Государственное образовательное учреждение высшего профессионального образования

«Московский государственный технический университет имени Н. Э. Баумана»

(МГТУ им. Н.Э. Баумана)



ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ЛАБОРАТОРНАЯ РАБОТА №7

Отчет о лабораторной работе на тему:

«Поиск подстроки в строке»

Мирзоян Сергей

ИУ7-55

Москва 2018

**Оглавление**

Введение

1. Аналитическая часть 3

1.1 Постановка задачи 3

1.2 Описание алгоритмов 3

2. Конструкторская часть 4

2.1 Разработка алгоритма Кнута-Морриса-Пратта 4

2.2 Разработка алгоритма Бойера-Мура 4

3. Технологическая часть 7

3.1 Требования к программному обеспечению 7

3.2 Средства реализации 7

3.3 Листинг кода 7

4. Экспериментальная часть 9

4.1 Примеры работы 9

Заключение 10

**Введение**

В программах, предназначенных для редактирования текста, часто необходимо найти все фрагменты текста, совпадающие с заданным образцом. Обычно. текст - это редактируемый документ, а образец - искомое слово, введённое пользователем. Эффективные алгоритмы решения этой задачи(часто именуемой сопоставлением строк) могут сокращать время реакции текстовых редакторов на действия пользователя.

Цель работы: изучить и реализовать алгоритмы поиска подстрок.

Задачи:

1) реализовать алгоритмы Кнута-Морриса-Пратта и Бойера-Мура;

2) реализовать ПО, включающее данные алгоритм, которое будет принимать на вход параметры алгоритма;

3) провести исследования зависимости получаемого результата от входных параметров;

4) описать и обосновать полученные результаты в отчете о выполненной лабораторной работе.

**1. Аналитическая часть**

В данном разделе приведено теоретическое описание реализованных алгоритмов.

1.1 Постановка задачи

Реализовать 2 алгоритма поиска подстроки в строке: алгоритм Кнута-Морриса-Пратта и алгоритм Бойера-Мура.

1.2 Описание алгоритмов

Алгоритм поиска строки Бойера — Мура считается наиболее быстрым среди алгоритмов общего назначения, предназначенных для [поиска подстроки в строке](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B8%D1%81%D0%BA_%D0%BF%D0%BE%D0%B4%D1%81%D1%82%D1%80%D0%BE%D0%BA%D0%B8). Был разработан [Робертом Бойером](https://en.wikipedia.org/wiki/Robert_S._Boyer) и [Джеем Муром](https://en.wikipedia.org/wiki/J_Strother_Moore) в [1977 году](https://ru.wikipedia.org/wiki/1977_%D0%B3%D0%BE%D0%B4). Преимущество этого алгоритма в том, что ценой некоторого количества предварительных вычислений над шаблоном (но не над строкой, в которой ведётся поиск) шаблон сравнивается с исходным текстом не во всех позициях — часть проверок пропускаются как заведомо не дающие результата.

Общая оценка [вычислительной сложности](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9) современного варианта алгоритма Бойера — Мура — , если не используется таблица стоп-символов (смотрите ниже), и , если используется таблица стоп-символов, где  — длина строки, в которой выполняется поиск,  — длина шаблона поиска,  — алфавит, на котором проводится сравнение.

