Quiz 1

- Do not open this quiz booklet until directed to do so. Read all the instructions on this page.
- When the quiz begins, write your name on the top of every page of this quiz booklet.
- You have 120 minutes to earn a maximum of 120 points. Do not spend too much time on any one problem. Skim them all first, and attack them in the order that allows you to make the most progress.
- **You are allowed one double-sided letter-sized sheet with your own notes**. No calculators, cell phones, or other programmable or communication devices are permitted.
- Write your solutions in the space provided. Pages will be scanned and separated for grading. If you need more space, write "Continued on S1" (or S2, S3, S4) and continue your solution on the referenced scratch page at the end of the exam.
- Do not waste time and paper rederiving facts that we have studied in lecture, recitation, or problem sets. Simply cite them.
- When writing an algorithm, a **clear** description in English will suffice. Pseudo-code is not required. Be sure to argue that your **algorithm is correct**, and analyze the **asymptotic running time of your algorithm**. Even if your algorithm does not meet a requested bound, you **may** receive partial credit for inefficient solutions that are correct.
- **Pay close attention to the instructions for each problem**. Depending on the problem, partial credit may be awarded for incomplete answers.

Problem	Parts	Points
0: Information	2	2
1: Binary Array	1	10
2: Arboreal Attributes	3	15
3: Slice Merge	3	18
4: Nick Sorts	3	18
5: Sum-Chums	1	15
6: Alien Arrival	1	20
7: Local Dynamic Seq.	1	22
Total		120

Name: ______________________________

School Email: ______________________________

Problem 0. [2 points] **Information** (2 parts)

(a) [1 point] Write your name and email address on the cover page.

(b) [1 point] Write your name at the top of each page.

Problem 1. [10 points] **Binary Array**

A **binary array** is an array of integers that contains only 0's and 1's. Given a length-n binary array that is also **sorted increasing**, describe an $O(\log n)$-time algorithm to compute the sum of the array. Remember to argue the correctness and running time of your algorithm.

Problem 2. [15 points] **Arboreal Attributes** (3 parts) In this class, we've defined three properties of binary trees that store keys at their nodes:

- **Max-Heap Property:** for any node `v`, the key stored at `v` is **at least as large** as every key stored in the subtree rooted at `v`.
- **BST Property:** for any node `v`, the key stored at `v` is **at least as large** as every key in the subtree rooted at `v.left`, and **less or equal to** every key in the subtree rooted at `v.right`.
- **AVL Property:** for any node `v`, the **absolute difference** between the heights of the subtrees rooted at `v.left` and `v.right` is at most 1.

An AVL Tree satisfies both the AVL and BST Properties, while a Max-Heap satisfies both the AVL and Max-Heap Properties. Answer the following questions about binary trees and these properties.

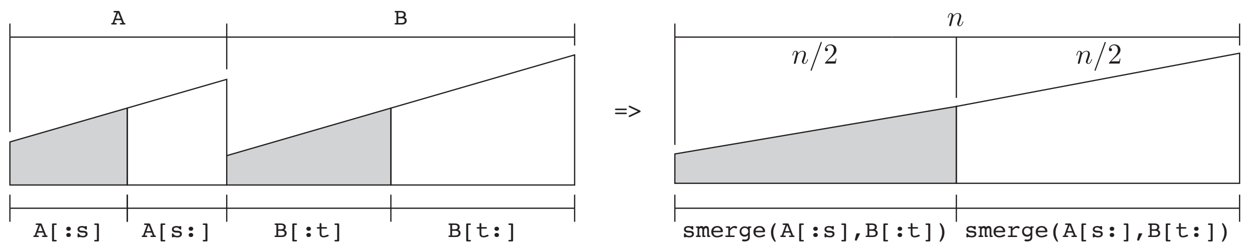
(a) [5 points] Draw a binary tree on the keys $\{1, 2, 3, 4, 5, 6, 7\}$ that satisfies the Max-Heap Property, **but not** the BST or AVL Property (tree should **not** be height balanced).

