**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра MOЭBM**

отчет

**по лабораторной работе №5**

**по дисциплине «Построение и анализ алгоритмов»**

**Тема:** «Алгоритм Ахо-Корасик»

|  |  |  |
| --- | --- | --- |
| Студентка гр. 6381 |  | Вероха В. Н. |
| Преподаватель |  | Филатов А.Ю. |

Санкт-Петербург

2018

**Цель работы**

1. Разработать программу, решающую задачу точного поиска набора образцов с помощью алгоритма Ахо-Корасик.
2. Используя реализацию точного множественного поиска, решить задачу точного поиска для одного образца с джокером.

**Условия задания**

1. Разработать программу,  решающую задачу точного поиска набораобразцов.  
   Вход:  
   Первая строка содержит текст *T*(1≤|*T*|≤100000).  
   Вторая - число *n* (1≤*n*≤3000), каждая следующая из *n* строк содержит шаблон из набора *P*={*p*1,…,*pn*}1≤|*pi*|≤75.  
   Все строки содержат символы из алфавита {*A*,*C*,*G*,*T*,*N*}.  
   Выход:  
   Все вхождения образцов из *P* в *T*.  
   Каждое вхождение образца в текст представить в виде двух чисел - *i* *p.*   
   Где *i* - позиция в тексте (нумерация начинается с 1), с которой начинается вхождение образца с номером *p* (нумерация образцов начинается с 1).  
   Строки выхода должны быть отсортированы по возрастанию, сначала номера позиции, затем номера шаблона.
2. Используя реализацию точного множественного поиска, решить задачу точного поиска для одного образца с *джокером*. В шаблоне встречается специальный символ, именуемого джокером (wild card), который "совпадает" с любым символом. По заданному содержащему шаблоны образцу *P* необходимо найти все вхождения *P* в текст *T*. Например, образец *ab??c?* с джокером ? встречается дважды в тексте *xabvccbababcax*.

Символ джокер не входит в алфавит, символы которого используются в *T*. Каждый джокер соответствует одному символу, а не подстроке неопределенной длины. В шаблоне входит хотя бы один символ не джокер, те шаблоны вида ??? недопустимы. Все строки содержат символы из алфавита {*A*,*C*,*G*,*T*,*N*}.  
Вход:  
Текст *T*(1≤|*T*|≤100000)  
Шаблон *P*(1≤|*P*|≤40)  
Символ джокера  
Выход:  
Строки с номерами позиций вхождений шаблона (каждая строка содержит только один номер).  
Номера должны выводиться в порядке возрастания.

**Описание алгоритма Ахо-Корасик**

Суть алгоритма заключена в использование структуры данных — бора и построения по нему конечного детерминированного автомата. Строится бор последовательным добавлением исходных строк. Изначально есть 1 вершина, корень - пустая строка. Добавление строки происходит так: начиная в корне, двигаемся по дереву, выбирая каждый раз ребро, соответствующее очередной букве строки. Если такого ребра нет, то мы создаем его вместе с вершиной. Так как процесс добавления строки может остановиться во внутренней вершине, то для каждой строки будем дополнительно хранить признак того является она строкой из условия или нет. Далее, строим конечный детерминированный автомат. Состояние автомата — это какая-то вершина бора. Переход из состояний осуществляется по 2 параметрам — текущей вершине v и символу ch по которому нам надо сдвинуться из этой вершины. Назовем суффиксной ссылкой вершины v указатель на вершину u, такую что строка u — наибольший собственный суффикс строки v, или, если такой вершины нет в боре, то указатель на корень. В частности, ссылка из корня ведет в него же. Если из текущей вершины есть ребро c символом v, то пройдем по нему, в обратном случаем пройдем по суффиксной ссылке и запустимся рекурсивно от новой вершины. Алгоритм завершится, когда мы дойдем до конца строки.

**Описание алгоритма точного поиска образца с джокером**

Алгоритм точного поиска образца с джокером является аналогом алгоритма Ахо-Корасик и использует его в своей работе. Строка-шаблон разделяется по символу-джокеру на строки, каждая из которых добавляется в бор.

**Описание основных функций**

Bor make\_Bor(int p, int c) *-* функция принимает на вход элемент и элемент-родитель и по этим данным создает новую вершину в боре.

void add(const string & patt, int number, vector<int> & pos, vector<Bor> & bor, int j) - добавление строки в бор.