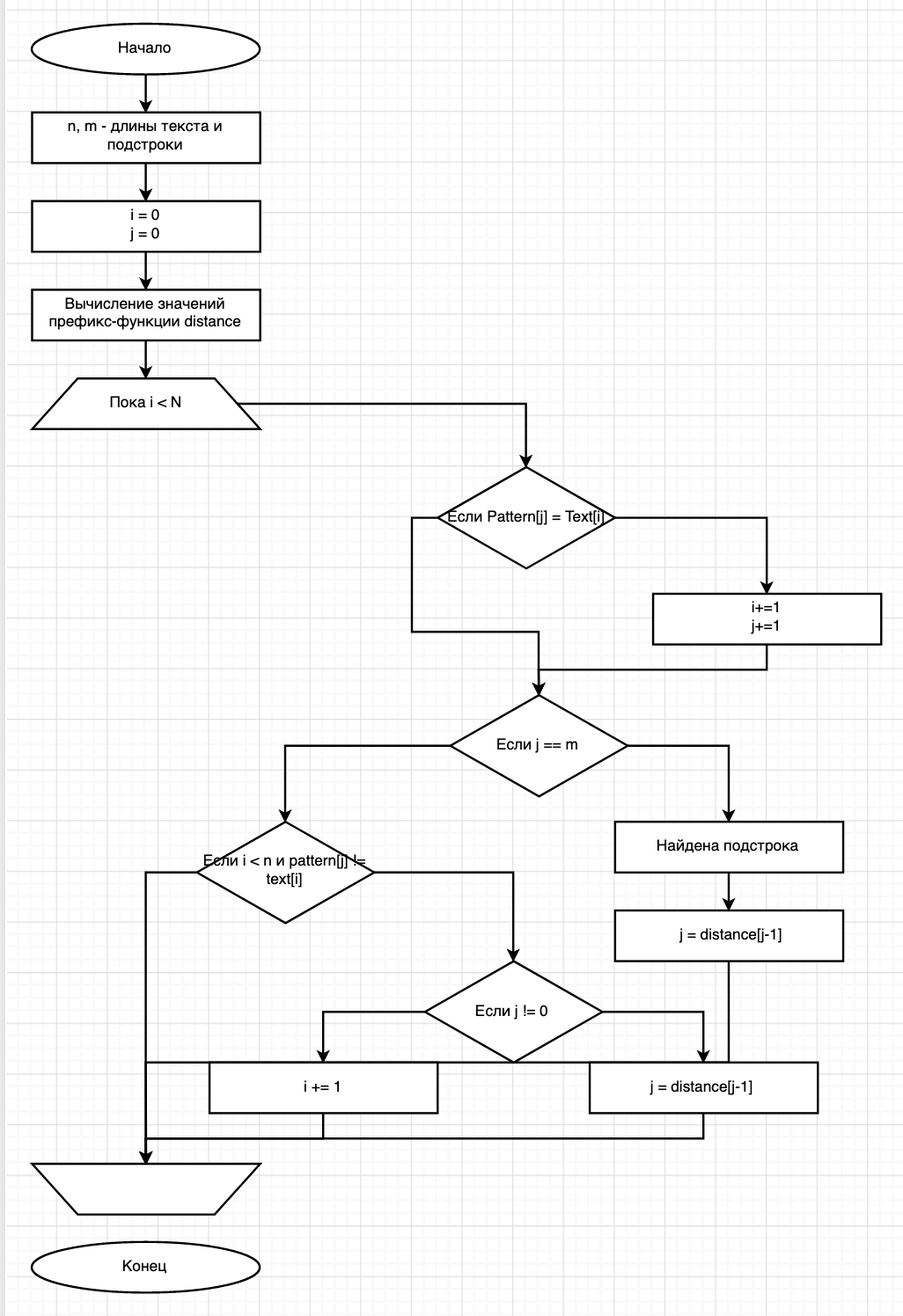
Алгоритм Кнута — Морриса — Пратта (КМП-алгоритм) — эффективный [алгоритм](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC), осуществляющий [поиск подстроки в строке](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B8%D1%81%D0%BA_%D0%BF%D0%BE%D0%B4%D1%81%D1%82%D1%80%D0%BE%D0%BA%D0%B8). Время работы алгоритма линейно зависит от объёма входных данных, то есть разработать [асимптотически более эффективный](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D1%8C) алгоритм невозможно.

Алгоритм был разработан [Д. Кнутом](https://ru.wikipedia.org/wiki/%D0%9A%D0%BD%D1%83%D1%82,_%D0%94%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%B4_%D0%AD%D1%80%D0%B2%D0%B8%D0%BD) и [В. Праттом](https://ru.wikipedia.org/w/index.php?title=%D0%9F%D1%80%D0%B0%D1%82%D1%82,_%D0%92%D0%BE%D0%BD&action=edit&redlink=1) и, независимо от них, [Д. Моррисом](https://ru.wikipedia.org/w/index.php?title=%D0%9C%D0%BE%D1%80%D1%80%D0%B8%D1%81,_%D0%94%D0%B6%D0%B5%D0%B9%D0%BC%D1%81&action=edit&redlink=1). Результаты своей работы они опубликовали совместно в [1977 году](https://ru.wikipedia.org/wiki/1977_%D0%B3%D0%BE%D0%B4).