(b) [5 points] Draw a binary tree on the keys $\{1, 2, 3, 4, 5, 6, 7\}$ that satisfies both the Max-Heap Property **and** the BST Property, **but not** the AVL Property.

(c) [5 points] Draw a binary tree on seven nodes containing any integer keys, which satisfies all three: the Max-Heap Property, the BST Property, **and** the AVL Property.

Problem 3. [18 points] **Slice Merge** (3 parts)

In this problem, we will analyze an alternative to the merge step of merge sort. Suppose `A` and `B` are sorted arrays with possibly **different** lengths, and let $n =$ `len(A)+len(B)`. You may assume n is a power of two and all n items have **distinct** keys. The **slice merge** algorithm, `smerge(A,B)`, merges `A` and `B` into a single sorted array as follows:



(1) Find indices s and t such that $s+t = n/2$ and the prefix subarrays `A[:s]` and `B[:t]` together contain the smallest $n/2$ keys from `A` and `B` combined.

(2) Recursively compute `X = smerge(A[:s], B[:t])` and `Y = smerge(A[s:], B[t:])`, and return their concatenation `X + Y`, a sorted array consisting of all items from `A` and `B`.

For example, if `A = [1,3,4,6,8]` and `B = [2,5,7]`, we find $s = 3$ and $t = 1$ and then recursively compute:

`smerge([1,3,4], [2]) + smerge([6,8], [5,7]) = [1,2,3,4] + [5,6,7,8].`

(a) [10 points] Describe an algorithm to find indices s and t satisfying step (1) in $O(n)$ time, using only $O(1)$ additional space beyond arrays `A` and `B` themselves. Remember to argue the correctness and running time of your algorithm.

(b) [4 points] **Write** and **solve** a recurrence for $T(n)$, the running time of `smerge(A,B)` when `A` and `B` contain a total of n items (please show your work). How does this running time compare to the `merge` step of `merge_sort`?

(c) [4 points] Let `smerge_sort(A)` be a variant of `merge_sort(A)` that uses `smerge` in place of `merge`. **Write** and **solve** a recurrence for the running time of `smerge_sort(A)`.

Problem 4. [18 points] **Nick Sorts** (3 parts)

Solve each of the following sorting problems by choosing an algorithm or data structure that best fits the application, and justify your choice. **Don't forget this! Your justification will be worth more points than your choice.** You may choose any algorithm or data structure we have covered in this class, or you may modify them as necessary to fit the scenario. If you find that multiple solutions could be appropriate for a scenario, identify their pros and cons, and choose the one that best suits the application. State and justify any assumptions you make. "Best" should be evaluated primarily by asymptotic running time, with other factors secondary.

(a) [6 points] At her birthday party, Pelica Angickles receives one gift from each of her n friends. After the party, she searches online to find out how much each friend spent on their gift. The most expensive gift she received was from her cousin, Pommy, who spent $60.06. Pelica decides to rank her friends according to how much they spent. Help Pelica rank her friends.

(b) [6 points] The memory from Neummy Jitron's spaceship computer has malfunctioned, leaving only a **constant amount** of onboard memory uncorrupted and accessible. Luckily, Neummy has an external disk containing all his timestamped travel logs, but in a random order. The computer can **quickly read and compute** on a small amount of data from the disk at a time, but **writing** to the disk is **extremely slow** (i.e. all that matters is the number of external writes to disk). Help Neummy rewrite the disk to store travel logs chronologically.

(c) [6 points] The Bum Chucket is an underwater restaurant, owned by Pleldon Shankton and managed by an underwater computer named Plaren. During the day, Plaren records the name of each customer, and the amount they spent on food. Assume that no customer purchases food more than once per day. Periodically, Pleldon wants to know the names of the k highest spenders so far that day. Help Plaren store information to quickly maintain and generate this list for Pleldon upon request.

