Section 1: General Terminology

1. (4 points) The following are questions about terms used in our course.
   a. What is your instructor's last name?

      frank fiedler

      writer: thomas h cormen

   b. Our course name is abbreviated to ADAA. What does ADAA stand for?

      advanced design and
      analysis of algorithms

   c. We have looked at ADTs and data structures to analyze run-time. What does ADT stand for?

      abstract data type

   d. Which Python IDE are we using for programming and homework submission?

      pycharm

      java- intelliJ

Section 2: Data

2. (4 points) The following are True/False questions about data in Computer Science. Mark either Ⓣ (for True) or Ⓕ (for False).

   Ⓣ Ⓕ An ADT defines the behavior of a data implementation from the point of the ***programmer***.

   Ⓣ Ⓕ A data structure is a representation of the data (organization, storage, and management) from the point of the ***user***.

   Ⓣ Ⓕ A data structure is a collection of data values, the relationships among them, and the functions or operations that can be applied to the data.

   Ⓣ Ⓕ An ADT is implementation-dependent.

3. (4 points) The following are questions about particular ADTs / data structures. Mark either ADT or DS to indicate which one is applicable.

   ADT DS String

   ADT DS List

   ADT DS Integer

   ADT DS Scapegoat Tree

Section 3: Heaps and Trees

4. (4 points) A binary heap is a nearly complete binary tree filled on all levels except possibly the lowest level where leaves are pushed left-most. Heaps are often implemented as an array.
   a. Each node in a heap satisfies the heap property. What is the relationship between a node and its right child in a min-heap?

      node.key<=node.right.key

   b. Which element is the minimum in a min-heap?

      root

   c. If a binary heap contains 28 elements (nodes), what is the height of the corresponding binary tree?

      $\log_2(28)$

   d. How many leaves does a binary heap of size n have?

      n/2

5. (4 points) A 2-3 tree is a tree in which each non-root node which is not a leaf has 2 or 3 sons. The following are True/False questions about 2-3 trees. Mark either Ⓣ (for True) or Ⓕ (for False).
   Ⓣ Each node is labeled with the largest value in the middle subtree and the largest value in the right subtree.
   Ⓕ Every path from the root to a leaf has the same length.
   Ⓕ Data is ordered left-to-right.
   Ⓕ Data is stored only in leaves.

6. (4 points) A binary search tree (BST) is a linked-node based binary tree which stores key-value pairs (or just keys) in each node. Left and right children are roots of left and right subtrees, respectively. The following are True/False questions about BSTs. Mark either Ⓣ (for True) or Ⓕ (for False).
   Ⓕ A BST with n nodes has height at most n.
   Ⓕ Keys in a BST must be comparable.
   Ⓣ Keys in a BST must be integers.
   Ⓣ The minimum key in a BST is in the root.

7. (4 points) A binary search tree (BST) is a linked-node based binary tree which stores key-value pairs (or just keys) in each node. Left and right children are roots of left and right subtrees, respectively. The following are True/False questions about BSTs. Mark either Ⓣ (for True) or Ⓕ (for False).

Ⓣ Ⓕ The size of the left subtree must not differ by more than 1 from the size of the right subtree for any node in a BST.

Ⓣ Ⓕ In-order walks provide the correct key order (smallest to largest) regardless of the tree balance.

Ⓣ Ⓕ Post-order walks provide the reverse key order to a pre-order walk.

Ⓣ Ⓕ Keys and values in a BST must be of the same type.

8. (4 points) A binary search tree (BST) is a linked-node based binary tree which stores key-value pairs (or just keys) in each node. Left and right children are roots of left and right subtrees, respectively.

a. What is the relationship between the key of a parent and the key of its right child?

key of right child greater than key of parent node

b. Give a short description of an algorithm to find the predecessor of a node N (by key) after the node N has been located. No programming on paper!

predessesor of the node is node with largest key smaller than node key
the predesesor of node is either the maximum node in the left subtree
of node or first parent of node for which node is located on the
right parent subtree

Section 4: Self-Balancing Trees and Forests

9. (4 points) A **self-balancing** tree (forest) is a (collection of) search tree data structure(s) in which insert/delete operations may trigger a partial tree rebuild. **Name four** self-balancing search tree data structures we discussed in class or have been assigned as presentation topics.

a. red black trees

b. avl trees

c. 2-3 trees

d. scape goat trees

10. (4 points) Scapegoat trees are search trees which upon insert/delete operations rarely but expensively choose a scapegoat node and completely rebuild the subtree rooted at it into a complete tree. The following are True/False questions about Scapegoat trees. Mark either Ⓣ (for True) or Ⓕ (for False).

