**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по практической работе №4**

**по дисциплине «Программирование»**

Тема: Текстовые строки

|  |  |  |
| --- | --- | --- |
| Студент гр. 2372 |  | Соколовский В.Д. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2022

**Цель работы.**

Изучение текстовых строк как массивов символов и их обработке, а также структуры «строка» из области имен std. Изучение потоков, используемых для работы с файлами и непосредственно чтения текстовой строки из файла. Написание программы, способной отредактировать входной текст, исключив из него наиболее очевидные ошибки. Программа так же должна содержать паттерн поиска подстроки Кнута-Морриса-Пратта (далее КМП).

**Основные теоретические положения.**

Текстовые строки представляются с помощью одномерных массивов символов. В изучаемом языке C++ текстовая строка представляет собой набор символов, заканчивающийся нулевым символом ‘\n’. Поэтому для текстовой строки необходимо выделить память под N+1 символов, где N – количество символов в строке.

Чтение текстовых строк из потока ввода осуществляется до первого пробельного символа. Таким образом, чтобы прочесть всю введённую строку полностью, следует воспользоваться функцией получения строки из потока ввода.

Основными задачами при обработке текстовых строк являются следующие: определение фактической длины текста, копирование текста из одной строки в другую, объединение строк, сравнение (больше, меньше, равно) строк и т.п.

Для удобной работы с строками существует высокоуровневый класс string в пространстве имен std. Этот класс предоставляет готовый набор методов для работы со строками, что значительно упрощает разработку и уменьшает объем исходного кода, делая его более читабельным.

**Постановка задачи.**

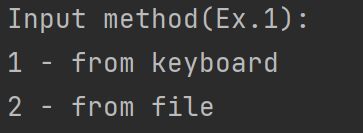
Разработать алгоритм и написать программу, которая:

1. Считывает строку с файла или с клавиатуры. Способ ввода выбирается пользователем.
2. Редактирует входной текст
   1. Удаляет лишние пробелы
   2. Удаляет лишние знаки препинания
   3. Исправляет регистр букв
3. Выводит на экран все слова, содержащие
   1. Только буквы
   2. Только цифры
   3. И буквы, и цифры
4. Выводит на экран ту же строку, заменив во всех словах цифры на буквы латинского алфавита, номера которых соответствуют заменяемой цифре (0 – a, 1 – b и т.д.)
5. Осуществляет поиск подстроки, введённой с клавиатуры.
   1. Производит поиск с помощью алгоритма линейного поиска
   2. Производит поиск с помощью алгоритма КМП

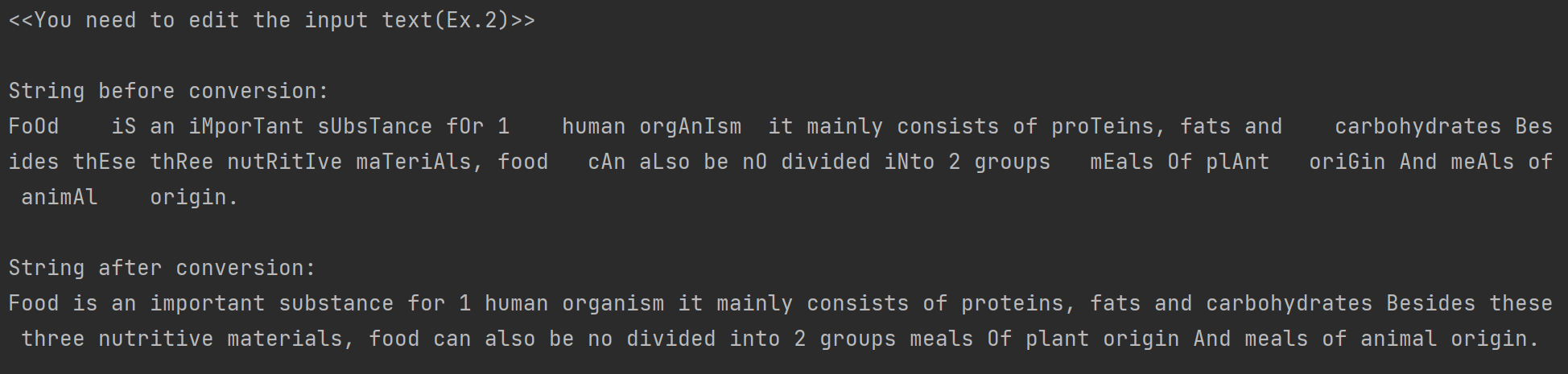
**Выполнение работы.**

Код программы представлен в приложении А.

