**{Из мудла}**

**Поиск в подстроке**

Алгоритм последовательного (прямого) поиска (SF-алгоритм)

(Straight Forward)

Задача. Дана строка S и образец P. Найти местоположение образца P в строке S.

Пpосматpиваем слово S слева напpаво до тех поp, пока не найдем совпадение пеpвой буквы подслова P с некотоpой буквой слова S.

Если пpосмотpены все буквы слова S, а искомого совпадения не найдено, то слово S не содеpжит подслова P. Если найдено пеpеходим к сопоставлению подслова X с соответствующим подсловом слова S. Если сопоставление не удалось, то пpодолжаем пpосмотp слова S до следующего совпадения пеpвой буквы слова X с некотоpой буквой слова S.

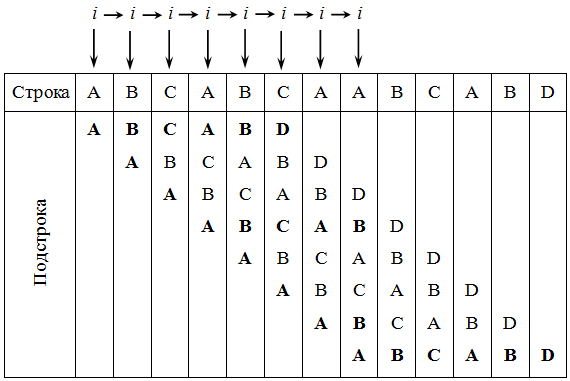
2. Обобщенный SF-алгоритм

Теперь приведём обобщённый SF-алгоритм - это SF-алгоритм для отыскания и замены (если необходимо) всех вхождений заданного подслова в слове на дpугое слово.

**Прямой поиск**

Данный *алгоритм* еще называется алгоритмом *последовательного поиска*, он является самым простым и очевидным.

Основная идея алгоритма прямым поиском заключается в посимвольном сравнении строки с подстрокой. В начальный момент происходит сравнение первого символа строки с первым символом подстроки, второго символа строки со вторым символом подстроки и т. д. Если произошло совпадение всех символов, то фиксируется факт нахождения подстроки. В противном случае производится сдвиг подстроки на одну позицию вправо и повторяется посимвольное сравнение, то есть сравнивается второй символ строки с первым символом подстроки, третий символ строки со вторым символом подстроки и т. д. ([рис. 39.1](https://intuit.ru/studies/courses/648/504/lecture/11468?page=1#image.39.1)) Символы, которые сравниваются, на рисунке выделены жирным. Рассматриваемые сдвиги подстроки повторяются до тех пор, пока конец подстроки не достиг конца строки или не произошло полное совпадение символов подстроки со строкой, то есть найдется *подстрока*.



**Рис. 39.1.**Демонстрация алгоритма прямого поиска

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

ifstream in("input.txt");

ofstream out("output.txt");

int main(){

string str; // строка

string substr; // подстрока которую ищем в строке

in >> str;

in >> substr;

int sl, ssl; // длины строк str и substr

sl = str.length();

ssl = substr.length();

for (int i = 0; i < sl - ssl + 1; i++){

for (int j = 0; j < ssl; j++){

if (substr[j] != str[i+j]) break; // break если символ подстроки

// несовпадает

// с символом строкой

else

if (j == ssl - 1 ){

cout << i << " "; // вывод индекса строки

}

}

}

}

Пример:

abababa

aba

вывод: 0 2 4

a b a b a b a

0 1 2 3 4 5 6

a b a

~~a~~

a b a

~~a~~

a b a

Поиск и замена строки с помощью

string::find(что найти, с какого индекса начать)

string::replace(с какого индекс начать, сколько символов изменить, на что изменить)

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

ifstream in("input.txt");

ofstream out("output.txt");

int main(){

string str; // строка

string substr; // подстрока которую ищем в строке

in >> str;

in >> substr;

int sl, ssl; // длины строк str и substr

sl = str.length();

ssl = substr.length();

int it = str.find(substr, 0); // ищет начальный индекс строки

// которая совпадает с подстрокой

// с 0-го индекса

while (it != string::npos){ // где npos означает отсутствие позиции

cout << it << " "; // вывод индекса строки

str.replace(it, ssl, "ABA"); // замена подстроки на ABA

it = str.find(substr, it); // ищет след. начальный индекс строки

}

cout << endl << str;

}

Пример:

abababa

aba

вывод: 0 4

a b a b a b a

0 1 2 3 4 5 6

a b a

A B A b a b a

0 1 2 3 4 5 6

a b a

A B A b A B A