1
```

**Figure 1**: Counting sort (taken from CLRS book).

1. Refer to Figure 5 for the counting sort algorithm. Suppose we are asked to rewrite the **for loop** header in **line 10** of the counting sort algorithm as:



Does the algorithm still works properly after this modification? Provide reasons.

- 2. Describe an algorithm that, given n integers in the range of 0 to k, preprocesses its input and then answers any query about how many of the n integers fall into a range [a...b] in O(1) time. Your algorithm should use  $\Theta(n+k)$  preprocessing time. Hint: use the counting-sort algorithm.
- 3. Here are 11 different words: *hat, ten, hen, two, pan, one, tea, rat, rag, box, bat.* Sort the words in increasing alphabetical order with **radix sort**. Show all steps needed.

# **Exercise 5 Hashing**

Suppose that we are given a key k to search for in a hash table with positions 0, 1, ..., m - 1, and suppose that we have a hash function h mapping the key space into the set  $\{0, 1, ..., m - 1\}$ . The search scheme is as follows:

1. Compute the value j = h(k), and set i = 0.

- 2. Probe in position j for the desired key k. If you find it, or if this position is empty, terminate the search.
- 3. Set i=i+1. If i now equals m, the table is full, so terminate the search. Otherwise, set  $j=(i+j) \bmod m$ , and return to step 2.

Assume that m is a power of 2.

Show that this scheme is an instance of the following general quadratic probing scheme,

$$h'(k,i) = (h(k) + c_1i + c_2i^2) \bmod m.$$

Find the constants  $c_1$  and  $c_2$ .

