CSE 100: Algorithm Design and Analysis
Midterm 2

Spring 2021

- This is a take-home exam with no proctoring. You can start at any time once you get access to this exam. You have to submit your solution by **4:30pm, 16-MAR**, through CatCourses (Midterm 2 under Assignments). You can resubmit any number of times until the deadline. If there are any technical issues with uploading, it is extremely important to immediately contact the instructor or the TA.
- This is an open-book exam. The **only** resources you can use during the exam are all **course materials** uploaded to CatCourses plus the **textbook**. In particular, this means that you are NOT allowed to search for solutions on the internet. No calculator is allowed. You have to take the exam by yourself.
- There are 9 problems in total. You can earn some partial points if you show progress even if you can't solve problems completely.
- Please make your answers concise as well as precise. If there is something in the question that you believe is open to interpretation, then please go ahead and interpret, but state your assumptions in your answer.
- If you have questions, you can email the instructor and the TAs. However, we may not answer your questions on time as we can't be available all the time. In particular, do not expect reply on Monday.

You're required to take the following honor pledge prior to taking the exam.

By completing this exam, I acknowledge and confirm that I will not give or receive any unauthorized assistance on this examination. I will conductmyself within the guidelines of the university academic integrity guidelines.

1. (10 points) For each of the following claims, determine if it is true or false. *No* explanation is needed.

   (a) (2 points) The running time of (Deterministic) Quicksort is $O(n^2)$.

   (b) (2 points) Suppose we resolve hash table collisions using chaining. Under the simple uniform hashing assumption, an unsuccessful search takes $\Theta(1+\alpha)$ time in expectation where α is the load factor of the current hash table.

   (c) (2 points) The decision tree of *any* comparison based sorting algorithm has a height $O(n \log n)$.

   (d) (2 points) One can sort n integers in the range between 0 and n^{10} in $O(n)$ time.

   (e) (2 points) The running time of Randomized-Quicksort that picks the pivot uniformly at random is always $O(n \log n)$ for any input of size n.

2. (12 points) Potpurri

   (a) (3 points) Consider a hash table with chaining using a hash function that is a good approximation to the simple uniform hashing assumption, and having a load factor α (= the total number of elements in the table n divided by the number of slots m). Give the expected time of an unsuccessful search (i.e., to search for a key that is not in the table); explain your answer (intuitively at least).

   (b) (3 points) Consider Randomized-Quicksort that picks the pivot uniformly at random. Is there some particular input that will *always* cause the worst-case runtime of $\Theta(n^2)$?

   (c) (6 points) Show a formal proof that the height of a Red-Black Tree h with n nodes, is at most $2.log(n+1)$, i.e. $h \leq 2.log(n+1)$. Show and proof any intermediate claim in order to prove the final claim.

3. (6 points) For each of the two binary search trees shown in Fig. 1 and 2 (NILs not shown), is it possible to color it as valid red-black tree If so, provide a red-black tree coloring. If not, write "impossible". In either case, no justification is needed. If you don't have red and black ink, write "r" to color a node red, or "b" to color it black. Each part is worth 3 points.

