*Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования*

|  |  |
| --- | --- |
| Gerb-BMSTU_01 | ***«Московский государственный технический университет  имени Н.Э. Баумана (национальный исследовательский институт)»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии

**Отчёт**

**по домашнему заданию №2**

**Дисциплина: Анализ Алгоритмов**

**Тема работы: Поиск слов в тексте**

Студентки гр. ИУ7-51б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сушина А.Д.**

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Волкова Л.Л.**

Москва, 2019г

Оглавление

[Введение 3](#__RefHeading___Toc2616_2744037635)

[1. Аналитическая часть 4](#__RefHeading___Toc2618_2744037635)

[1.1. Постановка задачи 4](#__RefHeading___Toc2620_2744037635)

[1.2 Регулярные выражения 4](#__RefHeading___Toc2622_2744037635)

[1.3 Конечный автомат 4](#__RefHeading___Toc2624_2744037635)

[3. Технологическая часть 5](#__RefHeading___Toc2626_2744037635)

[3.1. Требования к программному обеспечению 5](#__RefHeading___Toc5572_731354195)

[3.2. Средства реализации 5](#__RefHeading___Toc5574_731354195)

[3.3. Листинг кода 5](#__RefHeading___Toc5576_731354195)

[Заключение 7](#__RefHeading___Toc2628_2744037635)

# [Введение](#__RefHeading___Toc1989_2085773663)

Зачастую перед пользователем стоит задача поиска вхождений каких-либо специфичных строк в тексте большого объёма, и его физический просмотр глазами занял бы длительное время, что также не гарантирует отсутствие ошибок. Программные средства, решающие эту задачу, значительно экономят время.

Решить эту задачу программно можно двумя способами: с помощью конечного автомата или регулярных выражений.

Конечный автомат — абстрактный автомат, число возможных внутренних состояний которого конечно. С помощью конечных автоматов можно успешно решать обширный класс задач.

Регулярные выражения - это шаблоны используемые для сопоставления последовательностей символов в строках. Язык регулярных выражений предназначен специально для обработки строк.

Целью данной работы является осуществление поиска числительных с наращениями окончаний.

Для достижения этой цели поставлены следующие задачи:

1. Изучить возможности регулярных выражений и конечных автоматов.

2. Реализовать алгоритмы поиска слов в тексте.

# [1. Аналитическая часть](#__RefHeading___Toc2492_614358252)

В данном разделе будет описан алгоритм решения задачи регулярными выражениями и конечными автоматами.

## 1.1. Постановка задачи

На вход поступает текст. Требуется найти все числительные с наращениями. Под это определение подходят все конструкции вида число-окончание, где число — это любое число, записанное с помощью цифр, а окончание одно из множества [ й, я, м, ю, е, х, а, и, го, му, мя, ми].

## 1.2 Регулярные выражения

Регулярное выражение состоит из трех частей: число, дефис и окончание. Число может быть представлено одной или несколькими цифрами. Дефис — это обязательная часть, он должен быть в любой соответствующей шаблону подстроке. Окончание выбирается из множества [й, я, м, ю, е, х, а, и, го, му, мя, ми].

## 1.3 Конечный автомат

Проще всего описать конечный автомат с помощью схемы состояний и переходов, представленной на рисунке 1, где последний отработанный символ характеризует состояние, а поданный автомату на текущем шаге характеризует переход. Работа автомата начинается с состояния begin. Если автомат попадает в состояние end, это значит, что была найдена строка нужного формата.

|  |
| --- |
| Рис 1. Схема конечного автомата |

# 2[. Технологическая часть](#__RefHeading___Toc2512_614358252)

## 2[.1. Требования к программному обеспечению](#__RefHeading___Toc2514_614358252)

На вход программе поступает текст, необходимо найти все числительные с наращением окончания.

## 2[.2. Средства реализации](#__RefHeading___Toc2516_614358252)

Для реализации был выбран язык Python3, так как он хорошо подходит для работы с текстами и регулярными выражениями, а также знаком мне.

## 2[.3. Листинг кода](#__RefHeading___Toc2518_614358252)

Код программы находится на листингах 1-2.

|  |
| --- |
| Листинг 1. Поиск с помощью регулярного выражения   1. def regex(text): 2. pattern = r'(?:\t|\n| )\d+-(?:й|я|м|ю|е|х|а|и|го|му|мя|ми)\b' 3. print(pattern) 4. a = re.findall(pattern, text) |
| Листинг 2. Поиск с помощью конечного автомата   1. def fsm(text): 2. nums = '0123456789' 3. simple = 'йяюехаи' 4. after\_m = 'уяи' 5. dels = ' \t\n' 7. state = 'begin' 8. result = [] 9. i = 0 10. tlen = len(text) 11. begin = 0 12. for i in range(tlen): 13. c = text[i] 14. if state == 'begin': 15. if c in nums and (i == 0 or text[i-1] == ' ' or text[i-1] in dels): 16. begin = i 17. state = 'num' 18. elif state == 'num': 19. if c == '-': 20. state = '-' 21. elif not c in nums: 22. state = 'begin' 23. elif state == '-': 24. if c == 'м': 25. state = 'м' 26. elif c == 'г': 27. state = 'г' 28. elif c in simple: 29. state = 'end' 30. else: 31. state = 'begin' 32. elif state == 'м': 33. if c in after\_m: 34. state = 'end' 35. elif c == ' ' or c in dels: 36. result.append(text[begin:i]) 37. state = 'begin' 38. else: 39. state ='begin' 40. elif state == 'г': 41. if c == 'о': 42. state = 'end' 43. else: 44. state = 'begin' 45. elif state == 'end': 46. if c == ' ' or c in dels: 47. result.append(text[begin:i]) 48. state = 'begin' 49. else: 50. state = 'begin' |

# Заключение

Были изучены основные принципы работы регулярных выражений и конечных автоматов для поиска подстроки в тексте, а также разработана программа, решающая задачу поиска в тексте всех числительных с наращениями.