

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**"МИРЭА - Российский технологический университет"**

**РТУ МИРЭА**

Институт информационных технологий (ИТ)

Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №4**

**по дисциплине**

**«Структуры и алгоритмы обработки данных»**

Тема. Стандартные типы данных языка программирования С++ для представления текстовых данных

|  |  |  |
| --- | --- | --- |
| Выполнил студент группы ИКБО-04-22 |  | Кликушин В.И. |
| Принял старший преподаватель |  | Скворцова Л.А. |

Москва 2023

ОГЛАВЛЕНИЕ

[1 УСЛОВИЕ ЗАДАЧИ И ЗАДАНИЕ ВАРИАНТА 3](#_Toc131195264)

[2 РАЗРАБОТКА ПРОГРАММЫ 4](#_Toc131195265)

[2.1 ПОСТАНОВКА ЗАДАЧИ 4](#_Toc131195266)

[2.2 МОДЕЛЬ РЕШЕНИЯ 4](#_Toc131195267)

[2.2.1 Тестовый пример, приводящий к решению задачи 4](#_Toc131195268)

[2.2.2 Декомпозиция 4](#_Toc131195269)

[2.2.3 Концепция алгоритма решения задачи 5](#_Toc131195270)

[2.2.4 Набор тестов 6](#_Toc131195271)

[3 РЕАЛИЗАЦИЯ ЗАДАНИЯ 1 7](#_Toc131195272)

[3.1 ОПИСАНИЕ ПРИМЕНЯЕМЫХ ФУНКЦИЙ STRING.H 7](#_Toc131195273)

[3.2 КОД РЕАЛИЗАЦИИ ЗАДАНИЯ 1 7](#_Toc131195274)

[3.3 ВРЕМЯ ВЫПОЛНЕНИЯ ФУНКЦИИ, РЕАЛИЗУЮЩЕЙ ЗАДАЧУ 10](#_Toc131195275)

[3.4 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ 10](#_Toc131195276)

[4 РЕАЛИЗАЦИЯ ЗАДАНИЯ 2 13](#_Toc131195277)

[4.1 ОПИСАНИЕ ПРИМЕНЯЕМЫХ ФУНКЦИЙ STRING 13](#_Toc131195278)

[4.2 КОД РЕАЛИЗАЦИИ ЗАДАНИЯ 2 13](#_Toc131195279)

[4.3 ВРЕМЯ ВЫПОЛНЕНИЯ ФУНКЦИИ, РЕАЛИЗУЮЩЕЙ ЗАДАЧУ 16](#_Toc131195280)

[4.4 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ 16](#_Toc131195281)

[5 ВЫВОДЫ 18](#_Toc131195282)

[6 ИСПОЛЬЗУЕМЫЕ ИСТОЧНИКИ 19](#_Toc131195283)

# 1 УСЛОВИЕ ЗАДАЧИ И ЗАДАНИЕ ВАРИАНТА

Разработать программу, согласно задаче варианта, используя для представления в программе текста нуль терминальную строку (строку, завершающуюся нулем) и средства языка С для выполнения операций над этой строкой.

Дано предложение и управляющий символ, значением которого может быть один из символов Y, N. Вывести этот текст без входящих в него цифр, если значение символа управления = Y, а если этот символ = N, то перенести все цифры текста в конец предложения так, чтобы первая встреченная в исходном тексте цифра была последней цифрой, а последняя первой и был сохранен порядок следования остальных цифр.

# 2 РАЗРАБОТКА ПРОГРАММЫ

## 2.1 ПОСТАНОВКА ЗАДАЧИ

Необходимо получить представления о работе со строками в языках C и C++; ознакомиться с функциями файла <string.h>; изучить функции для работы со строкой, завершающейся нулем; разработать программу, согласно задаче варианта с использованием средств языка C, а затем переписать её, применяя современный инструмент для работы со строками в языке C++ - класс string, для чего рассмотреть функции ввода и вывода текста в строку в этом классе, изучить набор перегруженных операторов и встроенные методы класса.

## 2.2 МОДЕЛЬ РЕШЕНИЯ

## 2.2.1 Тестовый пример, приводящий к решению задачи

Предположим, что имеется непустая строка, содержащая набор различных символов, включая цифры.

str = “v1l2a3d4\*”

Пользователь будет вводить один из возможных символов управления N или Y. Таким образом, необходимо реализовать две функции, которые в зависимости от управляющего символа будут преобразовывать строку.

Функция, отвечающая за преобразование в соответствие с символом Y будет посимвольно перебирать каждый элемент строки, и если он не будет являться цифрой, то в объявленную в функции пустую строку будет записываться этот символ, а в конце программа скопирует содержимое новой строки и «подменит» исходную.

Функция, отвечающая за преобразование в соответствие с символом N будет иметь аналогичный алгоритм, только теперь придется отдельно «запоминать» все цифры, встретившиеся в строке и остальные символы в отдельные переменные. После завершения цикла прохода по исходной строке необходимо поменять местами первый и последний символ в строке с цифрами.