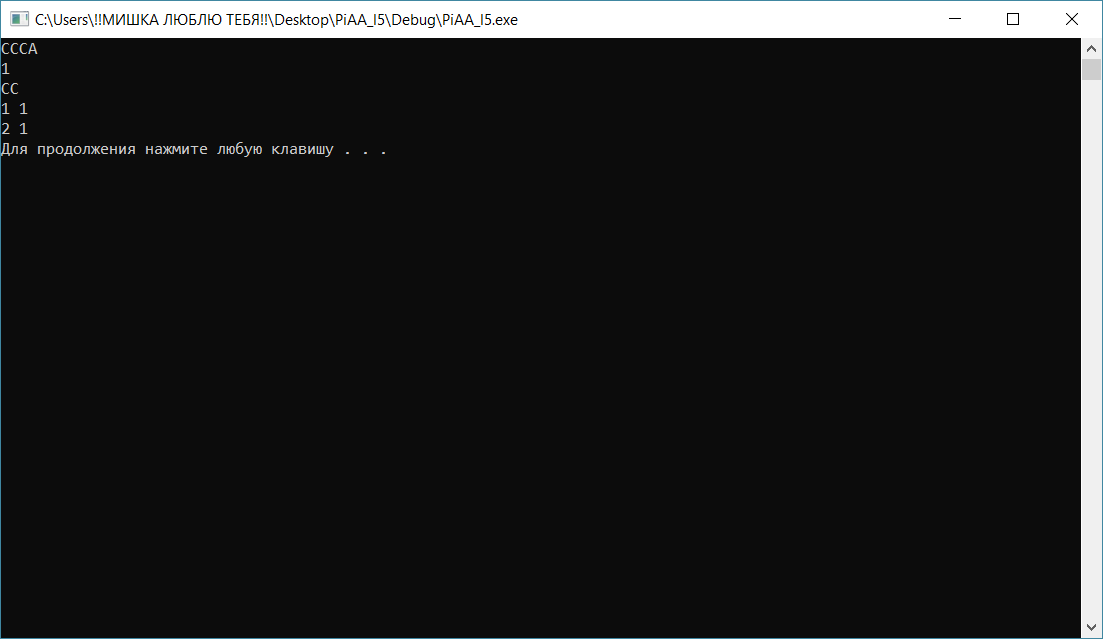
int search\_suffix(int v, vector<Bor> &bor) - функция находит суффиксную ссылку, если она не была найдена до этого, и возвращет ее.

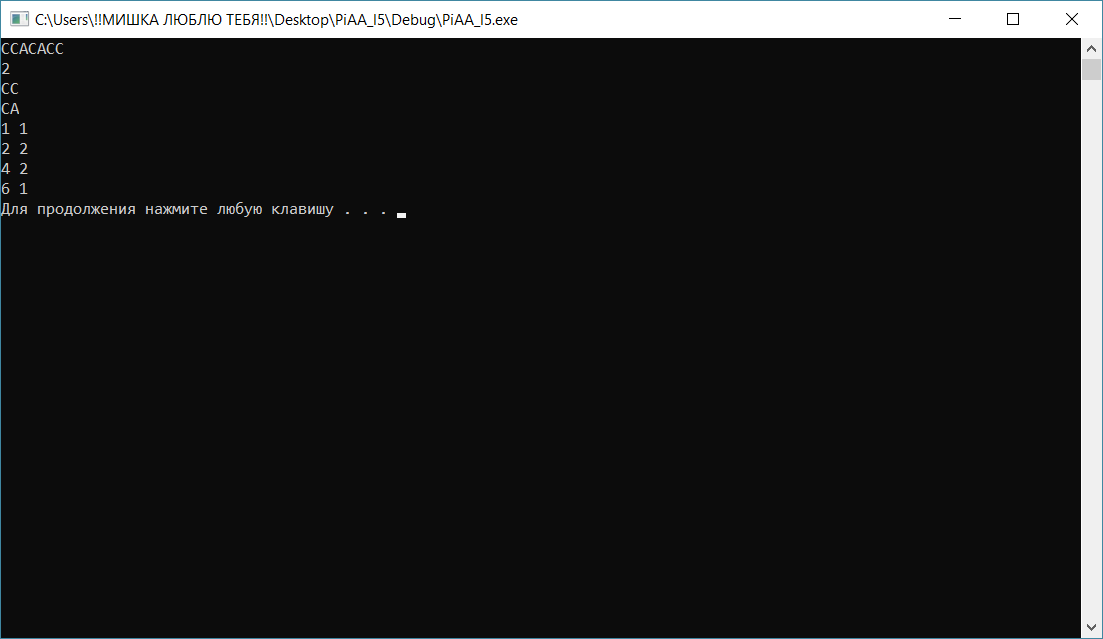
int search\_suffix(int v, int ch, vector<Bor> &bor) - функция находит и возвращает элемент в боре, в который следует переместиться из вершины *v* по символу *ch*.

void search(const string& s, vector<int> &pos, vector<Bor> &bor, int k) - функция находит позиции вхождения заданных шаблонов в строку.

