## Programming Language Translation

## Practical 4 Handin

## <Wynne Edwards g21e2079, Mila Davies g21d6937, Manu Jourdan g21j5408>

### Task 1 Palindromes (6 marks)

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

**No, this grammar can’t describe any palindromes. There is no base case for the recursion to terminate.**

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

**No, for the same reason as stated above.**

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

**Yes, it only describes odd number palindromes.**

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

**No, the grammar violates LL(1) rule 1.**

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

**Yes, it only describes even number palindromes.**

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

**No, the grammar violates LL(1) rule 1 & rule 2.**

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

**Yes, describes only odd number palindromes.**

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

**No, the grammar violates LL(1) rule 1 & rule 2.**

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

**No. You can accommodate for either even or odd but not both. Due to the recursive nature of the grammar, you can’t allow for even and odd palindromes whilst keeping the grammar LL(1) compliant.**

### 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).

**Exp = Term A**

**A = B Term A | null**

**B = + | -**

**Term = Fac C**

**C = D Fac C | null**

**D = \* | /**

**Fac = Pri E**

**E = ^ Exp E | null**

**Pri = a | b | c**

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.