**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: Строки.

|  |  |  |
| --- | --- | --- |
| Студент гр. 2372 |  | Братко В.В. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2022

**Цель работы.**

Изучение строк, работа со строками.

**Основные теоретические положения.**

Класс string предназначен для работы со строками типа char, которые представляют собой строчку с завершающим нулем (символ ‘\0’). Класс string был введен как альтернативный вариант для работы со строками типа char.

Чтобы использовать возможности класса string, нужно подключить библиотеку <string> и пространство имен std. Объявление же переменной типа string осуществляется схоже с обычной переменной:

string S1; // Переменная с именем s1 типа string

string S2 = “Пример”; // объявление с инициализацией

Создание нового типа string было обусловлено недостатками работы с строками символов, который показывал тип char. В сравнении с этим типом string имеет ряд основных преимуществ:

·        возможность использования для обработки строк стандартные операторы С++(=,+,<,==,>,+=,!=,<=,>=,[])(=,+,<,==,>,+=,!=,<=,>=,[]). Использование типа char приводило требовало написание чрезмерного программного кода;

·        обеспечение лучшей надежности программного кода;

·        обеспечение строки, как самостоятельного типа данных.

/\* string.cpp: Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы \*/

#include <iostream>

#include <string>

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian");

string A = "Пример"; // Объявляем и инициализируем строку А

string B; // Объявляем строку В

string C = "текста"; // Объявляем и инициализируем строку С

B = A; // Копируем текст из строки А в строку В

C = B + " " + C;

if(C == "Пример текста") // Проверяем корректно ли произошло присоединение строк

cout << C;

return 0;

}

Результат выполнения программы – вывод в консоль сообщения:

Пример текста

С string можно использовать оператор индексации и получать значения символа. Принцип действия такой же, как и у типа char:

string s1 = “hello!”;

cout << s1[0]; // Будет выведен символ ‘h’

Класс string обладает широким функционалом:

·        функция compare() сравнивает одну часть строки с другой;

·        функция length() определяет длину строки;

·        функции find() и rfind() служат для поиска подстроки в строке (отличаются функции лишь направлением поиска);

·        функция erase() служит для удаления символов;

·        функция replace() выполняет замену символов;

·        функция insert() необходима, чтобы вставить одну строку в заданную позицию другой строки;

Но весь функционал string накладывает и свой негативный отпечаток. Основным недостатком string в сравнении с типом char является замедленная скорость обработки данных.

**Постановка задачи.**

1. С клавиатуры или с файла (\*) (пользователь сам может выбрать способ ввода) вводится последовательность, содержащая от 1 до 50 слов, в каждом из которых от 1 до 10 строчных латинских букв и цифр. Между соседними словами произвольное количество пробелов. За последним символом стоит точка.
2. Необходимо отредактировать входной текст:

·        удалить лишние пробелы;

·        удалить лишние знаки препинания (под «лишними» подразумевается несколько подряд идущих знаков (обратите внимание, что «…» - корректное использование знака) в тексте);

·        исправить регистр букв, если это требуется (пример некорректного использования регистра букв: пРиМЕр);

1. Вывести на экран только те слова последовательности, в которых первая буква слова встречается в этом слове еще раз.
2. Вывести на экран ту же последовательность, переместив все цифры, содержащиеся в словах, в конец соответствующих слов.
3. Необходимо найти все подстроки, которую введёт пользователь в имеющейся строке. Реализуйте два алгоритма: первый алгоритма – Линейный поиск, а второй алгоритм согласно вашему номеру в списке. Четные номера должны реализовать алгоритм КМП, а нечетные – Бойера-Мура. (\*)

**Выполнение работы.**

|  |  |
| --- | --- |
| Ввод пользователем и обработка данных | Работа алгоритма и вывод на экран |
| 1.Создание строки | |
|  |  |

Продолжение Таблицы

|  |  |
| --- | --- |
| 2. Редактирование строки | |
|  |  |

|  |  |
| --- | --- |
| 3 Вывести на экран только те слова последовательности, в которых первая буква слова встречается в этом слове еще раз. | |
|  |  |

|  |  |
| --- | --- |
| 4 Поиск подстроки | |
|  |  |

|  |  |
| --- | --- |
| Цикличность программы | |
| Eсли пользователь желает, то он может запустить выполнение программы заново, нажав цифру 1 |  |

Продолжение Таблицы

|  |  |
| --- | --- |
| Проверка на ошибки | |
| Eсли пользователь вводит неправильный тип данных, то программа на дает ему это сделать |  |

|  |  |
| --- | --- |
| Полный вывод программы | |
| Здесь представлен весь вывод консоли. | Вывод в консоли: |

