Data Structures and Algorithms 2, Course Project 2021

Important – Read before starting

- The deadline for completing and submitting your assignment is strictly Friday 4th June 2021 at 18:00.
- VLE may be set up to not accept late submissions meaning that you will get zero marks if your submission if late. Please plan ahead (it is recommended that you upload and verify your work a day before).
- You must complete the project completion form (shown later) and include it in your report. Submissions without the statement of completion will not be considered.
- You must complete a plagiarism declaration form and include with your submission. Submissions without the form will not be considered.
- Projects must be submitted using VLE only. Physical copies or projects (including parts of) sent by email will not be considered.
- For your convenience, a draft and final submission area will be set up in VLE. Only projects submitted in the *final* submission area will be graded. Projects submitted to the draft area will not be considered.
- It is suggested that after submitting your project, you redownload it and check it again. It is your responsibility to ensure that your upload is complete, valid, and not corrupted. You can reupload the assignment as many times as you wish within the deadline.
- Your project must be submitted in ZIP format without passwords or encryption. Project submitted in any other archiving format will not be considered.
- The total size of your ZIP file must fit in VLE's file upload limit.
- Your submission should include your report in PDF format, your source code, executable file(s), and other relevant attachments.
- Do not include all your source code in the report (small snippets for discussion are allowed).
- It is expected that you submit a quality report with a proper introduction, discussion, evaluation of your work, and conclusions. Also, make sure you properly cite other people's work that you include in yours (e.g. diagrams, algorithms, etc…).
- In general, I am not concerned with which programming language you use to implement this project. However, unless you develop your artifact in BASIC, C, C++, Objective C, Swift, Go, Pascal, Java, C#, Matlab, or Python, please consult with me to make sure that I can correct it properly.
- Please provide clear instructions about how to run your program.
- This is not a group project.
- Plagiarism will not be tolerated.

AVL Trees and Red-Black Trees

- Write a command line program with the following behaviour.
- Let n be a random number between 1000 and 3000.
- Create a set X containing n integers. Each integer must be a random number in the range -3000 and $+3000$. Make sure that there are no duplicates in this set.

```
Set X contains ??? elements
```

- Let m be a random number between 500 and 1000.
- Create a second set Y containing m integers. Each integer must be a random number in the range -3000 and $+3000$. Make sure that there are no duplicates in this set.

```
Set Y contains ??? elements
```

- Note that the sets X and Y may have elements in common.

```
Sets X and Y have ??? elements in common
```

- Insert all the elements in the set X into an AVL tree and into a Red-Black tree. Both are binary search trees. **The AVL tree and Red-Black tree implementations must be your own.**
- After inserting all the values into each tree, you must show: the number of rotations performed in total, the height of the tree, the number of nodes in the tree, and the number of comparison operations (left/right decisions) made in total.

```
AVL: ?? tot. rotations req., height is ??, #nodes is ??, #comparisons is ??
RBT: ?? tot. rotations req., height is ??, #nodes is ??, #comparisons is ??
```

- Delete all the elements in the set Y from the AVL tree and from the Red-Black Tree. After deleting all the values into each tree, you must show: the number of rotations performed in total, the height of the tree, the number of nodes in the tree, and the number of comparison operations (left/right decisions) made in total.

```
AVL: ?? tot. rotations req., height is ??, #nodes is ??, #comparisons is ??
RBT: ?? tot. rotations req., height is ??, #nodes is ??, #comparisons is ??
```

- Let k be a random number between 1000 and 2000.
- Create a third set Z containing k integers. Each integer must be a random number in the range -3000 and $+3000$. Make sure that there are no duplicates in this set.
- Search for every element in the set Z in both the AVL tree and the Red-Black tree. Note that a search may be successful or not.

```
                k is ???
AVL: ??? total comparisons required
RBT: ??? total comparisons required
```

- In your report, write a high-quality discussion comparing AVL trees to Red-Black trees. In this section, make sure to discuss the suitability of each method in real-world applications.

Statement of completion – MUST be included in your report

Item	Completed (Yes/No/Partial)
AVL tree (insert, delete, search)	
Red-Black tree (insert, delete, search)	
Discussion comparing AVL to RBT	
If partial, explain what has been done	

Marking Breakdown

Description	Marks allocated
AVL tree	30%
Red-Black tree	30%
Discussion comparing AVL to RBT	30%
Overall report quality	10%