Задание

В этом домашнем задании вам предстоит разработать грамматику для небольшого языка программирования, а также реализовать его интерпретатор. Выражения в этом языке выглядят следующим образом:

```
a = 2

b = 3

c = 2 * 5

d = c - a
```

Каждая строка состоит из названия переменной, знака равно и выражения, которое присваивается в эту переменную. Выражения с правой части являются арифметическими выражениями, с числами (ограничьте их сверху чем-то разумным), названиями переменных, операциями `+, -, *, /`, скобками.

Следующие дизайн-решения остаются на ваше усмотрение:

- Какие идентификаторы считать корректными названиями переменных,
- Разрешать ли перезаписывать уже объявленные переменные
- Тип используемых переменных (32 бита, 64 бита, или произвольные)

Грамматика (1 балл)

Вы должны придумать и описать грамматику для этого языка. Оформите небольшое описание в формате Word/LaTeX/Markdown. Приложите описание грамматики к письму.

Интерпретатор (2 балла)

По построенной грамматике реализуйте интерпретатор, вычисляющий все выражения. Интерпретатор должен считывать все строчки из `stdin`/файла, выполнять каждую строчку, и вывести результаты вычислений в виде

```
a = 2

b = 3
```

```
c = 10

d = 8
```

При возникновении синтаксической ошибки, использования необъявленных переменных, деления на ноль в соответствующей строке следует вывести сообщение об ошибке (чем детальнее, тем лучше). Приложите к письму код или ссылку на репозиторий.

Можно использовать генераторы парсеров по грамматике, при этом, пожалуйста, опишите в мини-отчете то, что вы используете, а также как грамматика из первого пункта переносится во входные данные для генератора.

Тесты (1 балл)

Создайте несколько файлов с тестами, покрывающие поведение вашего интерпретатора. Тесты на ошибки парсинга и вычислений тоже должны присутствовать. Если вы реализуете какое-либо из расширений ниже, то их тоже стоит протестировать.

Расширения (2 балла)

Вы можете расширить язык, каждое расширение принесет +1 балл (не более +2 суммарно). Возможные расширения:

`if-then-else`

В качестве условия можно использовать число (если не ноль, то выполняется первая ветка, иначе вторая). Ветка с `else` может не присутствовать. Детали синтаксиса на ваше усмотрение.

```
if (a - b) {

    c = a - b

} else {

    c = b - a

    d = a + b

}
```

Функции

Ввести в грамматику возможность определять функции. Возможно отдельное определение вида `def f(x) {...}`, или `f = lambda x: x + 100 / (50 - x)`

Логические выражения с `and, or, xor, not`

Добавить сравнения чисел (`<, >, =, !=, ...`), тип которых - `bool`, и операции с `bool`.

Добавить в язык массивы и операции с ними

Комплексные числа?

Добавить новый тип данных (строки, матрицы, ...) и операции с ними

Циклы `for, while, ...`

Что-нибудь нетривиальное, что вы можете сами придумать

Формат и дедлайны

Отправляйте на стандартный ящик со стандартной темой и HW10.

Hard deadline: **Воскресенье, 17 мая, 23:55**.

Soft дедлайна как такового не будет, но я буду проверять все письма:

- Днем 9 мая
- Во вторую половину среды, 13 мая