**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ   
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»**

**ХАБАРОВСКИЙ ИНСТИТУТ ИНФОКОММУНИКАЦИЙ (ФИЛИАЛ)**

**СРЕДНЕЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАНИЕ**

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №7

по МДК 01.01 Системное программирование

на тему

Строки. Работа со строками. Методы для работы со строками.

Выполнил: А.В. Медведев

Проверил: Н.А. Насонова

Группа: ПКС-320

2023

Составление программ.

Для начала работы подключим библиотеки для работы с основными функциями. И подключим пространство имён std.

#include <iostream>

#include <sstream>

#include <string>

#include <algorithm>

#include <vector>

#include <iomanip>

using namespace std;

iostream — заголовочный файл с классами, функциями и переменными для организации ввода-вывода.

algorithm – заголовочный файл в стандартной библиотеке языка программирования C++, включающий набор функций для выполнения алгоритмических операций.

string – заголовочный файл в стандартной библиотеке языка С++ содержаший тип данных строки и набор функций для работы с ней.

sstream – заголовочный файл с классами, функциями и переменными для организации работы со строками, через интерфейс потоков, в языке программирования C++.

vector – стандартный шаблон обобщённого программирования языка C++, реализующий динамический массив.

iomanip – библиотека, в которой содержится модификатор setw.

В методе main опишем использования русского алфавита.

setlocale(LC\_ALL, "Russian");

Дан текст. а) Подсчитать количество слов в данной строке.

#include <iostream>

#include <sstream>

int main() {

std::string text = "My own text Medvedev";

std::istringstream iss(text);

std::string word;

int word\_count = 0;

while (iss >> word) {

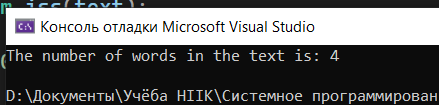
++word\_count;

}

std::cout << "The number of words in the text is: " << word\_count << std::endl;

return 0;

}



Подсчитать количество букв а в последнем слове данной строки.

#include <iostream>

#include <string>

using namespace std;

int main() {

string str;

getline(cin, str);

int count = 0;

int len = str.length();

for (int i = len - 1; i >= 0; i--) {

if (str[i] == ' ') {

break;

}

else if (str[i] == 'a') {

count++;

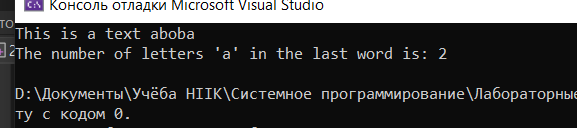
}

}

cout << "The number of letters 'a' in the last word is: " << count << endl;

return 0;

}



Найти количество слов, начинающихся с буквы б.

#include <iostream>

#include <string>

#include <sstream>

using namespace std;

int main() {

string str;

getline(cin, str);

stringstream ss(str);

string word;

int count = 0;

while (ss >> word) {

if (word[0] == 'b' || word[0] == 'B') {

count++;

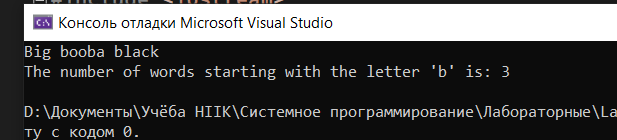
}

}

cout << "The number of words starting with the letter 'b' is: " << count << endl;

return 0;

}



Найти количество слов, у которых первый и последний символы совпадают между собой.

#include <iostream>

#include <string>

#include <sstream>

using namespace std;

int main() {

string str;

getline(cin, str);

stringstream ss(str);

string word;

int count = 0;

while (ss >> word) {

if (word[0] == word[word.length() - 1]) {

count++;

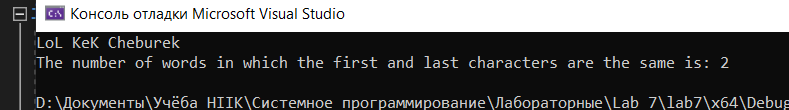
}

}

cout << "The number of words in which the first and last characters are the same is: " << count << endl;

return 0;

}



Найти длину самого короткого слова.

