Fall 2018, CMPSC 465: Exam 3.
Closed book and closed notes, no 'cheat sheet', no calculators allowed.
Please don't use cell phones during the exam.
Answer questions in the space provided.
The exam is for 40 points.

NAME: ______________________ SECTION: ____________

1. Draw the 11-item hash table resulting from hashing the keys 2, 4, 7, 8, 12, 13, 19, 20, 25, 28, and 30, using the hash function $h(i) = (2i + 5) \bmod 11$ and assuming collisions are handled by chaining.
(**6 points**)

6

0 → [25]
1 → [20]
2 → [4]
3 → ∅
4 → ∅
5 → ∅
6 → [28]
7 → [12]
8 → [7]
9 → [2 | 13]
10 → [8 | 19 | 30]

i	$h(i)$	
2	9 mod 11 =	9
4	13 " "	2
7	19 " "	8
8	21	10
12	29	7
13	31	9
19	43	10
20	45	1
25	55	0
28	61	6
30	65	10

-1 or -2 if table is incorrect and no calculations are shown

-1 if table entries are incorrect (storing i instead of $h(i)$)

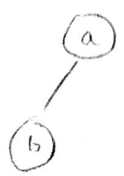
1 pt for open addressing

2. Let T be an ordered tree with more than one node. Is it possible that the preorder traversal of T visits the nodes in the same order as the postorder traversal of T? If so, give an example; otherwise, argue why this cannot occur. Likewise, is it possible that the preorder traversal of T visits the nodes in the reverse order of the postorder traversal of T? If so, give an example; otherwise, argue why this cannot occur.
(**6 points**)

No. Preorder traversal always visits root node first.
1 Postorder traversal " " " " last 2

Yes,

1 Consider



, Preorder → a, b 2

Postorder → b, a

3. Complete the pseudocode for the InsertAfter(p, e) operation in a doubly-linked list, where a new node with element e is inserted between node p and its successor. What is the worst-case asymptotic running time?
(5 points)

Algorithm insertAfter(p, e):

Create a new node v

v.element $\leftarrow e$

1 v.prev ← p /* link v to its predecessor */

1 v.next ← p.next /* link v to its successor */

1 (p.next).prev ← v /* link p's old successor to v */

1 p.next ← v /* link p to its new successor, v */

return v

Running time: O(1) 1

4. Describe how to implement a queue using two stacks. What is the worst-case running time of the **enqueue**() and **dequeue**() methods in this case?
(6 points)

Use two stacks S_1 and S_2

For enqueue, push to S_1 → O(1) time 1

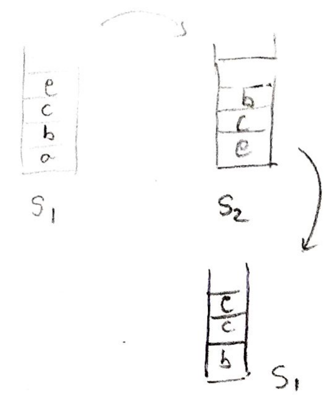
3 for explanation

For dequeue, if S_1 already has k elements, pop k-1 elements and push to S_2. → O(k) time.

Return the last element in S_1 after popping it

Push elements in S_2 back to S_1 and empty S_1

→ O(k) time.



In the worst case, dequeue takes O(n) time 2

Alternately, you can do O(1) dequeue and O(n) enqueue.

5. A *matched string* is a sequence of {, }, (,), [, and] characters that are properly matched. For example, "{{()[]}}" is a matched string, but this "{{()]}" is not, since the second { is matched with a]. Show how to use a stack so that, given a string of length n, you can determine if it is a matched string in $O(n)$ time.
(6 points)

Insert on seeing left symbol 2

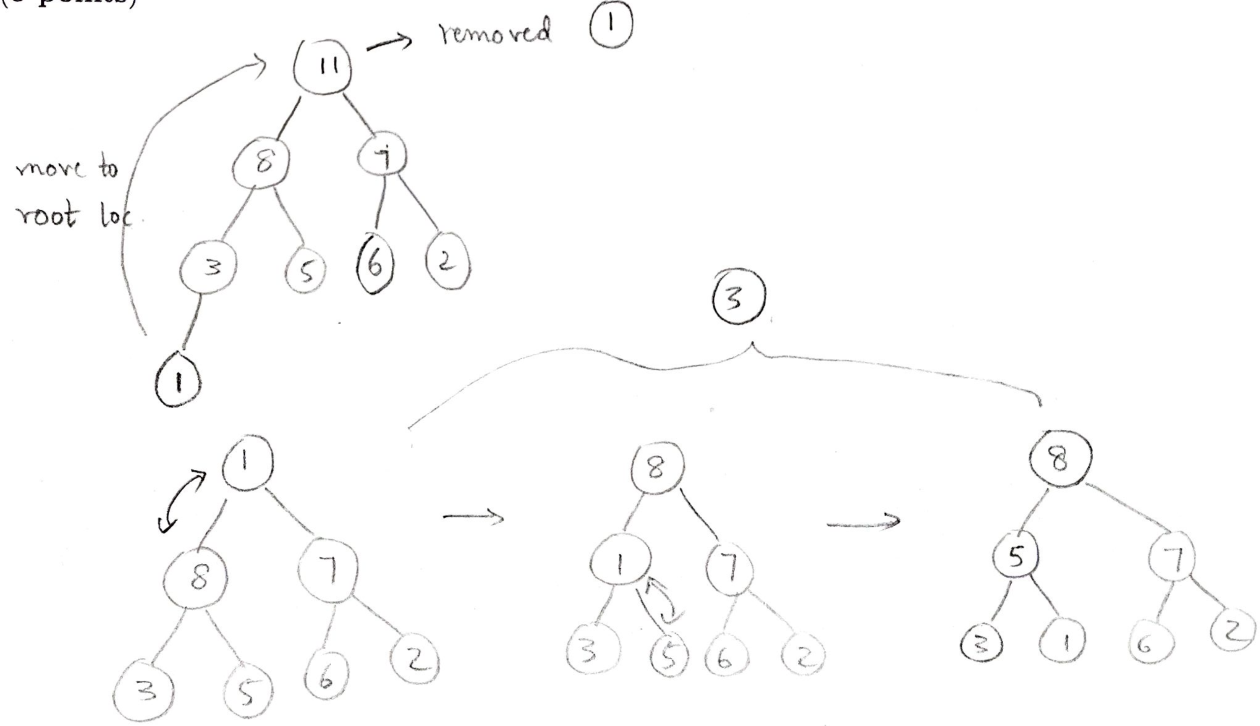
Remove on seeing right symbol 2

If left & right don't match, then it is not a matched string 1

Continue until end of string is reached 1

6. Suppose we perform a DeleteMax operation on the max heap $H = [11, 8, 7, 3, 5, 6, 2, 1]$ (the heap is stored here implicitly in the form of an array). Show the steps performed after deletion to restore the heap order of elements.
(5 points)

1



7. Suppose T is a binary search tree of height 4 (including the external nodes) that is storing all the integers in the range from 1 to 15, inclusive. Suppose further that you do a search for the number 11. Explain why it is impossible for the sequence of numbers you encounter in this search to be (9, 12, 10, 11).
(6 points)

tree → 2 points
tree structure

uniqueness → 1 point

first node in the sequence has to be 8. → 3 points