|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт Информационных технологий | |
|  | |
| Кафедра Математического обеспечения и стандартизации информационных технологий | |
|  | |
|  | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 3** | |
| **по дисциплине** | |
| **«**Структуры и алгоритмы обработки данных**»**  **Тема: «Структура данных - строка»** | |
|  | |
| Выполнил студент группы ИКБО-08-22 | Сенькевич Г.Д. |
| Принял преподаватель | Филатов А.С. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись руководителя)* |

Москва 2023

# **Цели работы**

* получение навыков в разработке алгоритмов обработки текста – извлечение отдельных элементов.
* получение навыков использования средств языка Си и С++ для реализации алгоритмов обработки текстовых данных.

# **Постановка задачи**

1. Разработать программу согласно задаче варианта, используя для представления обрабатываемого в программе текста нуль терминальную строку и средства языка С для выполнения операций над этой строкой.
2. Разработать программу согласно задаче варианта, используя для представления обрабатываемого в программе текста строку string стандартной библиотеки шаблонов и возможности класса для выполнения действий со строкой.
3. Составить отчет, отобразив в нем описание выполнения всех этапов разработки, тестирования и код всей программы со скриншотами результатов тестирования.

Вариант №23. Условие задания:

|  |  |
| --- | --- |
| Упражнение 1 | Дан текст программы на языке С++. Вывести все ключевые слова, встретившиеся в тексте программы и указать их количество. Использовать словарь ключевых слов языка С++ (некоторый набор). |

# **Решение**

Строки - это последовательности символов, которые могут использоваться для представления текстовых данных в программировании. В языке программирования C строки представлены как массивы символов, заканчивающиеся символом нуля '\0', который указывает конец строки. Это называется нультерминальным символом, и он обеспечивает завершение строки. Строки в C могут быть созданы как литеральные строки, так и как массивы символов. Литеральные строки представляются в двойных кавычках, например: "Hello, World!". Массивы символов также могут быть использованы для создания строк, например: char str[6] = {'H', 'e', 'l', 'l', 'o', '\0'}. Объем памяти для переменной строкового типа равен размеру массива, но количество символов в строке на 1 меньше.

В C++ помимо строк из C есть std::string - std::string представляет собой класс, который позволяет работать со строками. Он является частью стандартной библиотеки C++ и предоставляет множество методов для работы со строками, таких как добавление и удаление символов, поиск подстроки, сравнение строк и т.д. std::string может быть создан пустой, с заданным размером или с начальным значением. Для создания строки со значением можно использовать литералы строк, например: std::string str = "Hello, World!". std::string обеспечивает автоматическую выделение и освобождение памяти. В отличие от массивов символов, которые имеют фиксированный размер, std::string может изменять свой размер динамически в зависимости от потребностей программы.

Для решения первого упражнения были написаны две функции print\_keywards (одна для char[] и одна для std::string), выводящие на экран все ключевые слова языка C++, встретившиеся в строке и подсчитывающие их количество. Функция принимает на вход строку (char[] или std::string) – код, ключевые слова в котором нужно подсчитать и ничего не возвращает.

|  |
| --- |
| // Функция для char[]  void print\_keywords(char\* code)  {  int count = 0;  char current\_word[MAXIMUM\_WORD\_LENGTH] = "";  int last\_symbol\_index = -1;  std::cout << "Список ключевых слов в коде:" << std::endl;  for (int i = 0; code[i] != '\0'; i++)  {  if (code[i] == ' ' || code[i] == '\n' || code[i] == '\t')  {  for (int j = 0; j < NUMBER\_OF\_KEYWORDS; j++)  {  if (strcmp(current\_word, keywords[j]) == 0)  {  std::cout << keywords[j] << std::endl;  count++;  }  }  memset(current\_word, 0, sizeof(current\_word));  last\_symbol\_index = -1;  }  else if (last\_symbol\_index + 1 < MAXIMUM\_WORD\_LENGTH)  {  last\_symbol\_index++;  current\_word[last\_symbol\_index] = code[i];  }  }  std::cout << "Всего ключевых слов: " << count << std::endl;  }  // Функция для std::string  void print\_keywords(std::string code)  {  int count = 0;  std::string current\_word = "";  std::cout << "Список ключевых слов в коде:" << std::endl;  for (int i = 0; code[i] != '\0'; i++)  {  if (code[i] == ' ' || code[i] == '\n' || code[i] == '\t')  {  for (int j = 0; j < NUMBER\_OF\_KEYWORDS; j++)  {  if (current\_word == keywords[j])  {  std::cout << keywords[j] << std::endl;  count++;  }  }  current\_word.clear();  }  else  {  current\_word += code[i];  }  }  std::cout << "Всего ключевых слов: " << count << std::endl;  } |

При запуске программы пользователь видит пользовательское меню, позволяющее выбрать между двумя различными типами строки.