#include <iostream>

#include <string>

#include <sstream>

#include <climits>

using namespace std;

int main() {

string str;

getline(cin, str);

stringstream ss(str);

string word;

int minLength = INT\_MAX;

while (ss >> word) {

if (word.length() < minLength) {

minLength = word.length();

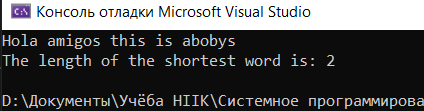
}

}

cout << "The length of the shortest word is: " << minLength << endl;

return 0;

}



Составить программу циклической перестановки букв в словах текста так, что i-я буква слова становится i+1-ой, а последняя - первой.

#include <iostream>

#include <string>

#include <sstream>

using namespace std;

int main() {

string str;

getline(cin, str);

stringstream ss(str);

string word;

while (ss >> word) {

int length = word.length();

char first = word[length - 1];

for (int i = length - 1; i > 0; i--) {

word[i] = word[i - 1];

}

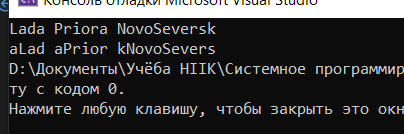
word[0] = first;

cout << word << " ";

}

return 0;

}



В каждом слове текста замените "а" на букву "е", если "а" стоит на четном месте, и заменить букву "б" на сочетание "ак", если "б" стоит на нечетном месте.

#include <iostream>

#include <string>

#include <sstream>

using namespace std;

int main() {

string str;

getline(cin, str);

stringstream ss(str);

string word;

while (ss >> word) {

for (int i = 0; i < word.length(); i++) {

if (word[i] == 'a' && i % 2 == 0) {

word[i] = 'e';

}

else if (word[i] == 'b' && i % 2 != 0) {

word.replace(i, 1, "ak");

i += 1;

}

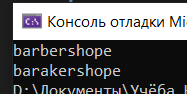
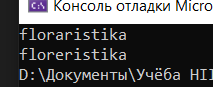
}

cout << word << " ";

}

return 0;

}

Дан текст, содержащий от 2 до 30 слов, в каждом из которых от 2 до 10 латинских букв; между соседними словами – не менее одного пробела. Напечатать все слова, отличные от последнего слова, предварительно преобразовав каждое из них по следующему правилу: 1) перенести первую букву в конец слова; 2) перенести последнюю букву в начало слова.

#include <iostream>

#include <string>

#include <sstream>

using namespace std;

int main() {

string str;

getline(cin, str);

stringstream ss(str);

string word;

int count = 0;

while (ss >> word) {

count++;

if (count == 1) {

continue;

}

char first = word[0];

char last = word[word.length() - 1];

word.erase(0, 1);

word.erase(word.length() - 1, 1);

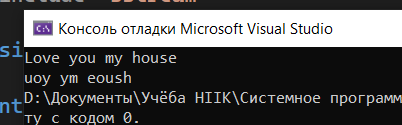
word = last + word + first;

cout << word << " ";

}

return 0;

}



Дан текст. Напечатать все слова, отличные от последнего слова, предварительно преобразовав каждое из них по следующему правилу: 1) оставить в слове только первые вхождения каждой буквы; 2) если слово нечетной длины, то удалить его среднюю букву

#include <iostream>

#include <string>

#include <sstream>

#include <unordered\_set>

using namespace std;

int main() {

string str;

getline(cin, str);

stringstream ss(str);

string word;

int count = 0;

while (ss >> word) {

count++;

if (count == 1) {

continue;

}

unordered\_set<char> seen;

string result = "";

for (int i = 0; i < word.length(); i++) {

if (seen.count(word[i]) == 0) {

result += word[i];

seen.insert(word[i]);

}

}

if (result.length() % 2 == 1) {

result.erase(result.length() / 2, 1);

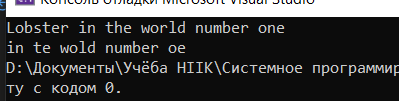
}

cout << result << " ";

}

return 0;

}



Составить таблицу слов данного текста, начинающихся с буквы "*А*", с указанием числа повторений каждого слова.