Продолжение Таблицы

**Полный код программы:**

#include <iostream>  
#include <fstream>  
#include <chrono>  
  
using namespace std;  
  
int checkInput(){  
 int input;  
 try {  
 cin >> input;  
 if (cin.fail()) {  
 throw 1;  
 }  
 } catch (int exeption) {  
 cout << "ERROR!!!";  
 exit(0);  
 }  
 cin.sync();  
 return input;  
}  
//1.С клавиатуры или с файла (\*) (пользователь сам может выбрать способ ввода) вводится последовательность,  
// содержащая от 1 до 50 слов, в каждом из которых от 1 до 10 строчных латинских букв и цифр.  
// Между соседними словами произвольное количество пробелов. За последним символом стоит точка.  
string function1(){  
 incorrect:  
 cout << "Which type of input do you prefer?(1 - file; 2 - keyboard):";  
 int input = checkInput();  
 string str = "";  
 if (input == 1){  
 ifstream File;  
 File.open("C:\\FirstCursProgramm\\C++\\stepik\_lab\_4\\lab\_4.txt");  
 if(File.is\_open()){  
 while(!File.eof()){  
 string str1;  
 getline(File, str1);  
 str += str1;  
 }  
 } else{  
 cout << "file could not be opened";  
 return "";  
 }  
 File.close();  
 } else if (input == 2){  
 getline(cin, str);  
 }else{  
 goto incorrect;  
 }  
 return str;  
}  
//2. Необходимо отредактировать входной текст:  
//·удалить лишние пробелы;  
//·удалить лишние знаки препинания (под «лишними» подразумевается несколько подряд идущих знаков;  
//·исправить регистр букв, если это требуется (пример некорректного использования регистра букв: пРиМЕр);  
string function2(string str){  
 string strRes = "";  
 cout << "String before editing:" << "\n" << str << "\n";  
 bool flag = true;  
 for (int i = 0; i < str.length(); i++){  
 if (str[i] == ' '){  
 if (flag){  
 continue;  
 }else {  
 flag = true;  
 strRes+=str[i];  
 }  
 }else if(str[i]>='a' && str[i]<='z' || str[i]>='A' && str[i]<='Z' || str[i]>='0' && str[i]<='9'){  
 if (str[i]>='A' && str[i]<='Z' && !flag){  
 strRes+=str[i]+32;  
 }else{  
 strRes+=str[i];  
 }  
 flag = false;  
 }else{  
 flag = false;  
 if(str[i] == '.' && str[i+1] == '.' && str[i+2] == '.'){  
 i+=2;  
 strRes += "...";  
 continue;  
 }else if(str[i] == '.') {  
 strRes += '.';  
 break;  
 }  
 if(str[i] != str[i-1]){  
 strRes += str[i];  
 }  
 }  
 }  
 cout << "String after editing:" << "\n" << strRes;  
 return strRes;  
}  
  
//3.5. Вывести на экран только те слова последовательности, в которых первая буква слова встречается в этом слове еще раз.  
void function3(string str){  
 cout << "Word(s) in which the first letter consist in word:";  
 bool flagSpace = false, flag = false;  
 char first;  
 string str1 = "";  
 for (int i = 0; i < str.length(); i++){  
 if (str[i] == ' '){  
 if (flag){  
 cout << str1 << " ";  
 }  
 flagSpace = true;  
 flag = false;  
 str1 = "";  
 }else{  
 if (flagSpace){  
 flagSpace = false;  
 str1 += str[i];  
 first = str[i];  
 continue;  
 }  
 if (first == str[i]){  
 flag = true;  
 }  
 str1 += str[i];  
 }  
 }  
}  
  
//4.5. Вывести на экран ту же последовательность, переместив все цифры, содержащиеся в словах, в конец соответствующих слов.  
void function4(string str){  
 cout << "String after exercise 4.5: " << "\n";  
 string str1 = "";  
 bool flag = true;  
 for (int i = 0; i < str.length(); i++){  
 if (str[i]>='0' && str[i]<='9'){  
 str1 += str[i];  
 str.erase(i,1);  
 flag = true;  
 if (str[i]>='0' && str[i]<='9'){  
 i--;  
 }  
 }else if ((str[i] == ' ' || !(str[i]>='a' && str[i]<='z' || str[i]>='A' && str[i]<='Z')) && flag){  
 str.insert(i, str1);  
 i += str1.length() - 1;  
 str1 = "";  
 flag = false;  
 }  
 }  
 cout << str;  
}  
  
