Давайте сконструируем язык **L** через **ПРЯМУЮ (префиксную) польскую запись**.

Зададим алфавит $\Sigma_0 = \{a, b, ..., z, A, B, ..., Z, 0, ..., 9, \_\}$ - буквы, цифры и нижнее подчёркивание
Зададим наш терминальный алфавит $\Sigma = \Sigma_0 \cup \{\ ,\ ...\ ,\ \}$, то есть это Σ_0 и все пробельные символы
Нетерминальный алфавит $N = \{s_0, E_0, V_0, D_0, \_\}$, где s_0 - стартовое состояние, E_0 - это Expr, V_0 - это Var и Ident (в данном случае это одно и тоже), D_0 - это числа, $\_$- все пробельные символы (причём давайте считать, что это всевозможные их комбинации, то есть может быть несколько пробелов подряд)
!Важное замечание про пробелы! : все токены разделены пробелами,о есть ключевые слова, переменные, операторы и числа. Далее в грамматике указаны пробелы только внутри одной конструкции, но между конструкциями пробелы тоже обязаны быть.

В нашем языке есть ключевые слова **If, While, Assign, Read, Write и Seq, Nop**
(Nop - пустая инструкция)
Для каждого из них введём правило:

- $s_0 \to If\_E_0\_s_0\_s_0$
  (Порядок веток прямой, соответственно если условие E_0 выполнится, то будет исполнятся первый s_0, иначе второй)
- $s_0 \to While\_E_0\_s_0$
- $s_0 \to Assign\_V_0\_E_0$
- $s_0 \to V_0\_Read$
- $s_0 \to Write\_E_0$
- $s_0 \to Seq\_s_0\_s_0$
  (Порядок прямой, соответственно вначале идёт первое действие, затем второе)
- $s_0 \to Nop$
  (пустая инструкция)

Выражения:
Правила для бинарных операторов (порядок везде прямой):

- $E_0 \to -\_E_0\_E_0$
  (Из первого E_0 вычитаем второе E_0)
- $E_0 \to +\_E_0\_E_0$
  ...
- $E_0 \to \&\&\_E_0\_E_0$

- $E_0 \to ||\_E_0\_E_0$

Правила для унарных операторов (унарный минус и отрицание):

- $E_0 \to --\_E_0$ (унарный минус)
- $E_0 \to !\_E_0$ (отрицание)

Правила для чисел и переменных:

- $E_0 \to D_0$
- $E_0 \to V_0$

Переменные и идентификаторы - это строки, которые начинаются обязательно с буквы либо нижнего подчёркивания и состоят из символов Σ_0, причём обязательно **НЕ совпадают** с ключевыми словами

- $V_0 \to (a|b|...|\_|\_a|...|aa|ab|...|Alya\_0|.....|Fkb1aa1c\_d3|...)\ \Sigma_0^*$

(Перечисляем все корректные слова длины < 15 **(кроме ключевых слов)** и дальше добавляем что угодно, то есть Σ_0^*)

Числа:

- $D_0 \to (0|1|...|9)^+$

Пробелы:

- $\_ \to (\backslash t \mid \backslash n \mid ...)^+$ (все пробельные символы и любые их комбинации)

UPD: 16.04.2020
Функции
Появились новые ключевые слова: **Fun, With, ThatsAll, Call, Return, WithFun, ThatsAllFun**

Fun - объявление функции
With - конкатенация двух списков аргументов
ThatsAll - синоним With , но только с нулём списков
Call - вызов функции
Return - вернуть значение из функции
WithFun - как Seq только для функций, то есть просто их перечисление, принимает два

списка функций
ThatsAllFun - синоним WithFun , но только с нулём списков

Формально:

- $Arg \to ThatsAll \mid With\_V_0\_Arg$
  (список аргументов)
  Примеры:
  $ThatsAll$
  $With\ kek\ With\ mem\ ThatsAll$
  $With\ kek\ ThatsAll$
  (то есть список всегда заканчивается ThatsAll-ом)

- $F_0 \to Fun\_V_0\_Arg\_s_0$
  (Объявление функции, это НЕ часть LAst, поэтому нельзя например объявлять функцию внутри функции)

- $ArgCall \to ThatsAll \mid With\_E_0\_Arg$
  (список аргументов, но уже для вызова функции, то есть вместо переменных тут выражения)
  Примеры:
  $ThatsAll$
  $With\ 1\ With\ 10\ ThatsAll$
  $With\ +\ 1\ 2\ ThatsAll$
  (то есть список всегда заканчивается ThatsAll-ом)

- $E_0 \to Call\_V_0\_ArgCall$
  (Вызов функции, это часть Ast)

- $s_0 \to Return\_E_0$
  (return, это часть LAst)

- $FunList \to ThatsAllFun \mid WithFun\_F_0\_FunList$
  (Это просто список функций, всегда оканчивается на ThatsAllFun)

Корректная программа на языке L
Корректным кодом на нашем языке является выражение вида:
$Start- > FunList\_s_0$