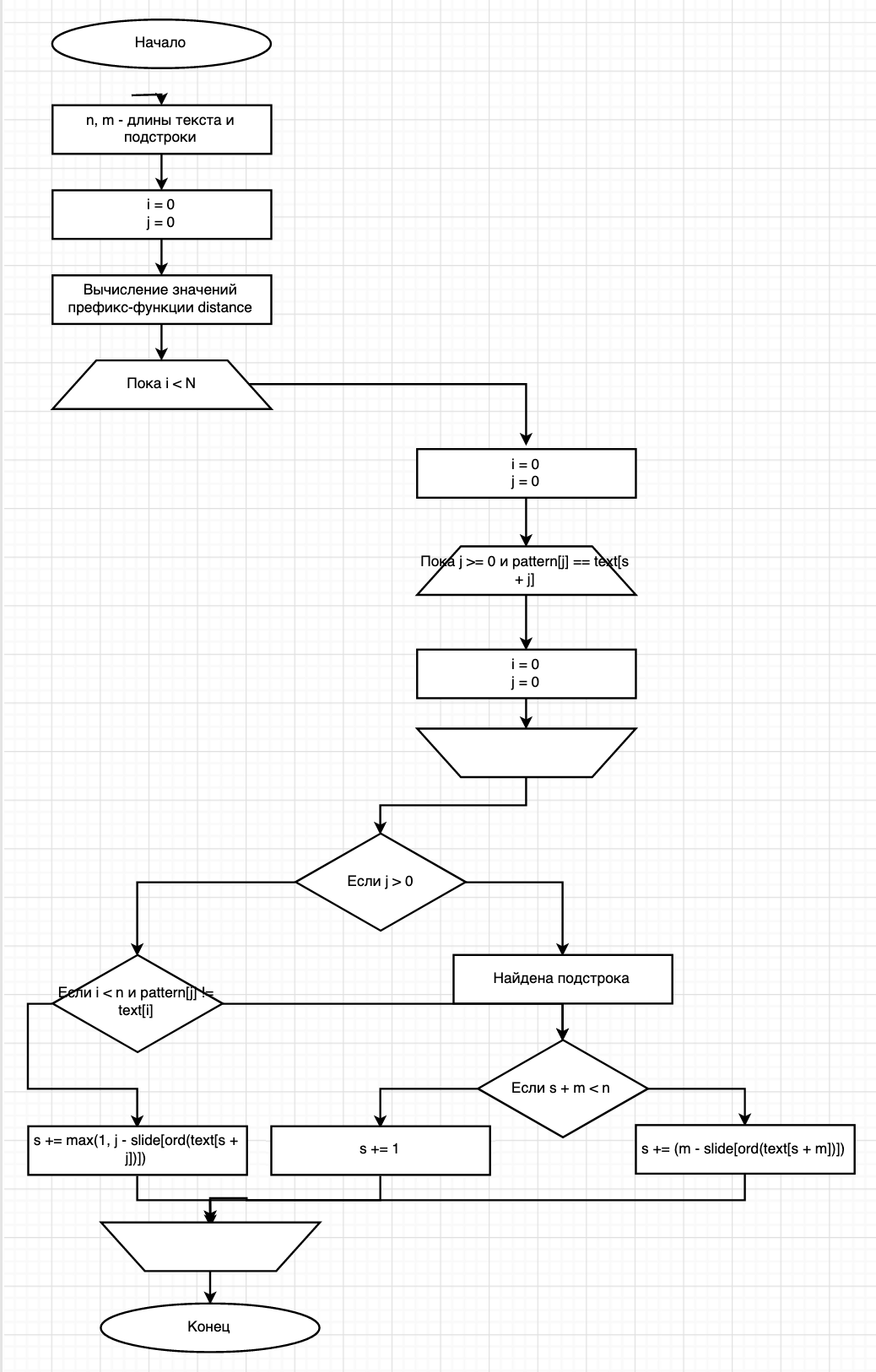
**2. Конструкторская часть**

В данном разделе приведена схема реализуемых в лабораторной работе алгоритмов.

2.1 Разработка алгоритма Кнута-Морриса-Пратта



2.2 Разработка алгоритма Бойера-Мура



**3. Технологическая часть**

В данном разделе представлены требования к разрабатываемому программному обеспечению, средства, использованные в процессе разработки для реализации поставленных задач, а также листинг кода программы.

3.1 Требования к программному обеспечению

Программное обеспечение должно работать на всех системах, поддерживающих Python 3.x версии.

3.2 Средства реализации

Программа реализовывалась на языке программирования Python. Данный язык был выбран из-за простоты использования в задачах подобного рода.

3.3 Листинг кода

Ниже приведен листинг кода для алгоритмов

Листинг 1. Код алгоритма Кнута-Морриса-Пратта:

1. **def KMPsearch(search\_string, pattern):**
2. search\_string.lower()
3. pattern.lower()
4. n = len(search\_string)
5. m = len(pattern)
6. shift\_array = compute\_array(pattern, m)
7. found = False
8. k = 0
9. p = 0
10. while k < n:
11. if pattern[p] == search\_string[k]:
12. k += 1
13. p += 1
14. if p == m:
15. found = True
16. print("KMP: match was found on index " + str(k - p) + "\n")
17. p = shift\_array[p - 1]
18. else:
19. if p == 0:
20. k += 1
21. if k > (n - m):
22. break
23. else:
24. p = shift\_array[p - 1]
25. if not found:
26. print("KMP: pattern not found")
27. **def compute\_array(pattern, m):**
28. shift\_array = [0] \* m
29. j = 0
30. i = 1
31. while i < m:
32. if pattern[i] == pattern[j]:
33. shift\_array[i] = j + 1
34. i += 1
35. j += 1
36. else:
37. if j == 0:
38. shift\_array[i] = 0
39. i += 1
40. else:
41. j = shift\_array[j - 1]
42. return shift\_array

Листинг 2. Код алгоритма Бойера-Мура:

1. **def BMHsearch(search\_string, pattern):**
2. search\_string.lower()
3. pattern.lower()
4. n = len(search\_string)
5. m = len(pattern)
6. shift\_array = compute\_shift(pattern, m)
7. found = False
8. i = 0
9. while i <= (n - m):
10. j = m - 1
11. while (j >= 0) and pattern[j] == search\_string[i + j]:
12. j -= 1
13. if j < 0:
14. found = True
15. print("BMH: match was found on index " + str(i) + "\n")
16. i += (shift\_array[search\_string[i + m]] if i + m < n else 1)
17. else:
18. if shift\_array.get(search\_string[i + j]) is None:
19. i += m
20. else:
21. i += shift\_array[search\_string[i + j]]
22. if not found:
23. print("BMH: pattern not found.\n")
24. **def compute\_shift(pattern, m):**
25. shift\_array = {}
26. value = m - 1
27. for i in range(m - 1):
28. shift\_array[(pattern[i])] = value
29. value -= 1
30. return shift\_array

**4. Экспериментальная часть**

В экспериментальном разделе представлены примеры работы разработанного программного обеспечения.

**4.1. Примеры работы**

Пример работы программы №1

text1 = "JHKLKLKJKJ"

pattern1 = "KLK"

kmp(pattern1, text1)Вывод:

Вывод: 2 4

Пример работы программы №2

text2 = "ABCDFDGDABCAAA"

pattern2 = "ABC"

bm(text2, pattern2)

Вывод: 0 8

**Заключение**

В процессе выполнения лабораторной работы были изучены два алгоритма поиска подстрок.

Во время разработки программного обеспечения в соответствии с поставленными требованиями были получены практические навыки реализации указанных алгоритмов на языке Python.