## Programming Language Translation

## Practical 4 Handin

## <Group names here>

### Task 1 Palindromes (6 marks)

1. Does grammar 1 describe palindromes? If not, why not?

No. The grammar does not terminate (it has non-terminals in the final production). Therefore, it does not describe palindromes.

Is it an LL(1) grammar? If not, why not?

Yes, it is LL(1) compliant. The sets of the First symbols for Palin1’s options are disjoint.

2. Does grammar 2 describe palindromes? If not, why not?

No.

Is it an LL(1) grammar? If not, why not?

No. The sets of first symbols for Palin2’s options are not disjoint. “a” and “b” are the start of more than one alternative.

3. Does grammar 3 describe palindromes? If not, why not?

Is it an LL(1) grammar? If not, why not?

No. Rule 2.

4. Does grammar 4 describe palindromes? If not, why not?

Is it an LL(1) grammar? If not, why not?

No. Rule 1.

5. Can you find a better grammar to describe palindromes? If so, give it, if not, explain why not.

No. The productions are EBNF – type 2 grammars. This means that they cannot determine (with LL(1)) anbncn.

### Task 2 Expressions again [8 marks]

Is this an ambiguous grammar? (Hint: try to find an expression that can be parsed in more than one way). Give the parse trees (hand drawn figures are fine).

This grammar is ambiguous. You could parse b ^ a - b

Is expression.atg LL(1) compliant?

If not, why not and can you find a suitable grammar that is LL(1)? Hand in the .atg file of the new grammar.

Prove, by applying Rule 1 and Rule 2 (if applicable) that the new grammar is LL(1) compliant.