Министерство образования и науки Кыргызской Республики

Кыргызско-Германский институт прикладной информатики

Лабораторная работа №1

по теме «Распознавание типов формальных языков и грамматик»

Выполнила: студентка группы AIN-1-22

Тахирова К.

Проверила:

Кошоева Б. Б.

Оглавление

[**1** **Цель** 3](#_Toc178538004)

[**2** **Постановка задачи** 3](#_Toc178538005)

[**3** **Составление грамматики** 3](#_Toc178538006)

[**4** **Определение типа формальной грамматики и языка** 4](#_Toc178538007)

[**5** **Разработка программное средство, распознающее тип введенной пользователем грамматики по классификации Хомского** 5](#_Toc178538008)

[5.1 Описание программы GrammarClassifier 5](#_Toc178538009)

[5.1.1 Цель 5](#_Toc178538010)

[5.1.2 Основные компоненты 5](#_Toc178538011)

[5.1.3 Пример вывода программы 8](#_Toc178538012)

[**6** **Вывод** 9](#_Toc178538013)

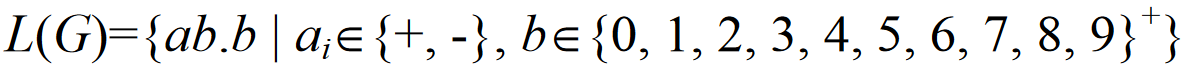
# **Цель**

* закрепить понятия «алфавит», «цепочка», «формальная грамматика» и «формальный язык», «выводимость цепочек», «эквивалентная грамматика»;
* сформировать умения и навыки распознавания типов формальных языков и грамматик по классификации Хомского, построения эквивалентных грамматик.

# **Постановка задачи**

При выполнении лабораторной работы следует реализовать следующие действия:

1. составить грамматику, порождающую формальный язык, заданный в соответствии с вариантом;
2. определить тип формальной грамматики и языка по классификации Хомского;
3. разработать программное средство, распознающее тип введенной пользователем грамматики по классификации Хомского.

Индивидуальное задание из таблицы 1.1 №12. Формальный язык: 

# **Составление грамматики**

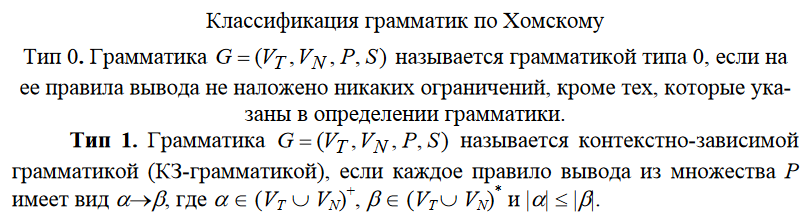
Общий вид формальной грамматики: G = (, , P, S),

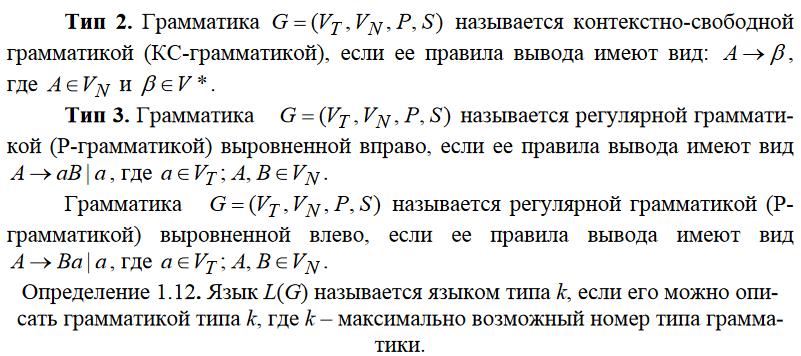
где:

* = {S, A, B}
* = {+, −, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, .}
* P:
  1. S → AB.B
  2. A → + ∣ −
  3. B → bB ∣ ε
  4. b → 0 ∣ 1 ∣ 2 ∣ 3 ∣ 4 ∣ 5 ∣ 6 ∣ 7 ∣ 8 ∣ 9

или G = ({+, −, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, .}, {S, A, B}, P, S)

# **Определение типа формальной грамматики и языка**





Грамматика, заданная правилами: G = ({+, −, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, .}, {S, A, B}, P, S) относится к **контекстно-зависимой** **грамматике (Тип 1)** по следующим причинам:

1. ***Форма правил:* в этой грамматике каждая продукция (правило) имеет вид** α→β, где α и β  **являются строками, состоящими из терминалов и нетерминалов.**
2. *Контекстно-зависимые правила:* в соответствии с определением контекстно-зависимой грамматики, для каждого правила α→β выполняется условие, что ∣α∣ ≤ ∣β∣. Например, в правиле B→bB длина правой части больше или равна длине левой части, что подтверждает, что грамматика контекстно-зависимая.
3. *Наличие ε-правил:* правило B→ε допустимо в контекстно-зависимых грамматиках, поскольку оно не нарушает основное правило ∣α∣ ≤ ∣β∣.
4. *Сложность:* данная грамматика допускает конструкции, которые выходят за рамки регулярной грамматики, что также подтверждает её принадлежность к контекстно-зависимым.

