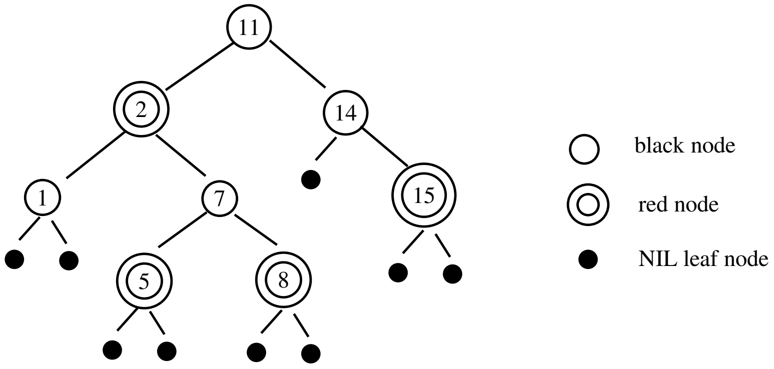
CSCD320 Homework3, Eastern Washington University, Spokane Washington.

Please follow these rules carefully:

1. Verbal discussions with classmates are encouraged, but each student must independently write his/her own solutions, without referring to anybody else's solution.
2. The deadline is sharp. Late submissions will **NOT** be accepted (it is set on the Blackboard system). Send in whatever you have had by the deadline.
3. Submission must be computer typeset in the **PDF** format and sent to the Blackboard system. I encourage you all to use the LaTeX system for the typesetting, as what I am doing for this homework as well as the class slides. LaTeX is a free software used by publishers for professional typesetting and are also used by nearly all the computer science and math professionals for paper writing.
4. Your submission PDF file must be named as: **firstname_lastname_EWUID_cscd320_hw3.pdf**
   (1) We use the underline '_' not the dash '-'.
   (2) All letters are in the lower case including your name and the filename's extend.
   (3) If you have middle name(s), you don't have to put them into the submission's filename.

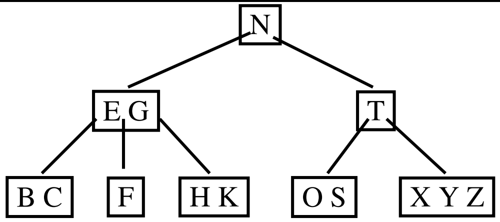
Problem 1 (20 points). *Show the trace of the construction of the Red-Black tree for the sequence* $21, 19, 17, 12, 15, 9$. *That is, you need to draw the state of the tree after inserting each number.*

Problem 2 (20 points). *Show the trace of deleting the nodes from the Red-Black tree below in the order of* $2, 5, 1, 14, 11, 15, 7, 8$. *That is, show the state of the tree after deleting each node.*



Problem 3 (20 points). *Trace the ONE-PASS construction of the B-tree for the sequence* $\{S, G, W, H, O, U, M, A, C, X, P\}$. *Draw the configuration of the B-tree after inserting each letter. We use* $t = 2$ *as the branching degree threshold of the B-tree, so that: (1) all the non-leaf node must have at least* $t - 1 = 1$ *key and at most* $2t - 1 = 3$ *keys; and (2) The root node of an non-empty B-tree must have at least one key and at most* $2t - 1 = 3$ *keys.*

Problem 4 (20 points). *Trace the deletion the sequence of keys* $\{T, G, F, O\}$ *from the B-tree below. Draw the configuration of the B-tree after each deletion. We use* $t = 2$ *as the branching degree threshold of the B-tree, so that: (1) all the non-leaf node must have at least* $t - 1 = 1$ *key and at most* $2t - 1 = 3$ *keys; and (2) The root node of an non-empty B-tree must have at least one key and at most* $2t - 1 = 3$ *keys.*



Problem 5 (20 points)**.** *We know binary search trees support the operations of finding (1) the minimum and maximum node of a given subtree; and (2) the successor and predecessor of a given node in the tree. Now you are asked to present the algorithmic idea and also the psuedocode for the following operations on a B-tree. You can assume all the keys in the B-tree are distinct. Don't forget the DISK_READ operations.*

- `B_tree_min(root)`
  - *input:* `root`.
    "`root`*" is the reference to the root of the B-tree.*
  - *output:* `(node, i)` *or* `NULL`.
    If the tree is not empty, then the "`i`*"th key in the node "*`node`*" is the minimum key;*
    Otherwise return `NULL`.
- `B_tree_successor(node, i)`
  - *input:* `node`, `i`
    A node referenced by "`node`*" and its "*`i`*"th key.*
  - *output:* `(successor_node, j)` *or* `NULL`.
    If the successor of the "`i`*"th key of "*`node`*" exists, return the node referenced by "*`successor_node`*" whose "*`j`*"th key is the successor of the "*`i`*"th key of "*`node`*";*
    Otherwise, return `NULL`.
  - *Notes: (1) You can use the* `B_tree_min` *as a subroutine for* `B_tree_successor` *if needed; (2) You can assume that each node has a parent link that points to the node's parent, and the parent link at the B-tree's root points to NULL; (3) you can assume the given "*`node`*" and "*`i`*" are valid, meaning that they exist in the tree.*

`B_tree_max` *and* `B_tree_predecessor` *are asymmetry to* `B_tree_min` *and* `B_tree_successor` *respectively, so you don't have to work on them.*