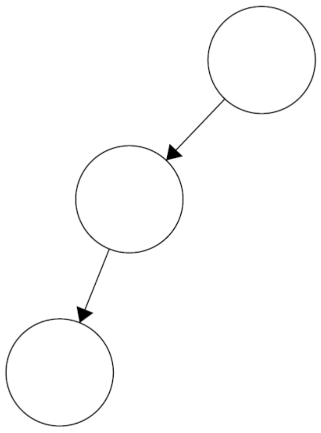


Figure 1: First Tree

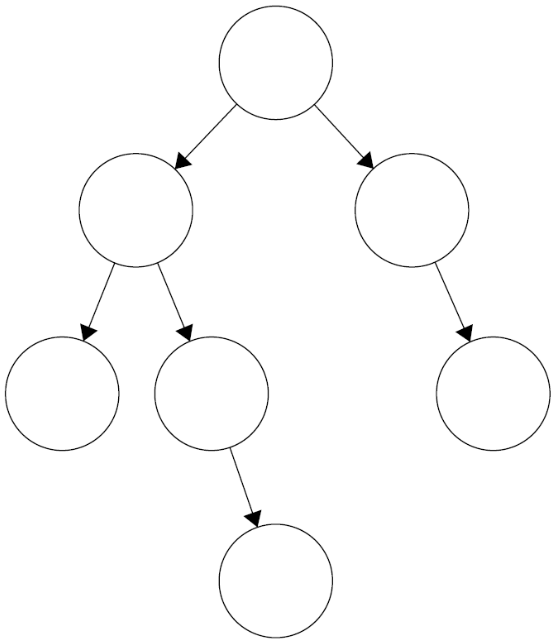
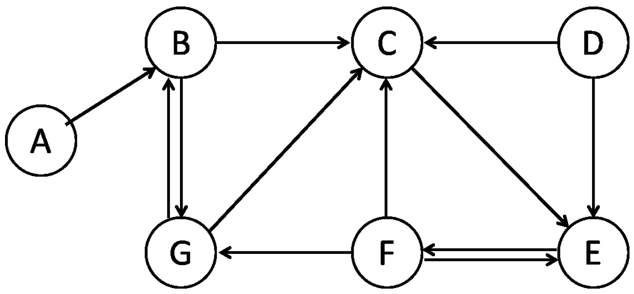


Figure 2: Second Tree

4. (35 points) Graphs. Consider the following directed graph.



(a) (3 points) Draw the adjacency-list representation of G, with each list sorted in increasing alphabetical order.

(b) (3 points) Give the adjacency matrix of G.

(c) (7 points) Do depth-first search in G, considering vertices in increasing alphabetical order. Show the final result, with vertices labeled with their starting and finishing times.

(d) (15 points) Based on your results, proceed to run the SCC algorithm find the strongly connected components of G (show the result of the DFS with vertices labeled with their starting and finishing times).

(e) (7 points) Run BFS with B as starting vertex. Show the tree edges produced by BFS along with $v.d$ of each vertex v. More precisley, run BFS with B as starting point assuming that each adjacency list is sorted in increasing alphabetical order.

5. (5 points) Consider a hash table with $m = 10$ slots and using the hash function $h(k) = k \mod 10$. Say we insert (elements of) keys $k = 3, 5, 13, 23, 15, 29$ in this order. Show the final table when chaining is used to resolve collisions. Insert the element at the *beginning* of the linked list.

6. (8 points)

(a) (4 points) Consider a hash table with chaining using a hash function that is a good approximation to the simple uniform hashing assumption, and having a load factor α, but instead of using a linked list to handle collisions, we use a binary search tree. Give the expected time to search for a key and explain it.

(b) (4 points) Based on that, would using a BST instead of a linked list (in each slot) be better forsearching? Consider load factors $\alpha = 0.5$ and $\alpha = 128$. Reason your answer carefully.

7. (4 points) Quicksort implementation. Give a *pseudo-code* of Quicksort(A, p, r) that sorts $A[p...r]$ via quick-sort. You can assume that all elements in A have distinct values. You can use the following helper function, Partition(A, p, r):

```
Partition(A, p, r)
1. x = A[r]
2. i = p-1
3. for j = p to r-1
4.    if A[j] <= x
5.       i = i+1
6.       exchange A[i] with A[j]
7. exchange A[i+1] with A[r]
8. return i+1
```

8. (8 points) You are given two sequences, $\langle a_1, a_2, ..., a_n \rangle$ and $\langle b_1, b_2, ..., b_n \rangle$, where each sequence consists of distinct integers. Describe a linear time algorithm (in the average case) that tests if a sequence is a permutation of the other. Assume that the simple uniform hashing assumption holds. Explain the running time of your algorithm.

9. (12 points) Consider again the previous problem but now allowing both the input sequences to have repeated integers. Modify your algorithm so that it also works in this case.