*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

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

ФАКУЛЬТЕТ \_\_\_\_ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_Компьютерные Системы и сети (ИУ6)\_\_\_\_\_\_\_\_\_\_\_

**Отчет**

**по домашней работе № 2**

**Название домашней работы: Программирование на С++**

**Дисциплина: Объектно-ориентированное программирование**

Студент гр. ИУ6-22Б  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. П. Плютто**



(Подпись, дата) (И.О. Фамилия)



Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О. А. Веселовская**

(Подпись, дата) (И.О. Фамилия)

Москва, 2023

**Задание 1.**

Дана последовательность строк. Каждая строка состоит из слов, разделенных пробелами. Написать программу, обеспечивающую ввод строк и их корректировку. Корректировка заключается в удалении лишних пробелов и слов, состоящих из одного символа. Лишними считаются пробелы в начале и конце строки, а также более одного пробела между словами. Вывести на печать исходную и скорректированную последовательности строк. Реализовать как с использованием библиотечных функций работы со строками, так и с помощью собственной их реализации. Оценить время выполнения программы разными реализациями.

**Решение**

Так как необходимо выполнить задание двумя способами: с помощью стандартных библиотек и полностью своим решением, то для первого решения я воспользуюсь самым распространённым методом любой корректировки строк регулярными выражениями. Стоит заметить, что в C++ в стандартной библиотеке regax регулярные выражения компилируются, а не поддерживаются нативно, из-за этого скорость программы может уменьшаться. Для собственного решения я буду использовать стандартный цикл, а для того, чтобы не пропустить, например тройные пробелы в начале строки после каждого нахождения необходимой комбинации символов буду обнулять счетчик.

*#include* "iostream"

*#include* "regex"

*#include* <chrono>

using std::cin, std::cout,

std::regex, std::string,

std::regex\_replace, std::regex\_search;

string regexDesition(string str)

{

*auto* start\_time = std::chrono::steady\_clock::now();

regex re1("\\s[a-zA-Z1-9]\\s"), re2("\\s\\s"), re3("^\\s"), re4("\\s$");

*while* (regex\_search(str, re1) or

regex\_search(str, re2) or

regex\_search(str, re3) or

regex\_search(str, re4))

str =

regex\_replace(

regex\_replace(

regex\_replace(

regex\_replace(

str, re4, ""),

re3, ""),

re2, " "),

re1, " ");

*auto* end\_time = std::chrono::steady\_clock::now();

*auto* elapsed\_ns = std::chrono::duration\_cast<std::chrono::nanoseconds>(end\_time - start\_time);

std::cout << "Time:" << elapsed\_ns.count() << " ns\n";

*return* str;

}

*int* strSpace(*int* c)

{

*return* c == 32;

}

string strGetSub(string str, *int* start, *int* end)

{

string subStr;

*for* (*int* i = start; i < end; i++)

subStr = subStr + str[i];

*return* subStr;

}

string strDelete(string str, *int* start, *int* step)

{

*return* strGetSub(str, 0, start) +

strGetSub(str, start + step, str.length());

}

string myDesition(string str)

{

*auto* start\_time = std::chrono::steady\_clock::now();

*int* i = 0;

*while* (i < str.length())

{

*if* (strSpace(str[i]) and strSpace(str[i + 1]))

{

str = strDelete(str, i + 1, 1);

i = 0;

};

*if* (strSpace(str[i]) and strSpace(str[i + 2]))

{

str = strDelete(str, i + 1, 2);

i = 0;

};

*if* (i == 0 and strSpace(str[i]))

{

str = strDelete(str, i, 1);

i = 0;

};

*if* (i == str.length() - 1 and strSpace(str[i]))

{

str = strDelete(str, i, 1);

i = 0;

};

i++;

};

*auto* end\_time = std::chrono::steady\_clock::now();

*auto* elapsed\_ns = std::chrono::duration\_cast<std::chrono::nanoseconds>(end\_time - start\_time);

std::cout << "Time:" << elapsed\_ns.count() << " ns\n";

*return* str;

}

*int* main()

{

string str;

getline(cin, str);

cout << "Input:" << str << "\n\n";

cout << "My desition:\n"

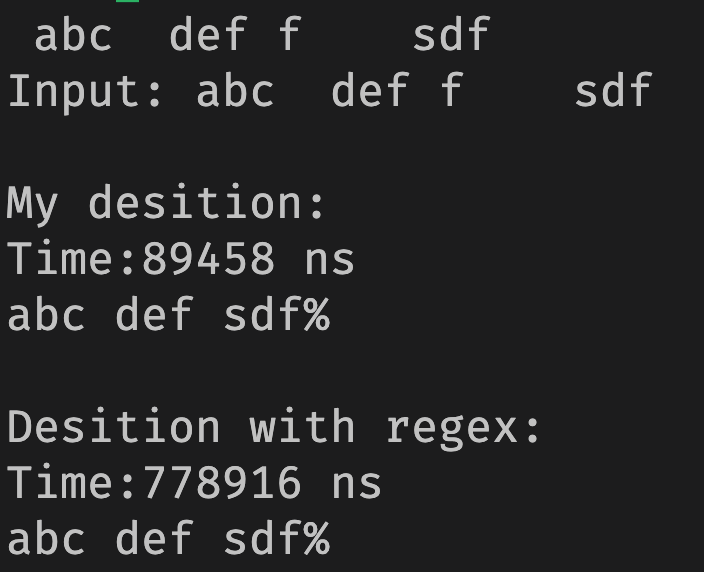
<< myDesition(str) << "%\n\n";

cout << "Desition with regex:\n"

<< regexDesition(str) << "%\n";

*return* 0;

}



Вывод: я научился изменять строки в C++