Universität Innsbruck **SS 2008**
Institut für Informatik **Exercise Sheet 5**
Prof. Dr. Clemens Ballarin **Due 12 June 2008**

Automatic Deduction — Introduction to Isabelle
LVA 703522

1 Two Grammars

The most natural definition of valid sequences of parentheses is this:

$$S \quad \rightarrow \quad \epsilon \quad | \quad '(' \; S \; ')' \quad | \quad S \; S$$

where ϵ is the empty word.

A second, somewhat unusual grammar is the following one:

$$T \quad \rightarrow \quad \epsilon \quad | \quad T \; '(' \; T \; ')'$$

▷ Make the definitions of sets `S` and `T` in Isabelle and give structured Isar proofs leading to the theorem `S = T`.

Hint: use lists over a specific type to repesent words.

The alphabet:

```
datatype alpha = A | B
```

2 Polynomial sums

▷ Produce structured proofs of the following theorems, using induction and calculational reasoning in Isar.

Note that the given tactic scripts are of limited use in reconstructing structured proofs; nevertheless the hints of automated steps below can be re-used to finish sub-problems. The $\sum$ symbol can be entered as "`\<Sum>`"; note that numerals in Isabelle/HOL are polymorphic.

```
theorem — problem
  fixes n :: nat
  shows "2 * (∑i=0..n. i) = n * (n + 1)"
  by (induct n) simp_all
```

```
theorem — problem
  fixes n :: nat
  shows "(∑i=0..<n. 2 * i + 1) = n^2"
```

```
  by (induct n) (simp_all add: power_eq_if nat_distrib)


theorem — problem
  fixes n :: nat
  shows "(∑i=0..<n. 2^i) = 2^n - (1::nat)"
  by (induct n) (simp_all split: nat_diff_split)


theorem — problem
  fixes n :: nat
  shows "2 * (∑i=0..<n. 3^i) = 3^n - (1::nat)"
  by (induct n) (simp_all add: nat_distrib)


theorem — problem
  fixes n :: nat
  assumes k: "0 < k"
  shows "(k - 1) * (∑i=0..<n. k^i) = k^n - (1::nat)"
  by (induct n) (insert k, simp_all add: nat_distrib)
```