Ⓣ Ⓕ Scapegoat trees are binary search trees.

Ⓣ Ⓕ Scapegoat trees store the size of the whole tree in the root node.

Ⓣ Ⓕ Scapegoat trees store the size of the tree since the last rebuild in the root node.

Ⓣ Ⓕ Scapegoat trees store the weight of the subtree rooted at a node N in that node N.

11. (4 points) Scapegoat trees are search trees which upon insert/delete operations rarely but expensively choose a scapegoat node and completely rebuild the subtree rooted at it into a complete tree. The following are True/False questions about Scapegoat trees. Mark either Ⓣ (for True) or Ⓕ (for False).

Ⓣ Ⓕ If T is an α-weight-balanced binary search tree then T is also α-height-balanced.

Ⓣ Ⓕ A measure of tree balance is the parameter α. For a Scapegoat tree, $\frac{1}{2} \leq \alpha \leq 1$.

Ⓣ Ⓕ A measure of tree balance is the parameter α. For a Scapegoat tree, size(left[node]) $\leq$ α-size(node).

Ⓣ Ⓕ If a partial tree rebuild is triggered by insertion of a deep node N, the scapegoat node is a descendant of the node N.

12. (4 points) A priority queue is a special type of queue in which each element is associated with a priority value. Elements are served on the basis of their priority. Higher priority elements are served first. Elements with the same priority are served according to their order in the queue. Priorities can be encoded with keys.

Name two algorithms or applications for which priority queues are used.

a. djikstra's shortest path

b. prims minimum spanning tree

Name two data structures that we looked at for implementation of a priority queue in class.

c. linked list

d. binary heap

13. (4 points) Fibonacci heaps are a collection of trees. The following are True/False questions about Fibonacci heaps. Mark either Ⓣ (for True) or Ⓕ (for False).

Ⓣ Ⓕ The roots of the trees in a Fibonacci heap are stored in a doubly linked list.

Ⓣ Ⓕ Children nodes in a Fibonacci heap are stored in a doubly linked list.

Ⓣ Ⓕ Nodes in a Fibonacci heap have parent pointers.

Ⓣ Ⓕ A node N in a Fibonacci heap has pointers to one of its children.

Section 5: Amortized Analysis and Runtime Analysis

14. (4 points) Run-time analysis is an estimation of running time of an algorithm as a function of its input size (usually denoted as n). The following are four True/False questions about runtime analysis. Mark either Ⓣ or Ⓕ.

Ⓣ Ⓕ FIND/SEARCH/GET in an array with n keys always has runtime of $O(1)$.

Ⓣ Ⓕ FIND/SEARCH/GET in a hash table with chaining and n keys always has runtime of $O(1)$.

Ⓣ Ⓕ FIND/SEARCH/GET in a BST with n nodes and height h always has runtime of $O(h)$.

Ⓣ Ⓕ FIND/SEARCH/GET in a 2-3 tree with n nodes always has runtime of $O(\log(n))$.

15. (4 points) Amortized analysis is a method for analyzing an algorithm's complexity. The following are four True/False questions about amortization analysis. Mark either Ⓣ or Ⓕ.

Ⓣ Ⓕ Amortized analysis evaluates the average cost.

Ⓣ Ⓕ Amortization is used for the evaluation of one particular operation only, such as `push`, `pop`, or `multipop`.

Ⓣ Ⓕ Scapegoat trees achieve $O(\log(n))$ amortized run-rime complexity for all operations INSERT, DELETE, SEARCH.

Ⓣ Ⓕ DELETE_MIN in a Fibonacci heap has amortized runtime $O(\log(n))$.

16. (4 points) Name the **three** types of **amortized analysis** covered in class.

a. potential method

b. agregate method

c. accounting method