***Заключение.* Таким образом, грамматика описывается как контекстно-зависимая (Тип 1), так как она удовлетворяет всем необходимым условиям, установленным для данного типа грамматик по Хомскому.**

# **Разработка программное средство, распознающее тип введенной пользователем грамматики по классификации Хомского**

## Описание программы GrammarClassifier

### Цель

Программа предназначена для ввода правил грамматики и их классификации по типам согласно иерархии Хомского. Она может определить, к какому из четырех типов принадлежит заданная грамматика:

* Регулярная (Тип 3)
* Контекстно-свободная (Тип 2)
* Контекстно-зависимая (Тип 1)
* Рекурсивно-неограниченная (Тип 0)

### Основные компоненты

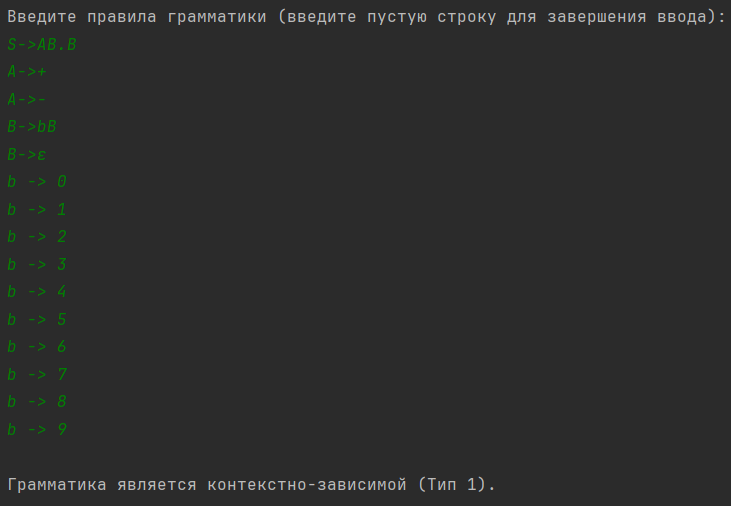
1. **Метод main**:
   * Запрашивает у пользователя ввод правил грамматики.
   * Хранит введенные правила в списке.
   * Вызывает метод determineGrammarType для классификации грамматики.
   * Выводит результат классификации на консоль.
2. **Метод determineGrammarType**:
   * Принимает список правил грамматики в виде ArrayList<String>.
   * Инициализирует три булевых переменных для отслеживания, является ли грамматика регулярной, контекстно-свободной или контекстно-зависимой.
   * Проходит по каждому правилу, проверяя его формат и соответствие условиям для каждой категории грамматик:
     + **Тип 3 (Регулярная грамматика)**: проверяет, что левая часть содержит один нетерминал и что правая часть соответствует условиям (например, A → aB или A → a).
     + **Тип 2 (Контекстно-свободная грамматика)**: проверяет, что левая часть содержит один нетерминал и правая часть может содержать любые терминалы и нетерминалы.
     + **Тип 1 (Контекстно-зависимая грамматика)**: проверяет, что длина правой части не меньше длины левой, кроме случаев с пустыми правилами.
   * В конце метод возвращает числовой код, соответствующий типу грамматики.

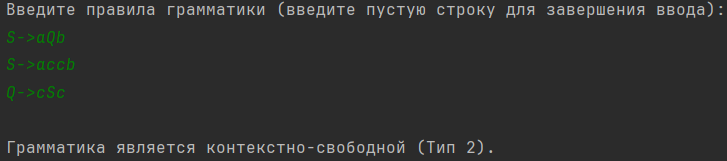
import java.util.ArrayList;  
import java.util.Scanner;  
  
public class GrammarClassifier {  
  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 ArrayList<String> rules = new ArrayList<>();  
  
 System.*out*.println("Введите правила грамматики (введите пустую строку для завершения ввода):");  
  
 // Ввод правил грамматики  
 while (true) {  
 String rule = scanner.nextLine();  
 if (rule.isEmpty()) {  
 break;  
 }  
 rules.add(rule);  
 }  
  
 // Определение типа грамматики  
 int grammarType = *determineGrammarType*(rules);  
  
