|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа № \_\_**2**\_\_**

**Дисциплина Конструирование компиляторов**

|  |  |
| --- | --- |
| **Тема Преобразования грамматик**  **Вариант №4**  **Студент \_Золотухин А.В.\_\_\_\_\_\_\_\_**  **Группа \_ИУ7-21М\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Ступников А.А.** |  |

Москва.

2025 г.

**Цель работы:** приобретение практических навыков реализации наиболее важных (но не всех) видов преобразований грамматик, чтобы удовлетворить требованиям алгоритмов синтаксического разбора.

**Задание**

Постройте программу, которая в качестве входа принимает приведенную КС-грамматику G = (Ν, Σ, P, S) и преобразует ее в эквивалентную КС-грамматику G' без левой рекурсии.

Вариант 4.

Устранение цепных правил. Определение. Правила вида A −> B, где A ∈ Ν и B ∈ Ν, будем называть цепными. Постройте программу, которая в качестве входа принимает произвольную КС-грамматику G = (Ν, Σ, P, S) без ε-правил и преобразует ее в эквивалентную КС-грамматику G' = (Ν, Σ, P', S) без ε-правил и без цепных правил.

Указания. Воспользоваться алгоритмом 2.11. [1]. При тестировании воспользоваться примером 2.24. [1].

**Результаты**

Входная грамматика

**Изображение выглядит как снимок экрана, Шрифт, текст, число

Контент, сгенерированный ИИ, может содержать ошибки.**

Грамматика без левой рекурсии

Изображение выглядит как текст, Шрифт, снимок экрана, число

Контент, сгенерированный ИИ, может содержать ошибки.

Грамматика без цепных правил

Изображение выглядит как текст, Шрифт, типография, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

**Контрольные вопросы**

1. *Как может быть определён формальный язык?*
   1. Простым перечислением слов, входящих в данный язык.
   2. Словами, порождёнными некоторой [формальной грамматикой](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)
   3. Словами, порождёнными [регулярным выражением](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F).
   4. Словами, распознаваемыми некоторым [конечным автоматом](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D0%B5%D1%87%D0%BD%D1%8B%D0%B9_%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82)
2. *Какими характеристиками определяется грамматика?*
   1. Σ – множество терминальных символов
   2. N – множество нетерминальных символов
   3. P – множество правил (слева – непустая последовательность терминалов/нетерминалов, содержащая хотя бы один нетерминал, справа – любая последовательность терминалов/нетерминалов)
   4. S – начальный символ из множества нетерминалов
3. Дайте описания грамматик по иерархии Хомского.
   1. Регулярные – контекстно-свободные, правила имеют вид A → Bγ (то есть левая часть только один нетерминал, справа – цепочка терминалов, может быть в конце/начале только один нетерминал)
   2. Контекстно-свободные – правая часть любая
   3. Контекстно-зависимые – в левой/правой часть могут быть цепочки, присутствующие в обеих частях, нужны для того, чтобы сработало правило
   4. Неограниченные – всё, что угодно может находиться в левой/правой частях.
4. *Какие абстрактные устройства используются для разбора грамматик?*
   1. Распознающие грамматики – устройства (алгоритмы), которым на вход подается цепочка языка, а на выходе устройство печатает «Да», если цепочка принадлежит языку, и «Нет» — иначе.
   2. Порождающие грамматики - вид устройств, использующийся для порождения цепочек языков по требованию.
5. *Оцените временную и емкостную сложность предложенного вам алгоритма*
   1. *O(* – временная сложность
   2. – ёмкостная сложность

**Выводы**

Были изучены и реализованы такие алгоритмы как: алгоритм устранения левой рекурсии и цепных правил.

**Список литературы**

1. АХО А., УЛЬМАН Дж. Теория синтаксического анализа, перевода и компиляции: В 2-х томах. Т.1.: Синтаксический анализ. - М.: Мир, 1978.
2. АХО А.В, ЛАМ М.С., СЕТИ Р., УЛЬМАН Дж.Д. Компиляторы: принципы, технологии и инструменты. – М.: Вильямс, 2008