**Data Structures and Algorithms, Spring 2020**

**Deadline: June 30, 2020, 12:00pm (noon)**

**Group Policy: No groups allowed – to be submitted individually**

**PROBLEM STATEMENET:**

**Implement the following tree:**

* Red-Black Tree

**Your project should have the following main menu:**

Press 1 to insert values in the tree

Press 2 to delete a value from the tree

Press 3 for searching a value from the tree

Press 4 for pre-order traversal NLR

Press 5 for in-order traversal LNR

Press 6 for post-order traversal LRN

Press 7 for pre-order traversal 2 NRL

Press 8 for in-order traversal 2 RNL

Press 9 for post-order traversal 2 RLN

Press 10 to destroy the tree (all nodes must be deleted)

Press 11 to delete all values in the tree greater than X

Press 12 for displaying parent of a node present in Tree

Press 13 to read integer values from the file “input.txt”

to create a red-black tree

Press 14 to delete all duplicate values from the tree

Press 15 to create copy of the tree

Press 16 to destroy the complete tree

Press 17 to EXIT

**The program should exit when option 17 from the main menu is selected. Your program should also have a copy constructor and a destructor. There shouldn’t be memory leakages or dangling pointers in your program.**

**Please note that in case of red-black tree, the colour of a particular node should also be displayed along with its value for options 3, 4, 5, 6, 7, 8 and 9**

**The non-empty “input.txt” will have the data in such a way that a new value will be placed on every new line. For example, the following file (containing 7 values) is valid for creating the red-black tree (there may be less or more than 7 values):**

10

16

2

-5

0

22

1024