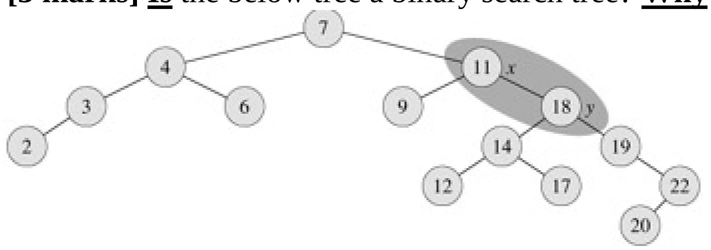
Part II: *Answer as much as you can. You can attempt parts of a question. Use pseudo-code or code to describe required algorithms. Maximum grade = **50 pts.***

Question 1 [**10 Marks**] **Advanced data structures.**

a. **[3 marks] Is** the below tree a binary search tree? **Why**?



b. **[3 marks] Draw** the above tree after doing a *left rotation* on node *x* (whose value is 11).

c. **[4 marks]** We can sort a given set of *n* numbers by first building a binary search tree containing these numbers inserting the numbers one by one and then printing the numbers by an in-order tree walk. **What** are the **worst-case** and **best-case** running times for this sorting algorithm?

Question 2 [**20 Marks**] Assume that there are two types of professional wrestlers مصارعين محترفين: "good guys" and "bad guys." Between any pair of professional wrestlers, there may or may not be a rivalry مواجهة. Suppose we have *n* professional wrestlers and we have a list of *r* pairs of wrestlers for which there are rivalries. Answer the following questions.

a. **[4 marks]** If we model the above problem using a graph *G*, **what** would be the vertices and what would be the edges of G? **How many** vertices and **how many** edges are there?

b. **[5 marks] Write** the **breadth-first** graph search algorithm. **What** is its running time?

c. **[4 marks] Define** a bipartite graph and **draw** an example of one. **Is** the length of *any* cycle in a bipartite graph odd or even? **Why**?

d. **[3 marks] Modify** the breadth-first search algorithm to determine if a graph is bipartite or not.

e. **[4 marks] Give** an $O(n+r)$-time algorithm that (1) determines **whether** it is possible to designate some of the wrestlers as good guys and the remainder as bad guys such that each rivalry is between a good guy and a bad guy and (2) **produces** such designation if found possible.

Question 3 [**27 Marks**] In the bin packing problem, you are given a set of *n* objects, where the size s_i of the *i*th object satisfies $0 < s_i < 1$. We wish to pack all the objects into the minimum number of **unit-size** bins. Each bin can hold any subset of the objects whose total size does not exceed 1.

a. **[3 marks] Define** NP-complete problems.

b. **[3 marks] Give** an example of the input and output of the bin-packing problem.

c. **[2 mark] Choose:** The bin-packing problem is a(n) _________ problem (optimization - decision).

d. **[3 marks] Formulate** a related decision problem for the bin-packing problem.

e. **[5 marks] Sketch** how to reduce to the decision problem you defined in part (d) from the subset sum problem. An instance of the subset-sum problem is a pair (S, t), where S is a set $\{x_1, x_2, ..., x_n\}$ of positive integers and t is a positive integer. This decision problem asks whether there exists a subset of S that adds up exactly to the target value t. **Why** would you do such reduction?

f. **[5 marks]** Suppose that you are given a "black-box" subroutine to solve the decision problem you defined in part (d). **Give** an algorithm to solve the bin-packing problem using the black box. **What** is the running time of your algorithm, where each query to the black box is counted as a single step.

g. **[6 marks]** The first-fit heuristic takes each object in turn and places it into the first bin that can accommodate it. **Give** a counter example in which the first-fit heuristic does not produce an optimal solution. **Give** an efficient implementation of the first-fit heuristic, and **analyze** its running time. **Show** that the first-fit heuristic has an approximation ratio of 2.