Problem 5. [15 points] **Sum-Chums**

Given an array A of n **positive** integers, a pair of array indices (i, j) are **sum-chums** if

$$(A[i] - i) + (A[j] - j) = 0 \qquad \text{or equivalently,} \qquad A[i] + A[j] = i + j.$$

Note that $A[i] + A[j] < 2n$ for any sum-chum pair (i, j), since $i, j \in \{0, \dots, n - 1\}$. Describe a **worst-case** $O(n)$-time algorithm to determine whether an array A contains a sum-chum pair. Maximum 10 points for an expected $O(n)$-time solution. Remember to argue the correctness and running time of your algorithm.

Problem 6. [20 points] **Alien Arrival**

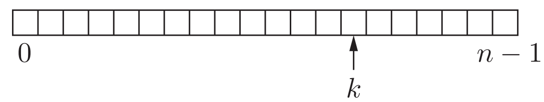
Aliens have landed on earth, but humans are having a difficult time communicating with them. The aliens have allowed linguistics professor Bouise Lanks onto their ship, which houses a library containing $\boldsymbol{B}$ **engraved blocks**, each of which is covered in many symbols. There are $\boldsymbol{S}$ **symbols** in the alien language: each symbol is made up of line segments, and no two symbols have the exact same number of segments. Using carbon dating technology, Dr. Lanks can measure how many earth days have passed since each block was engraved. Help Dr. Lanks design a database to store alien symbol information, supporting the following two operations each in $O(\log B)$ time. Remember to argue the correctness and running time of your operations. State whether the running time of each operation is a worst-case, amortized, and/or expected bound.

1. RECORD-SYMBOL(s, d): record that the symbol having s line segments was engraved once on some block d days ago.
2. COUNT-SYMBOL(s, d_1, d_2): return the number of times the symbol having s line segments was engraved by the aliens during the time between d_1 and $d_2 < d_1$ days ago.

Problem 7. [22 points] **Local Dynamic Sequence**

Design a data structure that maintains a sequence $x_0, x_1, \ldots, x_{n-1}$ of n items, maintains an index k satisfying $0 \le k < n$, and supports the following operations, each in $O(1)$ time. Remember to argue the correctness and running time of your operations. State whether the running time of each operation is a worst-case, amortized, and/or expected bound.

1. AT(i): return item x_i given **any** i where $0 \le i < n$. (Same as in the sequence interface.)
2. INDEX$()$: return the current index k.



3. MOVE-LEFT$()$: decrement the index: $k \leftarrow k - 1$.
4. MOVE-RIGHT$()$: increment the index: $k \leftarrow k + 1$.
5. INSERT-HERE(x): shift-right all items to the right of the index ($x_k \to x_{k+1} \to \cdots \to x_{n-1} \to x_n$), set $x_k = x$, and increment n. (Same as sequence-interface operation INSERT-AT(k, x).)
6. DELETE-HERE$()$: shift-left all items to the right of the index ($x_k \leftarrow x_{k+1} \leftarrow \cdots \leftarrow x_{n-1}$), and decrement n. (Same as sequence-interface operation DELETE-AT(k).)

Maximum 5 points if any operation takes $\Omega(n)$ amortized expected time.

SCRATCH PAPER 1. DO NOT REMOVE FROM THE EXAM.

You can use this paper to write a longer solution if you run out of space, but be sure to write "Continued on S1" on the problem statement's page.

SCRATCH PAPER 2. DO NOT REMOVE FROM THE EXAM.

You can use this paper to write a longer solution if you run out of space, but be sure to write "Continued on S2" on the problem statement's page.

SCRATCH PAPER 3. DO NOT REMOVE FROM THE EXAM.

You can use this paper to write a longer solution if you run out of space, but be sure to write "Continued on S3" on the problem statement's page.

SCRATCH PAPER 4. DO NOT REMOVE FROM THE EXAM.

You can use this paper to write a longer solution if you run out of space, but be sure to write "Continued on S4" on the problem statement's page.