Exam 2 CS-GY 6033 Fall 2023
November 18 2023

Instructions:

Scheduling:

- The exam runs from 6pm to 9pm EST on Nov 18th 2023. Your exam is run through Gradescope, and will available to download at 5:50pm. The exam is to be completed by 9:00pm. Gradescope stops accepting uploads at 9:15pm. There are absolutely no late uploads accepted by the system after that time. The time to complete the exam will depend on your preparation: a prepared student could finish it in less than two hours, an ill-prepared student might take up to three hours. It is your responsibility to allow time for uploading your exam.

Format:

- The exam consists of a total of 80 points. The remaining 20 points are for the online quiz.
- **Submission Penalty: if you do not submit your exam on Gradescope, or do not properly assign your pages, you receive a deduction of 3 points on your exam.**
- You may write your solutions directly on the downloaded exam paper, *or* in your own format. You are responsible for providing clear and legible solutions to the problems. Your exam must be resubmitted into Gradescope electronically. Ensure that you know how to quickly upload any handwritten material. This is entirely the student's responsibility. You may assign your pages after the deadline.

Questions during the exam:

- There is a ZOOM session for questions that will be open during the entire course of the exam (microphones OFF). You may ask questions with private chat during the exam. Any announcements made by the instructor during the exam will be made over ZOOM and also be email. **It is the student's responsibility to stay connected (either by ZOOM or email) during the exam.**

Rules:

- This exam is a **take-home exam**. You may use **only** the resources from the online class (any material on NYU classes for this course) and any type of calculator (although it is not needed).
- Your work must be entirely your own. It is **forbidden to discuss any work with *any* other person**. Furthermore, your work must be done without using internet searches (although this is completely unhelpful for this exam). Any breach of academic honesty will be handled in accordance with the *Student Code of Conduct*, (a copy of which is provided), and in this particular case, taken very seriously.
- You are asked to **read** the attached Student Code of Conduct Section III subsections A,B,C,D,E and **sign** below to acknowledge that you aware of the policy. Once signed, a copy of this page must be uploaded with your exam.

I acknowledge that my submitted Exam work is entirely my own. I have read and am in accordance with the Student Code of Conduct policy of NYU Tandon and fully accept the consequences of breaching the above instructions.

Name: _____

Signature: _____

Instructions

Read the following questions carefully. Recall that you may use algorithms from class. For example, you do not need to re-describe from scratch algorithms like inserting into a BST, Inorder Traversal, or using the Rank Algorithm.

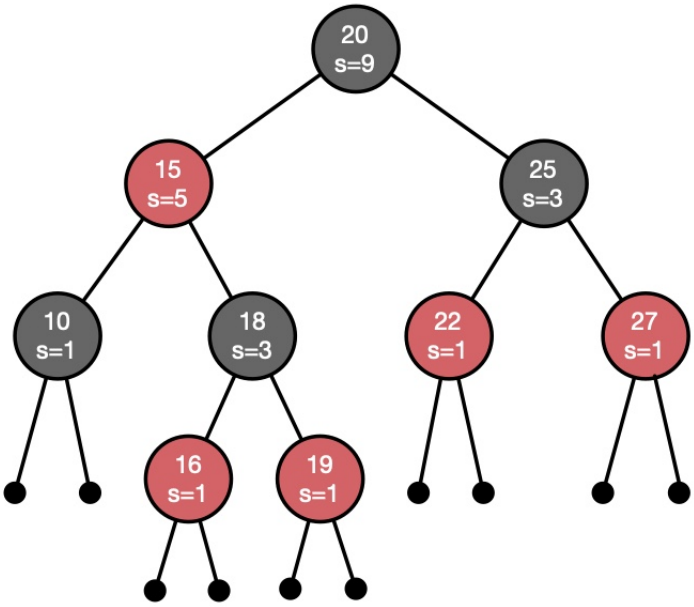
Your written exam has a total of 80 points.

BE SURE TO JOIN THE ZOOM SESSION DURING THE EXAM!!
MICROPHONES OFF!

