Ідея:

DLine – проста скриптова мова програмування. Вона виконується пострічно, що означає, що роздільником між виразами є перехід на новий рядок. Мова підтримує арифметичні операції, виклики деяких математичних функцій, а також створення змінних.

Коментарі починаються зі знака #, але охоплюють тільки увесь рядок, тобто перед коментарем у тому ж рядку не може йти вираз. Ось приклад програми, написаної на DLine:

```
# Solve the equation 5x^2 - 18x + 9 = 0
a = 5
b = -18
c = 9
D = b**2 - 4 * a * c
x1 = (-b + sqrt(D)) / (2 * a)
x2 = (-b - sqrt(D)) / (2 * a)
x1
x2
```

Результати виконання:

- ⇨ *3*
- ⇨ *0.6*

Можна помітити знак '**', який підносить вираз перед ним у якусь степінь. Щоб вивести значення якоїсь змінної або просто виразу, необхідно просто написати його, не використовуючи у виразі оператор присвоєння. Можна зчитати програму з файлу, якщо при запуску DLine з командного рядку другим параметром указати назву файлу.

Реалізація:

Головний клас, який відповідає за виконання програми – Parser. У ньому вираз аналізується методом рекурсивного спуску і розбивається на складові граматики. Граматика доволі проста і побудована таким чином, щоб унарні вирази були більш пріоритетні за множення, а множення більш пріоритетне за додавання, тощо.

Було б добре розбивати стрічку на токени, а потім вже працювати з токенами. Але через брак часу та простоту програми я вирішив розбирати правильність та обраховувати вираз, що називається, in-place.

Для більш коректного обчислення мені довелося створити клас OptionalDouble, ідею якого я запозичив з Java (і, як прочитав, він буде реалізований у C++20). Ідея проста – це double, але який не містить значення. Він дуже корисний для випадків, коли користувач здійснив якусь помилку, або коли ми не маємо повертати значення з функції, яка іноді його повертає.

Для обробки помилок я написав декілька розширень std::exception.

Плани:

Контекстно-вільні граматики одна з найулюбленіших моїх речей у інформатиці, тож у майбутньому я планую написати токенізацію та додати до мови функції та структури.