CS3000: Algorithms & Data — Summer 2025 — Laney Strange

Homework 3
Due Friday May 30th @ 9pm via Gradescope

Name:
Collaborators:

- Put your name on the first page. If you are using the LaTeX template we provided, then you can make sure it appears by filling in the `yourname` command.
- This assignment is due Friday May 30th @ 9pm via Gradescope. You may submit up to 48 hours late for no penalty, but expect a delay in grading.
- Show ALL your work, even if the problem doesn't specify it.
- You'll have an opportunity to resubmit one homework at the end of the semester.
- Solutions must be typeset, preferably in LaTeX. If you need to draw any diagrams, you may draw them by hand as long as they are embedded in the PDF. I recommend using the source file for this assignment to get started.
- Any solutions that include pseudocode must be in the CLRS style.
- I encourage you to work with your classmates on the homework problems. *If you do collaborate, you must write all solutions by yourself, in your own words.* Do not submit anything you cannot explain. Please list all your collaborators in your solution for each problem by filling in the `yourcollaborators` command.
- If you get stuck on a homework problem, come by office hours or post on Piazza! We recommend you spend about 30 minutes trying to figure out a problem, and then ask for help. We'll be happy to clarify material from class and algorithm concepts, but we will not give out solutions or confirm your answers are correct.
- Finding solutions to homework problems online, or by speaking with students not enrolled in the class, is strictly forbidden.

Problem 1. *Binary Search (5 points)*

Suppose you're searching for $key = 18$ in a binary search tree with 100 nodes with values 1-1000. For each sequence of comparisons below, indicate if it could actually happen.

1. 18 vs 20, 18 vs 250, 18 vs 345, 18 vs 400
2. 18 vs 250, 18 vs 150, 18 vs 149, 18 vs 15
3. 18 vs 10, 18 vs 5, 18 vs 18
4. 18 vs 10, 18 vs 20, 18 vs 15, 18 vs 18

Solution:

Problem 2. *Min-Heaps*

A min-heap is similar to a max-heap, except it maintains the property that the value of every node must be at least as as small as all of its descendents. One of the uses for a min-heap is for a min-priority queue, where you have items entering a queue and you always want to select the lowest-value first.

(a) Draw the Heap represented by this array: $[5, 14, 23, 32, 41, 87, 90, 50, 64, 53]$. Is it a valid min-heap?

Solution:

(b) In order to implement a min-priority queue, we need a function EXTRACT-MIN(A) which removes and returns the smallest value in the min-heap representend by A, and then re-heapifies. Give complete pseudocode for this procedure; you can assume that a procedure MIN-HEAPIFY(A, i) exists for a min-heap.

Solution:

(c) Suppose that you have access to an INSERT procedure, which inserts a key into a min-heap and re-heapifies. Given the following sequence, where a numeric value means INSERT and a star means EXTRACT-MIN, give the sequence of values returned by the EXTRACT-MIN operations.

$534 * 1 * 82441 * * * 3 * * 4 *$

Solution:

(d) What is the minimum number of swaps that happen during EXTRACT-MIN with a heap of size $n > 3$ that has no duplicate keys?

Solution:

(e) Give a heap of size 7 for which the minimum number of swaps happens, and explain what the swaps are.

Solution:

Problem 3. *Stacks and Queues (10 points)*

(a) Show how to implement a stack using two queues. Write the complete pseudocode that would implement the procedures PUSH and POP.

Solution:

(b) What is the bound on the run-time in the best and worst case for your PUSH procedure?

Solution:

(c) What is the bound on the run-time in the best and worst case for your POP procedure?

Solution:

Problem 4. *"Balanced" Binary Trees (10 points)*

Consider a binary tree T, with $|T| = n$ nodes. For a given node x in T, we can say that the subtree rooted at x is "approximately balanced", $AB(x)$, if $|x.right| \leq 2 \cdot |x.left|$ and $|x.left| \leq 2 \cdot |x.right|$.

(a) What is the maximum height of a binary tree T with n nodes if $AB(x)$ holds? (Recall that the height of a one-node tree is 0.)

Solution:

(b) Prove that, if $AB(x)$ holds for every node x in a tree T, then the number of nodes in T, $|T| \geq \frac{3}{2}^{height(T)}$. Use induction on on the height of T for your proof, with the base case being a tree of height 0.

Solution:

Problem 5. *Binary Search Trees (10 points)*

(a) Draw the Binary Search Tree that would result from inserting the following integers in the given order:

   (a) 17
   (b) 14
   (c) 12
   (d) 20
   (e) 33
   (f) 15

   Solution:

(b) In what order would the nodes of the tree print if you ran a post-order traversal (i.e., recursively traverse left and right subtrees before printing out the root)?

   Solution:

(c) Give complete pseudocode for a recursive algorithm to return the node with the minimum value in a binary search tree (e.g., your algorithm would return the node containing 12 in the example above). Your subroutine should be named TREE-MIN and have one parameter x, the root of the current subtree.

   Solution: