Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

# Лабораторная работа №4 по курсу «Дискретный анализ»

Студент: П.А. Земсков

Преподаватель: Н.К. Макаров

Группа: М8О-201Б

Дата: _____

Оценка: _____

Подпись: _____

Москва, 2025

# Содержание

# 1 Лабораторная работа №4

## Задача

Необходимо реализовать один из стандартных алгоритмов поиска образцов для указанного алфавита.
**Вариант алгоритма:** Z-функция.

# 2 Описание

Z-функция (англ. Z-function) от строки S и позиции x — это длина максимального префикса подстроки, начинающейся с позиции x в строке S, который одновременно является и префиксом всей строки S. Значение Z-функции от первой позиции не определено, поэтому его обычно приравнивают к нулю или к длине строки. Далее ниже приведен исходный код программы

# 3 Исходный код

**Общая идея:** Программа реализует поиск всех вхождений строки-образца в строке-тексте с использованием алгоритма Z-функции. Сначала образец и текст объединяются в одну строку через специальный символ-разделитель , чтобы избежать ложных совпадений. Далее для полученной строки вычисляется Z-функция, где для каждой позиции определяется длина наибольшего префикса, совпадающего с началом строки. Если значение Z-функции на позиции, соответствующей началу подстроки в тексте, равно длине образца, это означает полное совпадение образца с частью текста. Все такие позиции выводятся в порядке возрастания. Алгоритм работает за линейное время и эффективно находит все вхождения без необходимости использовать дополнительные структуры данных.

Листинг 1: Исходный код

```
#include <iostream>
#include <vector>
#include <string>
using namespace std;

vector<int> buildZ(const string &s)
{
    int n = s.size();
    vector<int> z(n);
    int l = 0, r = 0;
    for (int i = 1; i < n; ++i)
    {
        if (i <= r)
            z[i] = min(r - i + 1, z[i - l]);
        while (i + z[i] < n && s[z[i]] == s[i + z[i]])
            ++z[i];
        if (i + z[i] - 1 > r)
        {
            l = i;
            r = i + z[i] - 1;
        }
    }
    return z;
}

int main()
{
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    string text, pattern;
    getline(cin, text);
    getline(cin, pattern);

    string mask = pattern + '#' + text;
    vector<int> z = buildZ(mask);
    int length = pattern.size();

    for (int i = length + 1; i < z.size(); ++i)
    {
        if (z[i] == length)
        {
            cout << (i - length - 1) << '\n';
        }
    }

    return 0;
}
```

# 4 Консоль

Пример компиляции и демонстрация работы программы:

```
C:\Users\jocke\Documents\diskran\lab4> g++ --std=c++20 main.cpp -o main
C:\Users\jocke\Documents\diskran\lab4> ./main
abacaba
ab
0
4
```

# 5 Тест производительности

Вот примерный вывод и анализ производительности работы программы при поиске образца в тексте длиной $6 * 10^6$ :

```
C:\Users\jocke\Documents\diskran> g++ -std=c++20 lab4.cpp -o main
C:\Users\jocke\Documents\diskran> g++ -std=c++20 benchmark.cpp -o benchmark
C:\Users\jocke\Documents\diskran> .\benchmark
Z-function time: 55 ms
std::find time: 230 ms
```

### Результат:

В результате бенчмарка алгоритм Z-функции показал значительно большую производительность по сравнению с использованием стандартного метода std::string::find. Время работы Z-функции для текста длиной 6 млн символов составило примерно 55 мс, в то время как поиск с помощью std::find занял около 230 мс. Таким образом, Z-функция оказалась в 3-5 раз быстрее, что делает её более эффективным решением для поиска подстрок в больших текстах, особенно при многократных запросах.

# 6 Выводы

В ходе выполнения лабораторной работы было проведено сравнение эффективности двух методов поиска подстроки в строке: с использованием Z-функции и стандартного метода std::string::find. Результаты бенчмарка показали, что алгоритм Z-функции обладает значительно лучшей производительностью, особенно при работе с большими текстами. Время работы Z-функции для текста длиной 6 миллионов символов составило около 55 мс, в то время как метод std::find показал время около 230 мс, что в 3-5 раз медленнее. Это подтверждает, что Z-функция является более эффективным и быстрым инструментом для поиска подстрок, особенно при многократных запросах, и может быть предпочтительнее для работы с большими объёмами данных.

# 7 Список литературы


1. Кормен Т. Х., Лейзерсон Ч. И., Ривест Р. Л., Штайн К. *Алгоритмы: построение и анализ. 3-е изд.* – М.: Издательский дом «Вильямс», 2010.

2. Knuth D. E. *The Art of Computer Programming. Vol. 3: Sorting and Searching. 2nd ed.* – Addison-Wesley, 1998.

3. *Z-функция — ИТМО.* `https://neerc.ifmo.ru/wiki/index.php?title=Z-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F`.