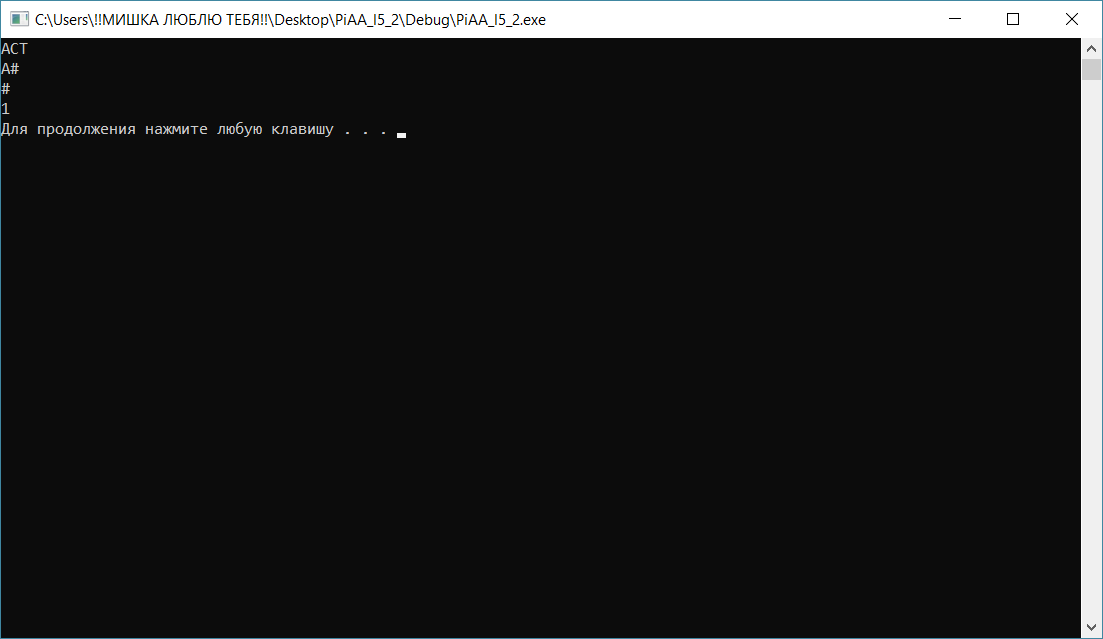
**Тестирование**

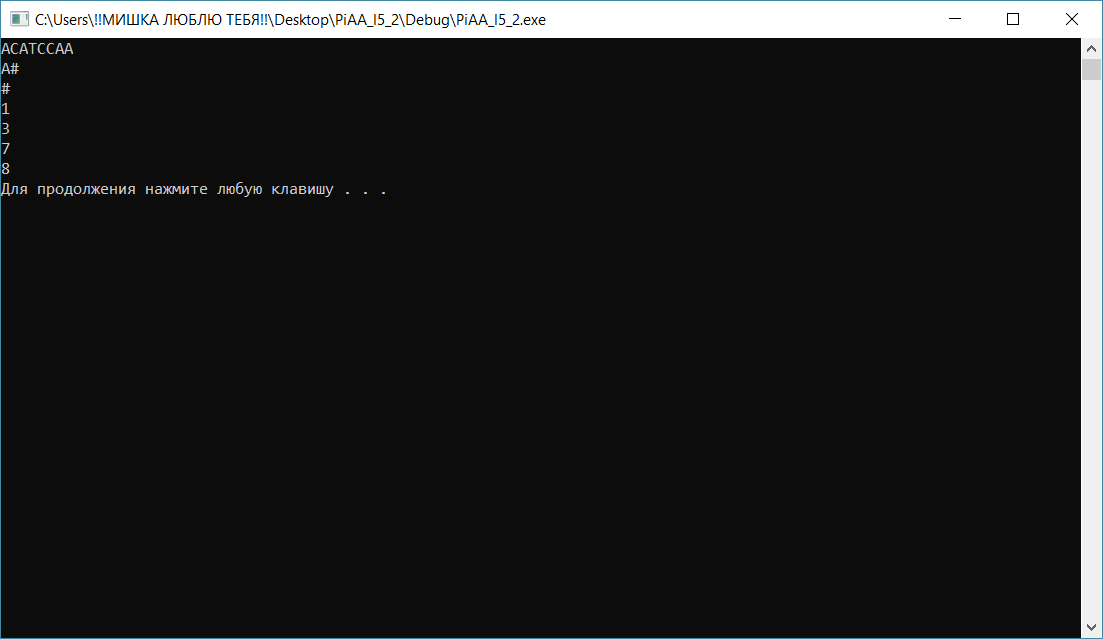
1. Алгоритм Ахо-Корасик





1. Алгоритма точного поиска образца с джокером





**Вывод**

В ходе выполнения лабораторной работы был реализован алгоритм, решающий задачу точного поиска набора образцов с помощью алгоритма Ахо-Корасик.