

figure=../../../../../BUseal.eps,height=1.125in,angle=0

Assignment 1

Out: Thursday, 2 Septmeber 2010

Due: Tuesday, 14 September 2010

Academic Integrity We pledge strict adherence to the university guidelines.

- All work you turn in must be *your own* unless specified otherwise.
- You are allowed to discuss problems with your classmates but you must write your own code and solutions.
- Please always remeber that every student deserves a chance to achieve a fair grade!

Total: 130 points + 50 bonus points

Exercise 1 (10 points) Prove the following statement using mathematical induction.

$(2n + 1)^2 - 1$ is a multiple of 8 for every natural number n .

Exercise 2 (70 points) Binary trees are defined as follows and we use $B(t)$ for the number of branch node B in t .

binary trees $t ::= E \mid B(t, t)$

Let us define function h on binary trees inductively.

$h(E) = 0$ $h(B(t_l, t_r)) = 1 + \max(h(t_l), h(t_r))$

Evidently, $h(t)$ computes the height of t . Braun trees are binary trees defined as follows.

- E is a Braun tree.
- $B(t_l, t_r)$ is a Braun tree if t_l and t_r are Braun trees and $B(t_r) \leq B(t_l) \leq B(t_r) + 1$.

Please do the following.

- (20 pts) Prove by structural induction on t that for every nonempty Braun tree t , $2^{h(t)-1} \leq B(t) < 2^{h(t)}$ holds.
- (20 pts) Implement in ATS a procedure that lists all Braun trees of height n when given a natural number n .

```
datatype BraunTree = E | B of (BraunTree, BraunTree)
```

```
fun listBraunTrees (height: int): list0 BraunTree = (* your code is here *)
```

- (30 pts) In general, what is the number of Braun trees of height n for a given natural number n ? Please justify your answer by mathematical induction.

Exercise 3 (50 points + 50 bonus points) A red-black tree is a binary tree in which each node is assigned either the red or the black color. The color of a red-black tree is the color of its root if the tree is non-empty, or it is black. In addition, a red-black tree must satisfy the following constraints:

- *the children of each red node must be black, and*
- *each path (from the root to a leaf) must contain the same number of black nodes (which is often referred to as **black height**).*

It can be readily proven that each red-black tree is a balanced binary tree.

- *(50 pts) Please implement a function in C that performs insertion operation on a doubly-linked red-black tree.*
- *(50 bonus pts) Please implement a function in C that performs removal operation on a doubly-linked red-black tree.*