* Each group should choose two exercises in 5 given ones.
* Exercise 1(done) - Design a binary search tree (BST) to store a collection of student records. Each record should contain a student's name (a string) and their GPA (a float). Create functions to insert new records, search for a student by name, and find the student with the highest GPA.
* This exercise simulates a scenario where a university wants to maintain a record of its students and find the top-performing student easily.
* Exercise 2(done) - Implement a binary expression tree to evaluate mathematical expressions. Given a mathematical expression in postfix notation (e.g., "5 3 + 4 \*"), construct a binary expression tree and write a function to evaluate the expression.  
  This exercise reflects the way calculators evaluate mathematical expressions and demonstrates the concept of expression trees.
* Exercise 3 (done)- Create a max-heap to manage a to-do list. Each task in the list has a priority (an integer) and a description (a string). Implement functions to add tasks with priorities, remove the highest-priority task, and display the tasks in order of priority.  
  This exercise models a task management system where tasks are assigned priorities, and the highest-priority tasks are completed first.
* Exercise 4 (done)- Implement a max-heap to keep track of the top K scores in a video game leaderboard. Write functions to add new scores and retrieve the K highest scores.  
  In online gaming, leaderboards often display the top players or scores. This exercise mirrors the leaderboard functionality.
* Exercise 5 - Build a priority queue using a max-heap to simulate an emergency room's patient triage system. Each patient has a priority (1-10), and the system should treat patients in order of priority.  
  Hospitals use priority queues to determine the order in which patients receive medical attention based on the severity of their conditions.