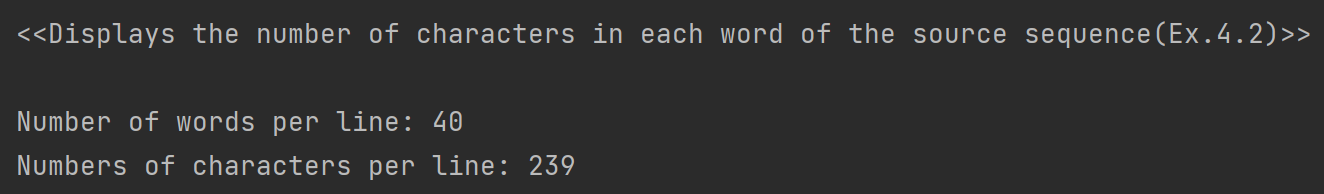
1. При запуске программы пользователю предлагается ввести строку. Если пользователь оставит ввод пустым, то строка будет прочитана из файла.



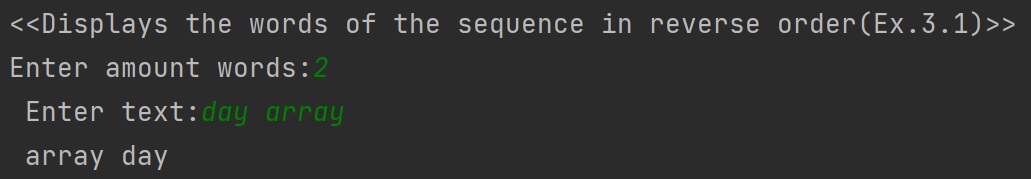
1. Следующий шаг зависит от введенной команды, если пользователь ввёл:



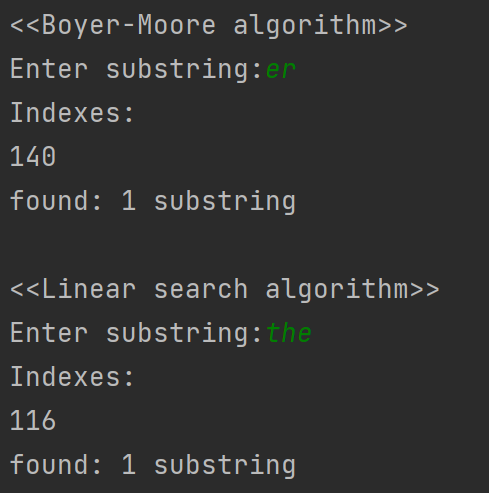
1. Выводится на экран количество символов в каждом слове исходной последовательности.



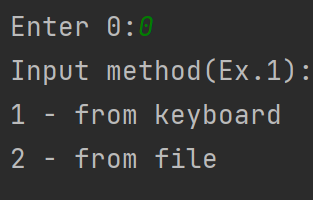
1. **Выводится на экран слова последовательности в обратном порядке.**

****

1. **Выполняется поиск всех вхождений подстроки линейным алгоритмом и алгоритмом Бойера-Мура**

****

1. **Повторить программу: ввести 0**

****

**Выводы.**

В ходе работы были изучены текстовые строки как массивы, класс string и методы обработки строк. Изучен паттерн поиска подстроки в строке Кнута-Морриса-Пратта.

Приложение А

рабочий код

#include <iostream>  
#include <string>  
#include <limits>  
#include <fstream> /\*provides functionality for reading data from a file and for writing to a file  
 (предоставляет функционал для считывания данных из файла и для записи в файл)\*/  
  
using namespace std;  
  
  
  
string text(){  
 cout << "\nEnter number:";  
 int n;  
 cin >> n;  
 cin.ignore((numeric\_limits<streamsize>::max)(), '\n'); /\* removes newline from cin  
 (убирает новую строку из cin)\*/  
  
 string s;  
  
  
 if(n == 1){  
 getline(cin, s);  
 }  
 else if(n == 2){  
 ifstream File;  
 File.open("C:\\Users\\User\\CLionProjects\\practic4\\gaaga.txt");  
 if (File.is\_open()) {  
 while (!File.eof()) { /\* eof - returns true if the file is empty, and returns false if the file is not empty  
 (выдает истину, если файл пуст, а если файл не пуст, то выдает ложь)\*/  
 string s1;  
 getline(File, s1);  
 s = s + s1;  
 }  
 }  
 }  
 else{  
 cout << "Error";  
 }  
 return s;  
}  
  
  
  
void newFile(string s){  
 fstream File("C:\\Users\\User\\CLionProjects\\practic4\\gaaga.txt", ios::out | ios::trunc);  
 if(File.is\_open()){  
 File << s;  
 }  
 else{  
 cout << "file could not be opened";  
 }  
}  
  
  
  
string charact(string s){ /\* counted together with spaces and dots  
 (считается вместе с пробелами и точками)\*/  
  
 int i = 0;  
 if(s[i] == ' '){  
 while(s[i] != '\0') { /\* "\0" - means end of line  
 (означает конец строки)\*/  
 i++;  
 }  
 }  
 int word = 0;  
 int count = 0; // подсчет  
 while(s[i] != '\0'){  
 if(s[i] != ' ' && word == 0){  
 word = 1;  
 count++;  
 }  
 else if(s[i] == ' '){  
 word = 0;  
 }  
 i++;  
 }  
 cout << "\n<<Displays the number of characters in each word of the source sequence(Ex.4.2)>>\n";  
 cout << "\nNumber of words per line: " << count;  
 cout << endl;  
  
 while(s[i] != '\0'){  
 i++;  
 }  
 cout <<"Numbers of characters per line: " << i;  
}  
  