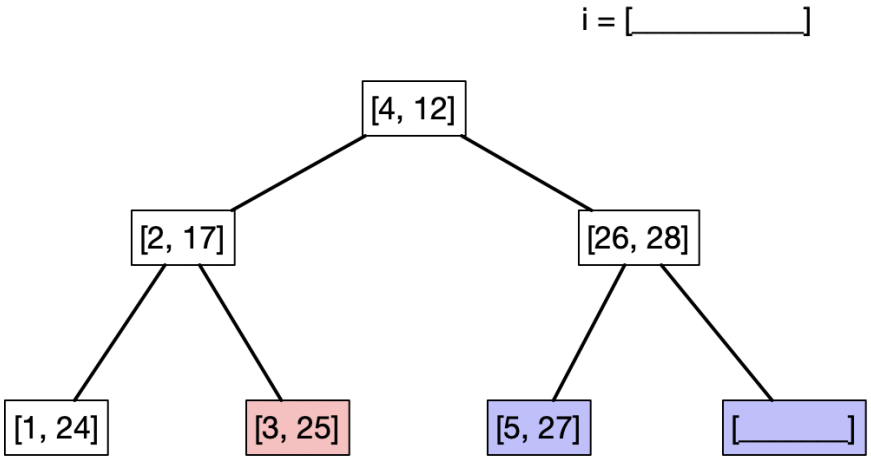
Question 1

(a) 6 points

The drawing below represents a valid red-black tree, augmented with subtree sizes. Show how to insert the new node 17. You must show both the initial insertion, and any changes made by RB-repair. You must also show the updates to any attributes of the tree nodes. Explain briefly why the black height of the tree does not change after this insert.



(b) 4 points An interval tree is partially drawn below. Your job is to fill in the intervals of the interval tree and give an example of an interval i, such that the following occurs : when INTERVAL-SEARCH(i) is executed with your interval i, the pink node is returned. Furthermore, your interval i must intersect with both blue nodes, as well as the pink node.



(c) 3 points For each of the following statements, decide if the statement is true for false, and briefly (one sentence!) justify your answer.

1. Does a red-black tree exist on 100 nodes with black height 4?
2. Does a red-black tree exist that is NOT also an AVL tree?
3. Can a single insertion into a red-black tree ever cause the black height to increase by 2?

Question 2

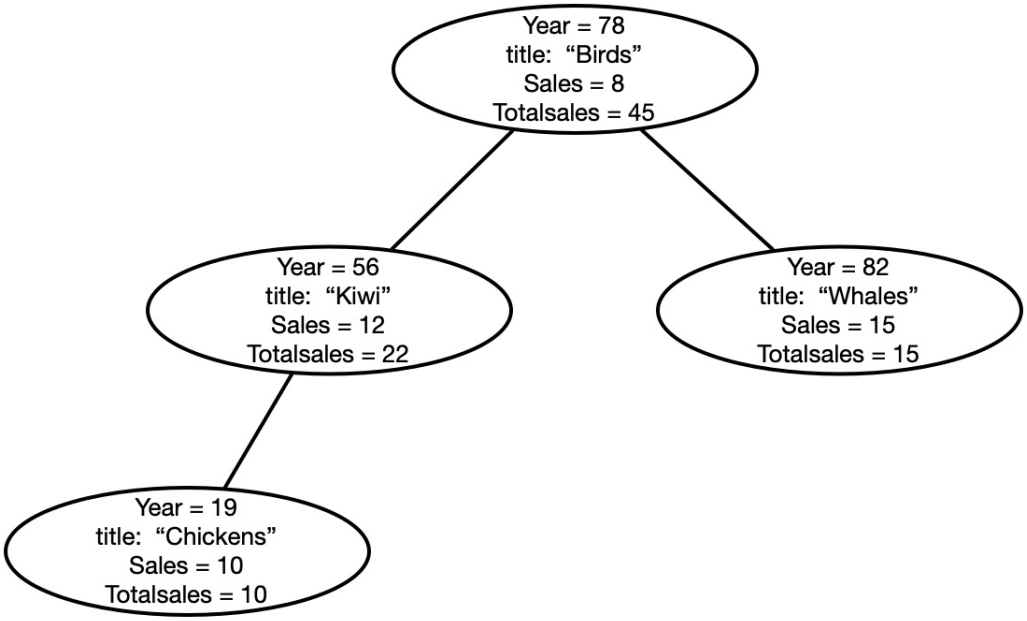
8 points Let T be a valid red-black tree. Write the pseudo-code for a recursive algorithm PrintBlack-Height(T,k) which prints out the keys of all nodes in the tree that have a black height of exactly k. Recall that only the nil (dummy) nodes of the tree have a black-height of 0. You may assume that the initial call is with $k \geq 1$ and the initial T is not an empty tree. You may also use $x.isNilNode$ in your pseudo-code, which is an attribute that is either True/False, depending on whether x is a nil node. Justify the runtime of $O(n)$.

Question 3

8 points

Let T be a binary search tree that stores information on published books. Each tree node x contains the following attributes: **x.title, x.year, x.sales and x.totalsales**. The attribute **x.year** is used as the **key** of the BST. In other words, the tree is organized by the release year of the books, where years are measured since 1900. The attribute **x.sales** is the value in dollars of the sales of book x, and the attribute **x.totalsales** is the **total of all book sales in the subtree rooted at** x An example of such a BST is shown below.

