CSCD320 Homework2, Eastern Washington University, Spokane, Washington.

Please follow these rules strictly:

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 Canvas system). Send in whatever you have by the deadline.
3. Submission must be computer typeset in the **PDF** format and sent to the Canvas 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 used by almost all computer science and math professionals for paper writing. But of course you are also allowed to use other software for editing (for ex., Microsoft Office Word) but save and submit the file as a PDF.
4. Your submission PDF file must be named as: **firstname_lastname_EWUID_cscd320_hw2.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.
5. Sharing any content of this homework and its keys in any way with anyone who is not in this class of this quarter is NOT permitted.

Problem 1 (15 points)**.** *Use the recursion tree to solve the recurrence:* $T(n) = 4T(n/2) + n^2$*. You need to show the changes of the recursion tree step by step for at least three steps as what we did in the lecture slides (three pictures), and then show how you get the result by observing the recursion tree.*

Problem 2 (15 points)**.** *Use the inductive proof technique to show that* $T(n) = 5T(n/4) + n^2 = O(n^2)$*. That is, you need to find two positive constant* n_0 *and* c*, such that when* $n \geq n_0$*,* $T(n) \leq cn^2$*.*

Problem 3 (15 points; 3 points each)**.** *Solve the following recurrences using the* Master Theorem.

1. $T(n) = 25T(n/5) + n^2 + \log n$
2. $T(n) = 25T(n/5) + 2n^3 + n \log n$
3. $T(n) = 25T(n/5) + 3n^4 - 3n^2$
4. $T(n) = 125T(n/5) + 4n^2 + 5n \log n$
5. $T(n) = 125T(n/5) + 5n^3 + 2n^2$

Problem 4 (35 points)**.** *Suppose you are given a BST by having the access to its root node. Every node of the BST has a left child link and right child link, but does NOT have a parent link. All the keys in the BST are distinct. Dr. Trouble wants you to convert this BST into a double linked list in a* **time-efficient** *manner, so that the nodes in the doubled linked list are in ascending order. Dr. Trouble also wants you to do it in a* **space-efficient** *manner. In particular, he does NOT allow you to allocate new space for the linked list, instead you are asked to recycle/reuse the existing tree nodes. In other words, you have to re-link all the tree nodes using the existing left/right child pointers to build up the double linked list.*

Present your key idea and the algorithm's structure. You may not have to write every detail of the code in a real-world programming language, but must be precise and clear about the main algorithmic idea, and reason why your algorithm works. Give the pseudocode of your algorithm and its time complexity of your algorithm using the big-oh notation and make the bound as tight as possible.

Hint1: Don't waste your time on Google searching. This problem does not exist on the Internet.

Hint2: Build your algorithm using the divide-conquer idea as it is used in the textbook's tree traversal algorithms.

Problem 5 (20 points)**.** *Conduct an independent study/research on the comparison between the hash tables and binary search trees in different scenarios. Do your research using whatever resource and discuss with your peers on this topic. Present whatever you find in your own language, and provide the source (url, book/article's title/author/year) of the ideas/opinions that you presented if they are not your owns.*