Lecturer: B. Gärtner

Informatik für Mathematiker und Physiker HS12

Exercise Sheet 12

Submission deadline: 3.15pm - Tuesday 11th December, 2012

Course URL: http://www.ti.inf.ethz.ch/ew/Lehre/Info1_12/

Note: You should test your implementations thoroughly. For each class you altered, write a program that tests all of its functionality.

Assignment 1 - (1+2+7 points)

From the course web page, download the following files:

- TreapNode.h - header file of the `TreapNode` class (i.e. class representing a node of a treap)
- TreapNode.cpp - implementation of the `TreapNode` class
- TreapSkeleton.h - partial header of the `Treap` class
- TreapSkeleton.cpp - partial implementation of the `Treap` class
- treapSkeletonTest.cpp - test program for the `Treap` class, with expected output

Inspect the provided code. The class `Treap` represents a randomized binary search tree. Your task is to implement the following methods in the `Treap` class that is partially implemented in the TreapSkeleton.cpp file:

1. `int Treap::size() const` - returns the *size* of `*this`, defined as the number of nodes in `*this`.
2. `int Treap::height() const` - returns the *height* of `*this`, defined as the length of the *longest* path from `root_` to a leaf of `*this`. If the treap is empty, there is no such path. In that case, the function should return -1. If the tree consists of only one node, the height is 0.
3. `void Treap::remove(int key)` - removes one node with the given `key`. After this is executed, `*this` remains a treap. If such a node is not present, `*this` remains unchanged.

Hint: For all three member functions, it is convenient to delegate the work to auxiliary private functions (see the case of `insert`). For this reason, you might need to enhance TreapSkeleton.h.

Assignment 2 - (6 points)

A *balanced binary search tree* (BBST) is a binary tree with the following property: for every node v, the height of the left subtree of v and the height of the right subtree of v differ by at most one.

Prove the following statement: The height of a BBST with n nodes is at most $2\log_2(n+1)$.

Hint: It may help to first prove the following three auxiliary statements.

1. A BBST of height h has at least F_{h+1} leaves, where F_k is the k-th Fibonacci number, recursively defined as

$$\begin{aligned} F_0 &:= 0, \\ F_1 &:= 1, \\ F_k &= F_{k-1} + F_{k-2}, \quad k \geq 2. \end{aligned}$$

2. $F_k \geq 2^{\lfloor (k-1)/2 \rfloor}$ for $k > 0$.

3. A binary tree with n nodes has at most $\frac{n+1}{2}$ leaves.

Challenge - (8 points)

Provide a member function for the class `Treap` that performs *pretty printing* i.e. outputs a drawing that visualizes the treap structure. Here is how this could look like for a treap with 10 nodes (the root is the leftmost vertex, so the tree "grows to the right" in this drawing).

```
        --(9)
       |
    --(8)
   |   |
   |    --(7)
   |
--(6)
   |
   |        --(5)
   |       |
   |    --(4)
   |   |   |
   |   |    --(3)
   |   |
    --(2)
       |
        --(1)
           |
            --(0)
```