COMP9024 25T1   # Week 9 Practical   Data Structures and Algorithms
                                   ## Search Tree Algorithms

1. (Rebalancing)

   Complete the Binary Search Tree ADT (BST.h, BST.c, also needs queue.h, queue.c) from the lecture by an implementation of the function:

   ```
   Tree rebalance(Tree t) { ... }
   ```

   We have created a script that can automatically test your program. To run this test you can execute the dryrun program that corresponds to this exercise. It expects to find the file named BST.c in the current directory with your implementation of the function rebalance().

   You can use dryrun as follows:

   ```
   prompt$ 9024 dryrun BST
   ```

2. (Red-black trees)

   Implement the pseudocode for Red-Black Tree Insertion from the lecture (slides 55–59) in our Red-Black Tree ADT (RBTree.h, RBTree.c) as the function:

   ```
   Tree TreeInsert(Tree t, Item it) { ... }
   ```

   Note: Ensure that you implement the pseudocode from the lecture. Other algorithms will not score full marks for this exercise even if they also result in the correct insertion of the new element.

   We have created a script that can automatically test your program. To run this test you can execute the dryrun program that corresponds to this exercise. It expects to find the file named RBTree.c in the current directory with your implementation of the function TreeInsert().

   You can use dryrun as follows:

   ```
   prompt$ 9024 dryrun RBTree
   ```

Due Date

Tuesday, 22 April, 11:59:59. Late submissions will not be accepted.

Submission

You should submit your file using the following give command:

```
prompt$ give cs9024 week9 BST.c RBTree.c
```

Alternatively, you can select the option to "Make Submission" at the top of this page to submit directly through WebCMS3.

Important notes:

- Make sure you spell all filenames correctly.
- You can run `give` multiple times. Only your last submission will be marked.
- Where you're expected to submit multiple files, ensure they form a single submission. If you separate them across multiple submissions, each submission will replace the previous one.
- Whether you submit through the command line or WebCMS3, it is your responsibility to ensure it reports a successful submission. Failure to submit correctly will not be considered as an excuse.
- You cannot obtain marks by e-mailing your code to tutors or lecturers.

Assessment

- Each question is worth 1 mark, for a total of 2 marks.
- Submissions will be auto-marked shortly after the deadline has passed.
- It is important that the output of your program follows exactly the format of the sample output, otherwise auto-marking will result in 0 marks.
- Ensure that your program compiles on a CSE machine with the standard options, i.e. `-Wall -Werror -std=c11`. Programs that do not compile will receive 0 marks.
- Auto-marking will use test cases different to `dryrun`. (Hint: do your own testing in addition to running `dryrun`).

Collection

Once marking is complete you can collect your marked submission using the following command:

```
prompt$ 9024 classrun -collect week9
```

You can also view your marks using the following command:

```
prompt$ 9024 classrun -sturec
```

You can also collect your marked submission directly through WebCMS3 from the "Collect Submission" tab at the top of this page.

Plagiarism

Group submissions will not be allowed. Your programs must be entirely your own work. Plagiarism detection software will be used to compare all submissions pairwise (including submissions for similar assessments in previous years, if applicable) and serious penalties will be applied, including an entry on UNSW's plagiarism register.

- ***Do not copy ideas or code from others***
- ***Do not use a publicly accessible repository or allow anyone to see your code***

Please refer to the on-line sources to help you understand what plagiarism is and how it is dealt with at UNSW:

- Plagiarism and Academic Integrity
- UNSW Plagiarism Policy Statement
- UNSW Plagiarism Procedure