Assignment:

- A value can be assigned to a variable using the form,
  <variable> is <value/expression>
- Examples,
  X is 2
  X is X+1

Arithmetic:

- Addition, subtraction, division, and multiplication of numbers can be done using the operators +, -, /, and *, respectively.
- Arithmetic takes the form,
  <variable/value/expression> <operator> <variable/value/expression>
- Examples,
  X + 1
  X+Y+Z
  X + Y - Z

Relational:

- Relational operation is possible. The relation operations “equal”, “less than”, and “greater than” use the operators =, <, and >, respectively.
- Examples,
  X < Y
  X > Y
  X=Y
- The relational operations “less than or equal” and “greater than or equal” can be achieved by using the operators =< and >=, respectively.
- Examples,
  X=<Y
  X>=Y

Logical:

- The logical operations AND, OR, and NOT use the operators **,** (comma), **;** (semicolon), **\+**, respectively.

Useful pre-defined Prolog elements:

- rem – takes two arguments, M and N, and returns the remainder of M/N process.
  rem(2,3) is 2
- write – accepts a literal (constant or string – enclosed in single quotation marks) and displays the argument (or value of the variable) on screen.
  write('HEY!')
  HEY!
  write('hello world!')
  hello world!
  X is 2
  write(X)
  2
  X is 2
  write('value: '), write(X)
  value: 2

Exercises:

Create FACTS and/or RULES for the following problems (use the 'function' or rule name in the prolog interface):

- squareNum(N) – takes a number and displays the square of the number
- compareXY(XY) – takes two numbers (assume >-1) and displays -1 if the numbers are equal, otherwise display the smaller value
- checkEven(N) – takes a number (should be >0) and checks if it is even or not
- primeNumber(N) – takes a number (should be >1) and checks if it is prime or not
- factorial(N) – displays the factorial of the argument (should be >=0)
- fibonacci(N) – displays the fibonacci number of N (should be >0; note: 1 for 1, 1 for 2, 2 for 3, 3 for 4, 5 for 5, etc.)