//5. Необходимо найти все подстроки, которую введёт пользователь в имеющейся строке. Реализуйте два алгоритма:  
// первый алгоритма – Линейный поиск, а второй – Бойера-Мура.  
void function5(string str){  
 string subString = "", strRes = "not exist";  
 int count = 0;  
 cout << "Input the substring you want to find in the string:";  
 getline(cin, subString);  
 cout << "Linear search:" << "\n";  
 string str1 = "";  
 auto start = std::chrono::duration\_cast<std::chrono::nanoseconds>(std::chrono::system\_clock::now().time\_since\_epoch()).count();  
 for(int i = 0; i <= str.length() - subString.length(); i++){  
 str1 = "";  
 for(int j = i; j < subString.length() + i; j++){  
 if (str[j] == subString[j-i]){  
 str1 += str[j];  
 } else {  
 continue;  
 }  
 }  
 if (subString == str1){  
 strRes = "exist";  
 count += 1;  
 }  
 }  
 auto end = std::chrono::duration\_cast<std::chrono::nanoseconds>(std::chrono::system\_clock::now().time\_since\_epoch()).count();  
 cout << "SubString - " << strRes << "; There is(are) " << count << " substring(s);" << "Spent time: " << end - start << "ns\n";  
 cout << "Boyer moore algorithm:" << "\n";  
 strRes = "not exist"; count = 0;  
 int arr[subString.length()];  
 start = std::chrono::duration\_cast<std::chrono::nanoseconds>(std::chrono::system\_clock::now().time\_since\_epoch()).count();  
 for(int i = 0; i < subString.length(); i++){  
 if(i != subString.length() - 1){  
 arr[i] = subString.length() - 1 - i;  
 }else{  
 bool flag = false;  
 for (int j = i - 1; j >= 0; j--){  
 if (subString[j] == subString[i]){  
 flag = true;  
 arr[i] = arr[j];  
 break;  
 }  
 }  
 if(!flag){  
 arr[i] = subString.length();  
 }  
 }  
 }  
 for(int i = subString.length() - 1; i < str.length();){  
 string str1 = "";  
 bool flag = true;  
 if(str[i] == subString[subString.length() - 1]){  
 for(int j = i; j > int(i - subString.length()); j--){  
 if(str[j] == subString[subString.length() - 1 - i + j]){  
 str1 = str[j] + str1;  
 }else{  
 flag = false;  
 i += subString.find(str[j]) != std::string::npos ? abs(arr[subString.find\_last\_of(str[j])] - int(i - j)) : subString.length() - int(i - j);  
 break;  
 }  
 }  
 if(flag){  
 strRes = "exist";  
 count += 1;  
 i += arr[subString.length() - 1];  
 }  
  
 }else{  
 i += subString.find(str[i]) != std::string::npos ? arr[subString.find\_last\_of(str[i])] : subString.length();  
 }  
 }  
 end = std::chrono::duration\_cast<std::chrono::nanoseconds>(std::chrono::system\_clock::now().time\_since\_epoch()).count();  
 cout << "SubString - " << strRes << "; There is(are) " << count << " substring(s);" << "Spent time: " << end - start << "ns\n";  
}  
  
void writeFile(string str){  
 fstream File ( "C:\\FirstCursProgramm\\C++\\stepik\_lab\_4\\lab\_4\_res.txt", ios :: out | ios :: trunc );  
 if(File.is\_open()){  
 File << str;  
 }else{  
 cout << "file could not be opened";  
 }  
}  
  
int main() {  
 string str;  
 int input = 0;  
 while(!input){  
 //1. С клавиатуры или с файла (\*) (пользователь сам может выбрать способ ввода) вводится последовательность,  
 // содержащая от 1 до 50 слов, в каждом из которых от 1 до 10 строчных латинских букв и цифр.  
 // Между соседними словами произвольное количество пробелов. За последним символом стоит точка.  
 str = function1();  
 //2. Необходимо отредактировать входной текст:  
 //·удалить лишние пробелы;  
 //·удалить лишние знаки препинания (под «лишними» подразумевается несколько подряд идущих знаков;  
 //·исправить регистр букв, если это требуется (пример некорректного использования регистра букв: пРиМЕр);  
 str = function2(str);  
 cout << "\n";  
 //5. Необходимо найти все подстроки, которую введёт пользователь в имеющейся строке. Реализуйте два алгоритма:  
 // первый алгоритма – Линейный поиск, а второй – Бойера-Мура.  
 function5(str);  
 cout << "\n";  
 //3.5. Вывести на экран только те слова последовательности, в которых первая буква слова встречается в этом слове еще раз.  
 function3(str);  
 cout << "\n";  
 //4.5. Вывести на экран ту же последовательность, переместив все цифры, содержащиеся в словах, в конец соответствующих слов.  
 function4(str);  
 // Записать полученную строку в файл  
 cout << "\n";  
 cout << "If you want to run the program again press 0:";  
 input = checkInput();  
 }  
 writeFile(str);  
 return 0;  
}

**Вывод:**

В процессе написания программы для практической работы было изучено:

1. Что такое класс String
2. Как работает со строками
3. Зачем нужны строки