Algorithms and Data Structures
Jacobs University Bremen
Dr. Kinga Lipskoch

Course: CH-231-A
March 30th, 2020

Homework 9

- Submit one ZIP file per homework sheet which contains one PDF file (including pictures, computations, formulas, explanations, etc.) and your source code file(s) with one makefile and without adding executable, object or temporary files.
- The implementations of algorithms has to be done using C, C++, Python or Java.
- The TAs are grading solutions to the problems according to the following criteria: https://grader.eecs.jacobs-university.de/courses/ch_231_a/2020_1/Grading_Criteria_ADS.pdf

Problem 9.1 *Understanding Red Black Trees* (9 points)

(a) (3 points) Draw (or describe by using preorder traversal) the red-black trees that result after successively inserting the values step by step in the following order $[13, 44, 37, 7, 22, 16]$ into an empty red-black tree. You are required to draw (or describe by using preorder traversal) the tree after each insertion, as well as any additional recoloring and balancing.

(b) (3 points) Draw (or describe by using preorder traversal) all valid red-black trees that store the values $\{1, 2, 3, 4\}$.

(c) **Bonus** (3 points) Consider a red-black tree formed by inserting n nodes with the algorithm described in the lecture slides. Prove that if $n > 1$, the tree contains at least one red node.

Problem 9.2 *Implementing Red Black Trees* (16 points)

Implement a red black tree (with integer nodes), closely following the specifications and algorithms from the lecture. Make sure you handle errors appropriately by printing messages or throwing exceptions. Your implementation has to be along the interface below with the following or equivalent components:

```
enum Color {RED, BLACK};
struct Node
{
  int data;
  Color color;
  Node *left, *right, *parent;
};
class RedBlackTree
{
  private:
    Node *root;
  protected:
    void rotateLeft(Node *&);
    void rotateRight(Node *&);
  public:
    RedBlackTree();
    void insert(int);
    void delete(Node *&);
    Node * predecessor(const Node *&);
    Node * successor(const Node *&);
    Node * getMinimum();
    Node * getMaximum();
    Node * search(int);
};
```

How to submit your solutions

You can submit your solutions via *Grader* at **`https://grader.eecs.jacobs-university.de`** as a generated PDF file and/or source code files.
If there are problems with *Grader* (but only then), you can submit the file by sending mail to `k.lipskoch@jacobs-university.de` **with a subject line that starts with CH-231-A**.

Please note, that after the deadline it will not be possible to submit solutions. It is useless to send solutions by mail, because they will not be graded.

This homework is due by Tuesday, April 14th, 23:00.