**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра САПР.**

**ОТЧЕТ**

**по курсовой работе**

**по дисциплине «Алгоритмы и структуры данных»**

| Студентка гр. 1301 |  | Ахметзянов Д.А. |
| --- | --- | --- |
| Преподаватель |  | Давидчук А.Г. |

Санкт-Петербург

2022

Оглавление

[Формулировка задания.](#_heading=h.gjdgxs) **3**

[Описание алгоритма.](#_heading=h.30j0zll) **3**

[Пример выполнения.](#_heading=h.3znysh7) **4**

# Формулировка задания.

Реализовать вычисление арифметических выражений с

бинарными операциями {+,-,\*,/,^} и числовыми операндами,

записанных в префиксной/инфиксной/постфиксной записи.

# Описание алгоритма.

По выражению в зависимости от формы записи строится бинарное дерево, в котором в листьях хранятся операнды, а в остальных узлах операторы.

1. Префиксная запись.

Если первое выражение в строке это число то узлу переданному в функцию присваивается его значение.

Иначе если в строке первым символом является оператор то создается узел для него создаются правый и левые потомки.

Для них функция вызывается рекурсивно с урезанным выражением (длина выражения обработанного выражения возвращается из функции).

1. Постфиксная запись.

Аналогично префиксной с отличием только в том, что обработка строки идет с конца.

1. Инфиксная запись.

Парсисится первый операнд, если после этого строка осталось пустой то возвращается узел со значением этого операнда, иначе парсится оператор, для него создается узел, первый операнд вставляется в его левого потомка, а правый получается из рекурсивного вызова функции с оставшейся частью выражения.

Для того чтобы учитывать порядок выполнения выражений при первом проходе обрабатываюстя только операторы + и -, то есть выражения вида 2 \* 2 будут помещены в узел как единое целое, затем для таких узлов происходит обработка операций \* и /, и в конце операции ^.

Для получения результата выражения, начиная с корня дерева у узлов вызывается метод computeValue, который применяет содержащийся в узле операнд к результатом того же метода своих левого и правого потомков.

# Пример выполнения.