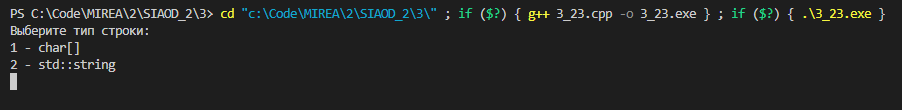


Рисунок 1. Интерфейс программы

# **Тестирование**

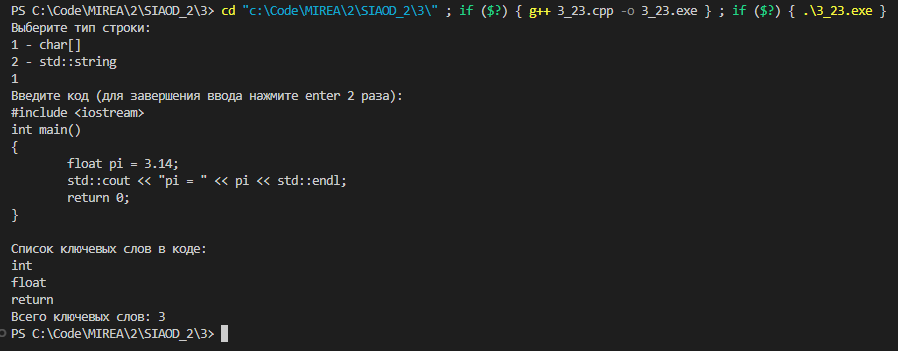
Протестируем выполнение программой первого упражнения с реализацией с помощью char[]. Для этого выберем в меню первый тип строки и введём код простой программы. На рисунке 2 видим, как программа вывела верный ответ.

Рисунок 2. Тестирование программы

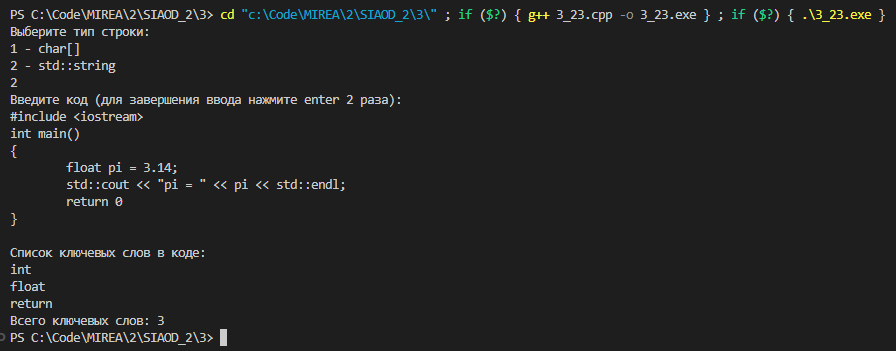
Протестируем выполнение программой первого упражнения с реализацией с помощью std::string. Для этого выберем в меню первый тип строки и введём код простой программы. На рисунке 3 видим, как программа вывела верный ответ.

Рисунок 3. Тестирование программы

Из результатов выполнения программы видно, что программа работает корректно, решая все поставленные задачи.

# **Вывод**

В результате выполнения работы я:

1. Освоил алгоритмы работы со строками на языке программирования C++.

