Homework: ArithLang

Learning Objectives:

1. Write programs in Arithlang
2. Understand and extend Arithlang interpreter

Instructions:

1. Total points: 50 pt
2. Early deadline: Feb 17 (Wed) 11.59 pm, Regular deadline Feb 19 (Fri) 11:59 pm
3. Download tutorial.zip and tutorial.pdf from Canvas
4. Set up the programming project according to the instructions in the tutorial
5. How to submit:
   - For questions 1–3, you can write your solutions in latex or word and then convert it to pdf; or you can submit a scanned document with legible handwritten solutions. Please provide the solutions in one pdf file.
   - For questions 4 and 5, please submit your solutions in two different zip files with all the source code files (just zip the complete project's folder).
   - Submit two zip files and one pdf file to Canvas under Assignments, Homework 2

Questions:

1. (4 pt) [Writing ArithLang programs] Write two Arithlang programs that compute 342, containing three or more operators.
2. (4 pt) [Arithlang syntax] Get familiar with the syntax of Arithlang: construct parse trees for the following programs in ArithLang:
   (a) (* (/ (- 30 10) 4) (+ 3 (- 5 4)))
   (b) (- 300 (+ 100 10) (* 6 5 4))
3. (6 pt) [ArithLang interpreter understanding] Get familiar with the ArithLang source code:
   (a) (2 pt) What is the purpose of the AST and Evaluator classes?
   (b) (2 pt) Use an example ArithLang program to explain how the visitor pattern works to compute the value of this program?
   (c) (2 pt) How is the visitor pattern implemented in the Evaluator and AST classes?

4. (10 pt) [Arithlang interpreter basics] Extend ArithLang to support a new expression greatest-of (>? a b) for positive numbers, e.g.,
   $ (>? 3)
   3
   $ (>? 3 4 2)
   4
   $ (>? 8 3)
   8
   $ (>? 8 -1)
   error
   $ (>? -1)
   error

   You will need to modify the following files:

   - (2 pt) Grammar file (ArithLang.g)
   - (2 pt) AST file (AST.java)
   - (1 pt) Printer file (Printer.java)
   - (5 pt) Evaluator (Evaluator.java)

5. (26 pt) [ArithLang interpreter advanced] Implement an interpreter for AbstractLang described as follows. You can modify the ArithLang code provided.

   (a) This language contains only six terminals, p, n, u, +, -

   (b) p represents positve numbers, n represents negative numbers and u represent unknown values

   (c) There are two operators + and - that can be applied on the terminals, their syntactic rules are similar to + and - in ArithLang, except that each operator only can take two operands

   (d) The semantics of AbstractLang are defined by the following rules (the first column in the table represents the first operand and the first row in the table represents the second operand of the operation):

+	**p**	**n**	**u**
p	p	u	u
n	u	n	u
u	u	u	u

-	**p**	**n**	**u**
p	u	p	u
n	n	u	u
u	u	u	u

   e.g.,
   (+ p p) = p
   (- n p) = n
   (- (+ n u) p) = u
   (+ p p p) = error
   (+ 3 4) = error
   (* 2 p) = error
   (+ a b) = error

   You will need to modify the following files:

   - (6 pt) Grammar file (AbstractLang.g)
   - (4 pt) AST file (AST.java)

- (3 pt) Printer file (Printer.java)
- (3 pt) Value file (Value.java)
- (10 pt) Evaluator (Evaluator.java)