Санкт-Петербургский государственный университет

Факультет прикладной математики - процессов управления

# Проект №2 по КА

Разработал:

студент 233 группы Докиенко Д.А.

Проверил:

д.ф-м.н., проф. Матросов А.В.

Санкт-Петербург

2018

**1. Постановка задачи**

В ходе данной работы необходимо:

1. Задать порождающую грамматику заданного языка
2. Построить по грамматике пункта a) распознающий НКА с магазинной памятью (МП НКА)
3. Представить программу (код и исполняемый файл), распознающую цепочки заданного языка на основе программной реализации МП НКА
4. Представить описание работы программы пункта c)
5. Разработать тесты для программы пункта c)

4 вариант задания:

Алфавит:

Язык:

**2. Порождающая грамматика и МП НКА**

Зададим порождающую грамматику Хомского:

, где

* – конечное множество нетерминальных символов
* – конечный алфавит терминальных символов
* – конечное множество продукций
* – начальный нетерминал

Графическое изображение МП НКА:

Также представим более формальное описание МП НКА:

, где

* – конечное множество состояний
* – конечное множество входных символов
* – конечный магазинный алфавит
* – функция переходов (см. рисунок)
* – начальное состояние
* – «маркер дна» магазина
* – множество допускающих состояний

**3. Программа**

Данная программа написана на языке С++.

Код:

#include <iostream>

#include <string>

#include <stack>

// Проверка на соответствие языку введенной строки

// Возвращает позицию первого несоответствия языку

// Возвращает позицию после конца строки, если

// в строке не содержатся все необходимые символы,

// например: 110000111000

// При полном соответствии языку возвращает -1

int isLineRelevant(std::string &);

int main() {

// Ввод строки

std::string line;

std::cout << "Enter string:" << std::endl;

std::getline(std::cin, line);

// Вывод результата

int position = isLineRelevant(line);

if (position == -1) std::cout << "The string belongs to the language" << std::endl;

else std::cout << "Position of the first error: " << position << std::endl;

// Пауза после окончания работы программы

system("pause");

return 0;

}

int isLineRelevant(std::string &e\_line) {

// Магазин автомата

std::stack<char> machineShop;

machineShop.push('A');

machineShop.push('A');

// Посимвольный разбор строки

for (size\_t i = 0, length = e\_line.length(); i < length; i++)

switch (e\_line[i]) {

case '1':

// 1, идущая после 1^(n)0^(2n)1^(m)0^(2m)

if (machineShop.empty()) return i;

// Если в стеке осталась В, а пришла 1

// это значит, что количество нулей не

// соответствует правилам языка

if (machineShop.top() == 'B') return i;

// Переход 1,A/ABB

machineShop.pop();

machineShop.push('B');

machineShop.push('B');

machineShop.push('A');

break;

case '0':

// 0, идущий после 1^(n)0^(2n)1^(m)0^(2m)

if (machineShop.empty()) return i;

// Переход e,A/e, где е-эпсилон

while (machineShop.top() == 'A') {

machineShop.pop();

// Если 0 стоит первым или нулей больше

// чем ожидалось исходя из количества 1

if (machineShop.empty()) return i;

}

// Переход 0,В/e, где е-эпсилон

machineShop.pop();

break;

default:

return i;

}

// Переход e,A/e, где е-эпсилон

while (!machineShop.empty()) {

if (machineShop.top() == 'B') return e\_line.length();

machineShop.pop();

}

// Строка соответствует языку

return -1;

}

4. Описание работы

Функция int isLineRelevant(std::string &) имитирует работу МП НКА, создавая стек (магазин) и разбирая строку посимвольно.

Ход работы программы:

1. Считываем строку, которую необходимо обработать в функции

int main().

1. Вызываем функцию int isLineRelevant(std::string &), которая возвращает позицию первого несоответствия языку. Если строка соответствует языку, то она возвращает -1.
2. Выводим результат.

5. Тесты

|  |  |
| --- | --- |
| (пустая строка) | The string belongs to the language |
| 100110000 | The string belongs to the language |
| 111000000 | The string belongs to the language |
| 1001001 | Position of the first error: 7 |
| 1001000 | Position of the first error: 7 |
| 100 | The string belongs to the language |
| 10 | Position of the first error: 3 |
| 10a | Position of the first error: 3 |
| 100100a | Position of the first error: 7 |
| 0100110000 | Position of the first error: 1 |
| 100100100 | Position of the first error: 7 |
| 111000000100 | The string belongs to the language |
| 1000100 | Position of the first error: 4 |
| 1 | Position of the first error: 2 |
| 0 | Position of the first error: 1 |
| 110000 | The string belongs to the language |
| 100110000 | The string belongs to the language |
| 11111111111 | Position of the first error: 12 |
| 00000000000 | Position of the first error: 1 |
| 001000011 | Position of the first error: 1 |