Homework: ArithLang

Learning Objectives:

1. Write programs in Arithlang
2. Understand ArithLang syntax, understand and extend Arithlang interpreter

Instructions:

1. Total points: 50 pt
2. Early deadline: Sept 12 (Wed) 6:00 pm, Regular deadline Sept 14 (Fri) 6:00 pm
3. Download hw2code.zip and TutorialSetupFramework.pdf from Canvas
4. Set up the programming project following 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) Write two Arithlang programs that compute 24, containing more than three operators.
2. (4 pt) Get familiar with the syntax of Arithlang: write the derivations for the following programs in ArithLang
   (a) leftmost derivation: (* (+ 2 (- 5 8)) (* 2 1))
   (b) rightmost derivation: (/ 34 (* 4 2) (+ 34 67))
3. (6 pt) Get familiar with the ArithLang source code: the Evaluator and Formatter classes used the visitor patterns, explain what are the purposes of the two classes and how the visitor patterns are implemented.
4. (10 pt) Extend ArithLang to support the greatest common factor operation, (gcf num1 num2), e.g.,
   (gcf 30 60) = 30
   (gcf (gcf 45 60) 90) = 15
   (gcf 45 60 90) = 15
   (gcf 10) error
   (gcf -10 20) error

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

   (a) This language contains only three terminals, 0, p, n, u

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

   (c) There are three operators *, -, + 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):

+	**0**	**p**	**n**	**u**
0	0	p	n	u
p	p	p	u	u
n	n	u	n	u
u	u	u	u	u

-	**0**	**p**	**n**	**u**
0	0	n	p	u
p	p	u	p	u
n	n	n	u	u
u	u	u	u	u

*	**0**	**p**	**n**	**u**
0	0	0	0	0
p	0	p	n	u
n	0	n	p	u
u	0	u	u	u