  
  
string convertion(string s) {  
 string str = "";  
 cout << "String before conversion: " << "\n" << s << "\n";  
  
 bool a = true;  
 for(int i = 0; i < s.length(); i++){ // s.length() - длина строки  
 if(s[i] == ' '){  
 if(a){  
 continue;  
 }  
 else{  
 a = true;  
 str = str + s[i];  
 }  
 }  
 else if(s[i] >= 'a' && s[i] <= 'z' || s[i] >= 'A' && s[i] <= 'Z' || s[i] >= '0' && s[i] <= '9'){  
 if(s[i] >= 'A' && s[i] <= 'Z' && !a){  
 str += s[i] + 32;  
 }  
 else{  
 str = str + s[i];  
 }  
 a = false;  
 }  
 else{  
 a = false;  
 if(s[i] == '.' && s[i + 1] == '.' && s[i + 2] == '.'){  
 i = i + 2;  
 str = str + "...";  
 continue;  
 }  
 else if(s[i] == '.'){  
 str = str + '.';  
 break;  
 }  
 if(s[i] != s[i - 1]){  
 str = str + s[i];  
 }  
 }  
 }  
 cout << "\n" << "String after conversion: " << "\n" << str;  
 return str;  
}  
  
  
  
string amount(){  
  
 int words;  
 cout << "Enter amount words: ";  
 cin >> words;  
  
 string \*Text = new string[words];  
  
 cout << "Enter text: ";  
 for(int i = 0; i < words; i++){  
 cin >> Text[i];  
 }  
  
 for(int i = words - 1; i >= 0; i--){  
 cout << Text[i] << " ";  
 }  
}  
  
  
  
void linearS(string s){  
 string sub;  
 int c = 0;  
  
 cout << "Enter substring:";  
 getline(cin, sub);  
  
 cout << "Indexes:\n";  
  
 string s1 = "";  
 for(int i = 0; i <= s.length() - sub.length(); i++){  
 s1 = "";  
 for(int j = i; j < sub.length() + i; j++){  
 if(s[j] == sub[j - i]){  
 s1 = s1 + s[j];  
 }  
 else{  
 continue;  
 }  
 }  
 if (sub == s1) {  
 cout << i + 1 << " ";  
 c = c + 1;  
 }  
 }  
 cout << "\nfound: " << c << " substring\n";  
}  
  
  
  
int max(int a, int b){  
 if(a > b){  
 return a;  
 }  
 else{  
 return b;  
 }  
}  
  
  
  
void charH(string s, int size, int chard[256]){  
  
 for(int i = 0; i < 256; i++){  
 chard[i] = -1;  
 }  
 for(int i = 0; i < size; i++){  
 chard[(int)s[i]] = i;  
 }  
}  
  
  
  
void boyer(string s){  
 string sub;  
 cout << "Enter substring:";  
 getline(cin, sub);  
 cout << "Indexes:\n";  
 int g = sub.size();  
 int k = s.size();  
 int c = 0;  
 int chard[256]; // max line  
  
 charH(sub, g, chard);  
  
 int s1 = 0;  
 while(s1 <= (k - g)){  
 int j = g - 1;  
  
 while(j >= 0 && sub[j] == s[s1 + j]){  
 j--;  
 }  
 if(j < 0){  
 cout << s1 << " ";  
 c = c + 1;  
 if (s1 + g < k){  
 s1 += g - chard[s[s1 + g]];  
 }  
 else{  
 s1 = s1 + 1;  
 }  
 }  
 else{  
 s1 += max(1, j - chard[s[s1 + j]]);  
 }  
 }  
 cout << "\nfound: " << c << " substring\t";  
}  
  
  
  
int main() {  
 int n = 0;  
 string s;  
 string sub;  
 while(!n){  
 cout << "Input method(Ex.1):\n";  
 cout << "1 - from keyboard\n";  
 cout << "2 - from file\n";  
 s = text();  
 cout << "\n";  
  
 cout << "\n<<You need to edit the input text(Ex.2)>>\n\n";  
 s = convertion(s);  
 cout << "\n";  
  
 cout << "\n";  
 charact(s);  
 cout << "\n";  
  
 cout << "\n<<Boyer-Moore algorithm>>\n";  
 boyer(s);  
 cout << "\n";  
  
 cout << "\n<<Linear search algorithm>>\n";  
 linearS(s);  
 cout << "\n";  
  
 cout << "\n<<Displays the words of the sequence in reverse order(Ex.3.1)>>\n";  
 amount();  
 cout << "\n";  
  
 cout << "\n";  
 cout << "Enter 0:";  
 cin >> n;  
 }  
 newFile(s);  
 return 0;  
}