Artificial Intelligence and Logic
Erlang Project

Write an expression manipulator that manipulates valid arithmetic expressions.

You can assume that:

1. Only integers are used;
2. +,-,*,/ are the only binary operators allowed;
3. "~" representing unary minus is the only unary operator allowed;
4. Brackets are used to indicate order of evaluation.

The expression manipulator will consist of the following components:

- **Tokeniser:** Takes as input a string and converts it into a list of tokens
  (30 marks)
- **Parser:** Takes as input a list of tokens and converts it into an appropriate internal syntax tree representation;
  (40 Marks)
- **Evaluator:** Takes as input the internal representation of an expression and returns the value of the expression;
  (30 Marks)

Example output of each stage:
String: "((2+3)-4)"

Tokenised Expression: [{sym, lbracket},{sym,lbracket},{num,2},{binOp,plus},{num,3},{sym,rbracket},{binOp,minus},{num,4},{sym,rbracket}]

Internal Representation: {minus, {plus, {num, 2}, {num 3}}, {num, 4}}

Evaluator: 1

Bonus Material (Optional)
Allow conditional expressions as part of an expression (0 is false, non-zero is true)
For Example: **if (1+(2-4)) then 5 else (6/3)**

Add different types (e.g. boolean and float)

Add type checking

Add local definitions (variables) as part of an expression
For Example: **let b = (5+(7-2)) in ~(3*(b/2))**