|  |  |  |
| --- | --- | --- |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное автономное  образовательное учреждение высшего образования  «Пермский государственный национальный  исследовательский университет» | | |
|  | Институт компьютерных наук и технологий | |
| **ОТЧЁТ**  по индивидуальной работе №2  по дисциплине «Язык программирования C++»  Вариант 20 | | |
|  | | Работу выполнил  студент группы ИТ-11,12-2024 1 курса  Обыдёнов Ю.Д.  «24» Июня 2025 г. |
| Работу проверил  Кнутова Н.С  «26» Июня 2025 г. |
| Пермь 2025 | | |

**Содержание:**   
1.Постановка задачи ....................................................................................... 3

2.Алгоритм решения ....................................................................................... 4

3.Тестирования ................................................................................................ 7

4.Код программы .............................................................................................8

# Постановка задачи

Написать программу, которая по заданной формуле строит дерево и производит вычисления с помощью построенного дерева. Формула задана в традиционной инфиксной записи, в ней могут быть скобки, максимальная степень вложенности которых ограничивается числом 10. Аргументами могут быть целые числа и переменные, задаваемые однобуквенными именами. Допустимые операции: +, -, \*, /. Унарный минус допустим. С помощью построенного дерева формулы упростить формулу, заменяя в ней все поддеревья, соответствующие формулам (f1\*f3+f2\*f3) и (f1\*f2+f1\*f3) на поддеревья, соответствующие формулам ((f1+f2)\*f3*) и (f1\*(f2*+f3)).

Разработать программу, которая:

* Принимает математическое выражение в инфиксной нотации
* Строит синтаксическое дерево выражения
* Упрощает выражение по заданным шаблонам:
  + (f₁ \* f₂ ± f₃ \* f₆) → (f₁ ± f₂) \* f₈
  + (f₁ \* f₂ ± f₁ \* f₈) → f₁ \* (f₂ ± f₃)
* Вычисляет результат выражения
* Поддерживает:
  + Целые числа и однобуквенные переменные
  + Операции: +, -, \*, /
  + Унарный минус
  + Скобки (максимальная вложенность: 10)
* Интерфейсы: ручной ввод, чтение из файла, генерация случайных выражений

#### **2. Алгоритм решения**

**Основные компоненты:**

1. **Класс Node**:
   1. Хранит значение (число, переменная, операция)
   2. Ссылки на левого и правого потомков
2. **Парсер (Parser)**:

Рекурсивный нисходящий парсер

1. **Функции обработки дерева**:

eval(): вычисление выражения с подстановкой значений переменных

print(): вывод выражения в инфиксной форме

simplify(): упрощение по заданным правилам

deepCopy(): глубокое копирование дерева

isSame(): сравнение структур деревьев

1. **Алгоритм упрощения**:
   1. **Рекурсивный обход** дерева (пост-ордер)
   2. **Правила замен**:

Вынесение общего множителя:// (a\*b) + (a\*c) → a\*(b+c)  
if (node->value == "+" && общий множитель найден) {

создаем новое поддерево: Node("\*", a, Node("+", b, c));  
}

Аналогично для - **Дополнительные оптимизации**:

-(-a) → a

0 \* a → 0

1 \* a → a

a + 0 → a

a - 0 → a

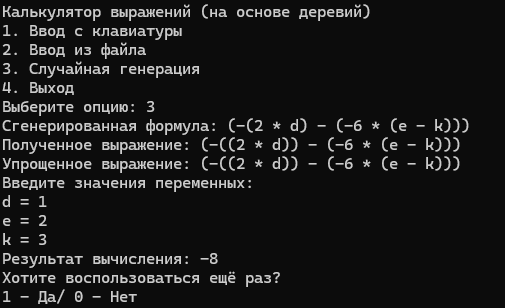
1. **Генерация случайных выражений**:

Рекурсивная генерация с контролем глубины

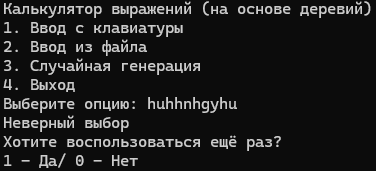
Случайный выбор: числа (1-9), переменные (a-z), операции

Тестирование

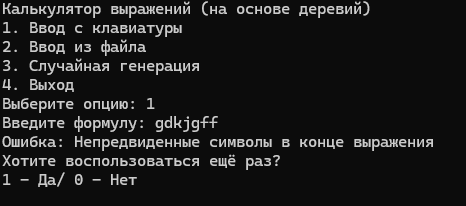
1. Штатная работа программы



1. Попытка неверного ввода



1. Попытка ввода неправильного выражения



Ошибки при вводе данных (буквы, символы, числа). Защита от дурака.

Проверялись как числа, так и операнды для ввода.

Также проверки на не подходящие для программы числа, символы и буквы

Защита от дурака работает корректно

# Код программы:

<https://github.com/Uzaki-chan/IKM_Formula_lite_20>