#include <iostream>

#include <string>

#include <sstream>

#include <map>

using namespace std;

int main() {

string str;

getline(cin, str);

stringstream ss(str);

string word;

map<string, int> table;

while (ss >> word) {

if (word[0] == 'A' || word[0] == 'a') {

table[word]++;

}

}

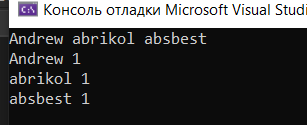
for (auto entry : table) {

cout << entry.first << " " << entry.second << endl;

}

return 0;

}



Составить программу для вычеркивания из слов текста всех букв, стоящих на нечетных местах после буквы "а".

#include <iostream>

#include <string>

#include <sstream>

using namespace std;

void crossOutLetters(string& word);

void crossOutLetters(string& word) {

bool foundA = false;

for (int i = 0; i < word.length(); i++) {

if (foundA && i % 2 == 1) {

word[i] = '-';

}

if (word[i] == 'a' || word[i] == 'A') {

foundA = true;

}

}

}

int main() {

string str;

getline(cin, str);

stringstream ss(str);

string word;

while (ss >> word) {

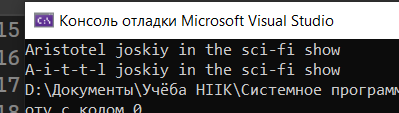
crossOutLetters(word);

cout << word << " ";

}

return 0;

}



12. Составить программы для перевода арабских чисел в римские и для обратной операции. Например, 255 = CCLV = сто + сто + пятьдесят + пять Замечание. Подобными алгоритмами перевода чисел из одной системы в другую мы пользуемся по нескольку раз на дню, когда ведем денежные расчеты. Сумма денег – это арабское число, которому соответствует определенный набор банкнот и монет (аналоги римских цифр).

#include <iostream>

#include <string>

#include <map>

using namespace std;

string intToRoman(int num) {

map<int, string> values = {

{1000, "M"}, {900, "CM"}, {500, "D"}, {400, "CD"},

{100, "C"}, {90, "XC"}, {50, "L"}, {40, "XL"},

{10, "X"}, {9, "IX"}, {5, "V"}, {4, "IV"}, {1, "I"}

};

string result;

for (auto i = values.rbegin(); i != values.rend(); i++) {

while (num >= i->first) {

result += i->second;

num -= i->first;

}

}

return result;

}

int main() {

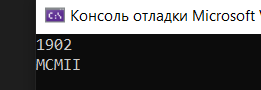
int num;

cin >> num;

cout << intToRoman(num) << endl;

return 0;

}



Подсчитать, сколько букв надо исправить в слове Х, чтобы получилось слово Y (Х,Y – слова одинаковой длины).

#include <iostream>

#include <string>

int main() {

std::string X, Y;

std::cin >> X >> Y;

int count = 0;

for (int i = 0; i < X.length(); i++) {

if (X[i] != Y[i]) {

count++;

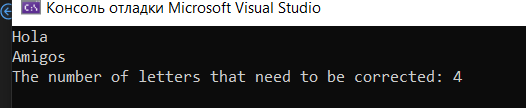
}

}

std::cout << "The number of letters that need to be corrected: " << count << std::endl;

return 0;

}



Составить программу для подсчета числа одинаковых букв в словах X и Y равной длины, стоящих на одних и тех же местах.