 // Вывод результата  
 switch (grammarType) {  
 case 3:  
 System.*out*.println("Грамматика является регулярной (Тип 3).");  
 break;  
 case 2:  
 System.*out*.println("Грамматика является контекстно-свободной (Тип 2).");  
 break;  
 case 1:  
 System.*out*.println("Грамматика является контекстно-зависимой (Тип 1).");  
 break;  
 case 0:  
 System.*out*.println("Грамматика является рекурсивно-неограниченной (Тип 0).");  
 break;  
 default:  
 System.*out*.println("Не удалось определить тип грамматики.");  
 }  
 }

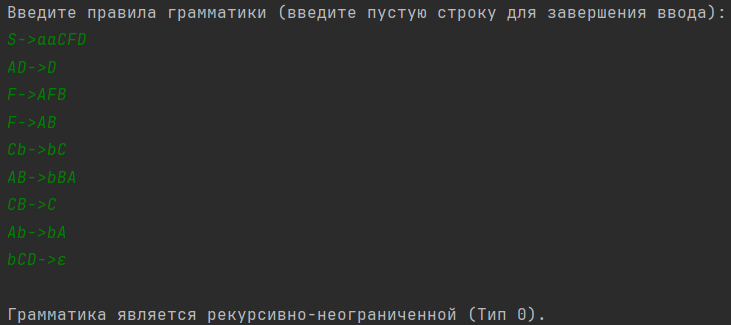
public static int determineGrammarType(ArrayList<String> rules) {  
 boolean isType3 = true; // Регулярная грамматика  
 boolean isType2 = true; // Контекстно-свободная грамматика  
 boolean isType1 = true; // Контекстно-зависимая грамматика  
  
 for (String rule : rules) {  
 String[] parts = rule.split("->");  
 if (parts.length != 2) {  
 System.*out*.println("Неправильный формат правила: " + rule);  
 return -1;  
 }  
  
 String left = parts[0].trim();  
 String right = parts[1].trim();  
  
 // Проверка на регулярную грамматику (Тип 3)  
 if (left.length() != 1 || !Character.*isUpperCase*(left.charAt(0))) {  
 isType3 = false; // Нетерминал слева должен быть один  
 }  
 if (right.length() > 2 || (right.length() == 2 && !Character.*isLowerCase*(right.charAt(1)))) {  
 isType3 = false; // Тип 3: A -> aB или A -> a  
 }  
  
 // Проверка на контекстно-свободную грамматику (Тип 2)  
 if (left.length() != 1 || !Character.*isUpperCase*(left.charAt(0))) {  
 isType2 = false; // Слева должен быть один нетерминал  
 } else {  
 // Проверяем, что правые части могут содержать любые терминалы и нетерминалы  
 for (char ch : right.toCharArray()) {  
 if (!Character.*isLowerCase*(ch) && !Character.*isUpperCase*(ch) && ch != 'ε' && ch != '.') {  
 isType2 = false; // Правые части должны содержать только терминалы и нетерминалы  
 }  
 }  
 }  
  
 // Проверка на контекстно-зависимую грамматику (Тип 1)  
 if (right.length() < left.length() && !right.equals("ε")) {  
 isType1 = false; // Длина правой части должна быть >= длины левой части (кроме ε)  
 }

/ Проверка на наличие пустых правил  
 if (right.equals("ε")) {  
 if (left.length() != 1) {  
 isType1 = false; // Пустые правила допустимы только для Типа 0  
 }  
 isType2 = false; // Пустые правила недопустимы для контекстно-свободной грамматики  
 }  
 }  
  
 // Итоговая классификация  
 if (isType3) return 3;  
 if (isType2) return 2;  
 if (isType1) return 1;  
 return 0; // Если не соответствует более строгим типам, то это грамматика типа 0  
 }  
  
}

### Пример вывода программы







# **Вывод**

В данной лабораторной работе была разработана программа на языке Java для классификации грамматик по типам Хомского. Программа анализирует заданные пользователем правила грамматики и определяет её тип в соответствии с иерархией, включающей:

1. **Регулярные грамматики (Тип 3)** – правила, соответствующие формату A → aB или A → a, где A и B – нетерминалы, а a – терминал.
2. **Контекстно-свободные грамматики (Тип 2)** – правила вида A → β, где A – один нетерминал, а β может содержать любую комбинацию терминалов и нетерминалов.
3. **Контекстно-зависимые грамматики (Тип 1)** – правила вида α → β, где длина правой части правила не меньше длины левой.
4. **Рекурсивно-неограниченные грамматики (Тип 0)** – правила, не имеющие ограничений на длину левой и правой части, за исключением условия, что левая часть не может быть пустой.