

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

## Assignment 2

**Out: Tuesday, 14 September 2010**  
**Due: Tuesday, 28 September 2010**

**Total:** 100 points + 40 bonus points

**Exercise 1** (30 points) Let us use the following datatypes `term` and `term1` for representing untyped (pure)  $\lambda$ -terms and their  $\alpha$ -normal forms, respectively:

`datatype term =`

`TmVar of string | TmLam of (string, term) | TmApp of (term, term)`

`datatype term1 =`

`TmVar1 of string | TmInd1 of int | TmLam1 of term1 | TmApp1 of (term1, term1)`

- (10 points) Please implement a function `nf_alpha` that translates a given  $\lambda$ -term into its alpha-normal form:

`extern fun nf_alpha (t: term): term1`

- (20 points) Please implement a function `subst` of the following type:

`extern fun subst (t: term, x: string, s: term): term`

Given a term  $t$ , a variable  $x$  and a term  $s$ , `subst(t, x, s)` should return  $t[x \mapsto s]$ .

**Exercise 2** (20 points) Please prove in ATS that  $(2n+1)^2 - 1$  is a multiple of 8 for every natural number  $n$ .

**Exercise 3** (30 points) The following declared dataprop `F91` encodes the MacCarthy's 91-function:

`dataprop F91 (int, int) =`

`| F91def1 (91, 91)`  
`| {i:int | i <= 100; i <> 91} {r1,r2:int}`  
`F91def2 (i, r2) of (F91 (i+11, r1), F91 (r1, r2))`  
`| {i:int | 101 <= i} {r:int}`  
`F91def3 (i, r) of F91 (i-10, r)`

Given integers  $i$  and  $r$ , if a proof value can be assigned the type `F91(i, r)`, then  $f91(i) = r$ , where  $f91$  is formally defined as follows:

$$f91(i) = \begin{cases} f91(i-10) & \text{if } i \geq 101; \\ f91(f91(i+11)) & \text{if } i \leq 100 \wedge i \neq 91; \\ 91 & \text{otherwise.} \end{cases}$$

Please construct a proof function `F91istot` in ATS that is declared as follows:

```
prfun F91listot {i:int} (): [r:int] F91 (i, r)
```

**Exercise 4** (40 bonus points) *The definition of Braun trees is encoded into the following declared dataprop `isBraun`:*

```
datasort bt = B of (bt, bt) | E of ()
```

```
dataprop isBraun (bt) =  
  | {t1,t2:bt} {s1,s2:nat | s2 <= s1; s1 <= 1 + s2}  
    isBraun_B (B (t1, t2)) of (isBraun t1, isBraun t2, btsz (t1, s1), btsz (t2, s2))  
  | isBraun_E (E ()) of ()
```

*Please construct a proof function `brauntree_height_lemma` in ATS that is declared as follows:*

```
prfun brauntree_height_lemma {t1,t2:bt} {h,h1:nat}  
  (pf0: isBraun (B (t1, t2)), pf1: btht (B (t1, t2), h), pf2: btht (t1, h1))  
  : [h == h1+1] void
```

*Please also construct a proof function `brauntree_size_height_lemma` in ATS that is declared as follows:*

```
prfun brauntree_size_height_lemma {t:bt} {s,h,n:nat}  
  (pf0: isBraun (t), pf1: btsz (t, s), pf2: btht (t, h), pf3: POW2 (h, n))  
  : [n <= s + s + 1] void
```

*Note that the dataprops `btsz`, `btht` and `POW2` are all declared in a file available on-line.*