#include <iostream>

#include <string>

int main() {

std::string X, Y;

std::cin >> X >> Y;

int count = 0;

for (int i = 0; i < X.length(); i++) {

if (X[i] == Y[i]) {

count++;

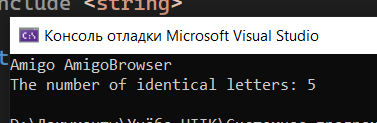
}

}

std::cout << "The number of identical letters: " << count << std::endl;

return 0;

}



15. Задано определенное количество конкретных сочетаний букв (например, УЩ, ЮЩ и др.). Определить, сколько таких групп символов содержится в тексте, вводимом с клавиатуры.

#include <iostream>

#include <string>

#include <vector>

int main() {

std::vector<std::string> combinations = { "USCH", "YUSCH", "..." };

std::string text;

std::cin >> text;

int count = 0;

for (const auto& combination : combinations) {

size\_t pos = text.find(combination);

while (pos != std::string::npos) {

count++;

pos = text.find(combination, pos + 1);

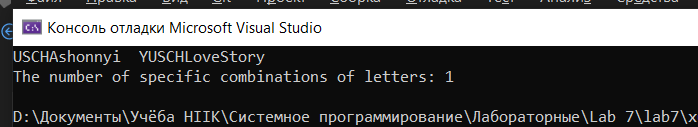
}

}

std::cout << "The number of specific combinations of letters: " << count << std::endl;

return 0;

}



С клавиатуры вводится текст. Подсчитать и вывести на печать количество слов текста, начинающихся с гласной.

#include <iostream>

#include <string>

int main() {

std::string text;

std::cin >> text;

int count = 0;

for (size\_t i = 0; i < text.length(); i++) {

if (text[i] == ' ') {

continue;

}

if (text[i] == 'a' || text[i] == 'e' || text[i] == 'i' || text[i] == 'o' || text[i] == 'u' || text[i] == 'A' || text[i] == 'E' || text[i] == 'I' || text[i] == 'O' || text[i] == 'U') {

count++;

}

while (i < text.length() && text[i] != ' ') {

i++;

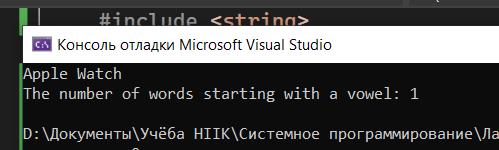
}

}

std::cout << "The number of words starting with a vowel: " << count << std::endl;

return 0;

}



Для заданного текста определить длину содержащейся в нем максимальной серии символов, отличных от латинских букв.

#include <iostream>

#include <string>

using namespace std;

int main() {

string text;

getline(cin, text);

int maxLength = 0;

int currentLength = 0;

for (int i = 0; i < text.length(); i++) {

char c = text[i];

if (!isalpha(c)) {

currentLength++;

if (currentLength > maxLength) {

maxLength = currentLength;

}

}

else {

currentLength = 0;

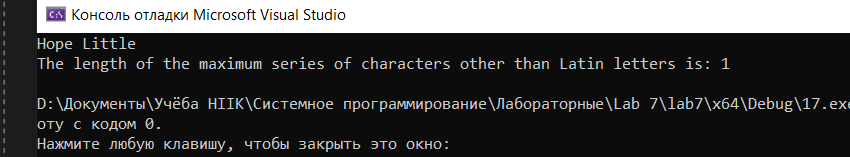
}

}

cout << "The length of the maximum series of characters other than Latin letters is: " << maxLength << endl;

return 0;

}



Записать программу, выясняющую, можно ли из букв слова X составить слово Y.

#include <iostream>

#include <string>

#include <algorithm>

using namespace std;

bool canForm(string x, string y) {

sort(x.begin(), x.end());

sort(y.begin(), y.end());

return x == y;

}

int main() {

string x, y;

cin >> x >> y;

if (canForm(x, y)) {

cout << "Yes, the letters of word X can form word Y." << endl;

}

else {

cout << "No, the letters of word X cannot form word Y." << endl;

}

return 0;

}