# **Исходный код программы**

|  |
| --- |
| #include <iostream>  #include <cstring>  #include <string>  #define MAXIMUM\_CODE\_LENGTH 1024  #define MAXIMUM\_WORD\_LENGTH 64  #define NUMBER\_OF\_KEYWORDS 97  // https://en.cppreference.com/w/cpp/keyword  const char\* const keywords[NUMBER\_OF\_KEYWORDS] = {  "alignas",  "alignof",  "and",  "and\_eq",  "asm",  "atomic\_cancel",  "atomic\_commit",  "atomic\_noexcept",  "auto",  "bitand",  "bitor",  "bool",  "break",  "case",  "catch",  "char",  "char8\_t",  "char16\_t",  "char32\_t",  "class",  "compl",  "concept",  "const",  "consteval",  "constexpr",  "constinit",  "const\_cast",  "continue",  "co\_await",  "co\_return",  "co\_yield",  "decltype",  "default",  "delete",  "do",  "double",  "dynamic\_cast",  "else",  "enum",  "explicit",  "export",  "extern",  "false",  "float",  "for",  "friend",  "goto",  "if",  "inline",  "int",  "long",  "mutable",  "namespace",  "new",  "noexcept",  "not",  "not\_eq",  "nullptr",  "operator",  "or",  "or\_eq",  "private",  "protected",  "public",  "reflexpr",  "register",  "reinterpret\_cast",  "requires",  "return",  "short",  "signed",  "sizeof",  "static",  "static\_assert",  "static\_cast",  "struct",  "switch",  "synchronized",  "template",  "this",  "thread\_local",  "throw",  "true",  "try",  "typedef",  "typeid",  "typename",  "union",  "unsigned",  "using",  "virtual",  "void",  "volatile",  "wchar\_t",  "while",  "xor",  "xor\_eq"  };  void print\_keywords(char\* code)  {  int count = 0;  char current\_word[MAXIMUM\_WORD\_LENGTH] = "";  int last\_symbol\_index = -1;  std::cout << "Список ключевых слов в коде:" << std::endl;  for (int i = 0; code[i] != '\0'; i++)  {  if (code[i] == ' ' || code[i] == '\n' || code[i] == '\t')  {  for (int j = 0; j < NUMBER\_OF\_KEYWORDS; j++)  {  if (strcmp(current\_word, keywords[j]) == 0)  {  std::cout << keywords[j] << std::endl;  count++;  }  }  memset(current\_word, 0, sizeof(current\_word));  last\_symbol\_index = -1;  }  else if (last\_symbol\_index + 1 < MAXIMUM\_WORD\_LENGTH)  {  last\_symbol\_index++;  current\_word[last\_symbol\_index] = code[i];  }  }  std::cout << "Всего ключевых слов: " << count << std::endl;  }  void print\_keywords(std::string code)  {  int count = 0;  std::string current\_word = "";  std::cout << "Список ключевых слов в коде:" << std::endl;  for (int i = 0; code[i] != '\0'; i++)  {  if (code[i] == ' ' || code[i] == '\n' || code[i] == '\t')  {  for (int j = 0; j < NUMBER\_OF\_KEYWORDS; j++)  {  if (current\_word == keywords[j])  {  std::cout << keywords[j] << std::endl;  count++;  }  }  current\_word.clear();  }  else  {  current\_word += code[i];  }  }  std::cout << "Всего ключевых слов: " << count << std::endl;  }  int main()  {  int mode = 0;  std::cout << "Выберите тип строки:" << std::endl  << "1 - char[]" << std::endl  << "2 - std::string" << std::endl;  std::cin >> mode;  if (mode == 1)  {  std::cout << "Введите код (для завершения ввода нажмите enter 2 раза):"  << std::endl;    char code[MAXIMUM\_CODE\_LENGTH] = "";  int last\_symbol\_index = -1;  char current = ' ';  char prev = ' ';  while (!(current == '\n' && prev == '\n'))  {  prev = current;  current = std::cin.get();  if (last\_symbol\_index + 2 < MAXIMUM\_CODE\_LENGTH)  {  last\_symbol\_index++;  code[last\_symbol\_index] = current;  }  else  {  break;  }  }    print\_keywords(code);  }  else if (mode == 2)  {  std::cout << "Введите код (для завершения ввода нажмите enter 2 раза):"  << std::endl;  std::string code;    char current = ' ';  char prev = ' ';  while (!(current == '\n' && prev == '\n'))  {  prev = current;  current = std::cin.get();  code += current;  }    print\_keywords(code);  }  else  {  std::cout << "Такого варианта нет!" << std::endl;  }    return 0;  } |