Your Job: Write the pseudo-code for a recursive algorithm called SalesByYear(T,k) which returns the total sales from all books that are released on or before year k. You must justify the runtime of $O(h)$ of your algorithm, where h is the height of your tree.



Question 4

6 points Let T_1 be a reference to the root node of a BST. Let T_2 be a reference to another BST. Suppose the keys of each tree are distinct from each other, and that each tree has height h (the same height). Design an algorithm that will join the two trees together, into one tree. Your algorithm must run in time $O(h)$, and you must justify why your procedure runs in this time. You do not need to provide the pseudo-code, but you must carefully outline your steps.

Question 5

(a) 6 points Let T be the root node of a Binary Search tree. We wish to augment each node x of the binary search with an attribute $x.depth$, which represents the depth of the node x in the tree T. Write the pseudo-code for an algorithm called SetDepth(T) that sets of the value of $x.depth$ for each node in the tree.

(b) 3 points Suppose that T is a red-black tree that is updated with the above attribute $x.depth$. Explain whether or not you can carry out an insert into a red-black tree in such a way that the $x.depth$ attributes are updated in $O(\log n)$ time.

Question 6

12 points

A project manager would like to store a set of n projects. Each project consists of a **time interval, (start and end time)** , a project **title** and a **project profit** (in dollars). For simplicity, you may assume that all start/end times are distinct.

An interval tree is used to implement this data set. In addition to the usual attributes required for interval trees, each node is augmented with **x.totalprofit** which represents the total profit of all nodes in the subtree rooted at x.

The manager would like a data structure that organizes the project objects in such a way that she can carry out the following operations in the specified runtimes:

The operations:

1. Insert a new project (including interval, title, and profit) *Time:* $O(\log n)$
2. Output the total profit out of all projects. *Time:* $O(1)$
3. Output the title of the project that is the **last to start**. *Time:* $O(\log n)$
4. Given a time t, print out the project title of the project that is the *next to start* after time t. *Time:* $O(\log n)$
5. Output the total profit of all projects that start **before time** t. *Time:* $O(\log n)$
6. Print the titles of all projects that start **after time** t. Do not simply traverse the entire tree.

Your job:

- Completely describe the attributes used in the interval tree, and how the tree is implemented.
- You must justify the runtime to build the structure in time $O(n \log n)$.
- You must justify specifically how to carry out the operations above in the specified time (either with pseudo-code or referencing algorithms from class)

Question 7

12 points

Recall the bridge problem from class, where m cities on the eastern side of a river, and n cities on the western side of a river. Bridges can be constructed across the river, as long as the bridge connects cities of the same altitude, and bridges are not allowed to cross one another. Let $E[1..m]$ be the altitudes of the eastern cities, and $W[1..n]$ be the altitudes of the western cities, each in meters. In this scenario, we will assume that bridges carry a varying amount of traffic, depending on which cities they connect. Let $T[i, j]$ be an array that represents the number of cars per hour that would be carried by a bridge connecting cities i and j, for $i = 1..m$ and $j = 1..n$.

Your Job: Update the relevant DP problem from class so that it returns the **maximum traffic** carried by building a set of bridges between the eastern and western cities. The input to the problem is $m, n, E[], W[]$ and $T[][]$.

You must properly define the DP table, explain the initialization, justify how you fill in the entries, provide the pseudo-code, clearly show which value is returned, and justify the runtime.

Question 8

12 points

I am actually a professional gymnast. At the next championships, there are n gymnastics events, which run over a period of n days (exactly 1 per day). Each event has a specific monetary prize, stored in array $p[1, \ldots, n]$, where $p[i]$ is the prize money for event i. I am hoping to win as much money as possible. The problem is, that I need to account for the time it takes to recover after an event. So I cannot simply participate in each event :(. My coach has provided me with an estimate of how long she thinks it will take me to recover, from each of the events. This is stored in array $r[1, \ldots, n]$, where $r[i]$ is the recovery time in days after participating in event on day i. Note that if the recovery time is 3 days for the event on day 7, then I could compete next in event on day $10, 11, 12, \ldots$etc. An example input is given below. In this case, the best choice is to participate on day $2, 4, 5$, giving a profit of \$120.

Day:	1	2	3	4	5	6
Prize $:	20	40	10	60	20	15
Recovery time :	3	2	4	1	2	1

Your Job:
Provide a DP solution that returns the maximum amount of money I could win by participating in events in such a way that I respect the necessary recovery times. You must provide:

- A properly defined DP table
- A clear explanation of the initialization of the table
- A justification of how you fill in entires of the table
- The pseudo-code for your solution, which clearly shows which value you are returning
- A justification of the runtime.

***Note! This problem is a direct variation of a problem from class! It may help to define a table where $T[i]$ represents the most money you can make up to and including event i.*