Таким образом, при управляющем символе Y рассматриваемая строка будет преобразована как “vlad\*”, а при символе N: “ vlad\*4231”.

## 2.2.2 Декомпозиция

Для реализации алгоритма задачи не потребовались функции, отвечающие за промежуточные подзадачи, тем не менее, опишем встроенные методы для записи алгоритма на псевдокоде.

1. Склеить две строки – встроенный метод.

char \*strcat(char \*s1, char \*s2);

1. Скопировать строку – встроенный метод

char \*strсpy(char \*s1, char \*s2);

1. Проверить, является ли строка подстрокой другой строки

char \*strchr (const char \*str, int c);

## 2.2.3 Концепция алгоритма решения задачи

Функция для символа управления «Y»:

transform\_Y(str)

str\_Y – пустая строка; numbers – строка из всех цифр

n – длина исходной строки, целый тип

for i ← 0 to n do

if strchr(str[i], numbers) then

strcat(str\_Y, str[i])

Endif

od

strcpy(str, str\_Y)

Функция для символа управления «N»:

transform\_N(str)

str\_N, str\_N\_num – пустые строки; numbers – строка из всех цифр

n – длина исходной строки, целый тип

temp – символ

for i ← 0 to n do

if strchr(str[i], numbers) then

strcat(str\_N, str[i])

else

strcat(str\_N, str[i])

Endif

od

strcpy(str, str\_N)

temp = str\_N\_num[0]

str\_N\_num[0] = str\_N\_num[n-1]

str\_N\_num[n-1] = temp

## 2.2.4 Набор тестов

|  |  |  |
| --- | --- | --- |
| Вывести текст без входящих в него цифр, если значение символа управления = Y, а если этот символ = N, то перенести все цифры текста в конец предложения так, чтобы первая встреченная в исходном тексте цифра была последней цифрой, а последняя первой и был сохранен порядок следования остальных цифр. | | |
| Номер теста | Входные данные | Эталон результата |
| 1 | symbol = N  str = «7» | str = «7» |
| 2 | symbol = N  str = «lLl» | str = «lLl» |
| 3 | symbol = N  str = «914q0werty752» | str = «qwerty2140759» |
| 4 | symbol = Y  str = «123» | str = «» (пустая строка) |
| 5 | symbol = Y  str = «123+-456abc» | str = «+-abc» |
| 6 | symbol = Y  str = «AHAHA» | str = «AHAHA» |

# 3 РЕАЛИЗАЦИЯ ЗАДАНИЯ 1

## 3.1 ОПИСАНИЕ ПРИМЕНЯЕМЫХ ФУНКЦИЙ STRING.H

В ходе реализации программы были использованы следующие встроенные функции библиотеки «string.h»:

1. int strlen(char \*s) - возвращает количество символов в строке s;
2. char \*strcat(char \*s1, char \*s2) - присоединяет s2 к s1, возвращает s1
3. char \*strсpy(char \*s1, char \*s2) - копирует строку s2 в строку s1, включая '\0', возвращает s1
4. char \*strchr (const char \*str, int c) - функция, используемая для нахождения вхождения символа в строку; возвращает порядковый указатель на первое вхождение символа, или NULL, если такового в строке нет

## 3.2 КОД РЕАЛИЗАЦИИ ЗАДАНИЯ 1

#include <iostream>

#include <time.h>

#include <string.h>

#include <stdio.h>

#pragma warning(disable : 4996)

using namespace std;

char\* transform\_N(char \*str)

{

char str\_N\_num[1024]{ '\0' };

char str\_N[1024]{ '\0' };

char numbers[11]{ '0','1','2','3','4','5','6','7','8','9','\0' };

for (int i = 0; i < strlen(str); i++)

{

char temp[2]{ str[i],'\0' };

if (strchr(numbers, str[i]) == NULL)

{

strcat(str\_N, temp);

}

else

{

strcat(str\_N\_num, temp);

}

}

strcpy(str, str\_N);

if (strlen(str\_N\_num) != 0)

{

char s = str\_N\_num[0];

str\_N\_num[0] = str\_N\_num[strlen(str\_N\_num) - 1];

str\_N\_num[strlen(str\_N\_num) - 1] = s;

}

strcat(str, str\_N\_num);

return str;

}

char\* transform\_Y(char \*str)

{

char\* str\_Y = new char [strlen(str)] { '\0' };

char numbers[11]{ '0','1','2','3','4','5','6','7','8','9','\0' };

for (int i = 0; i < strlen(str); i++)

{

char temp[2]{ str[i],'\0' };

if (strchr(numbers, str[i]) == NULL)

{

strcat(str\_Y, temp);

}

}

strcpy(str, str\_Y);

if (str\_Y != nullptr)

{

delete[] str\_Y;

}

return str;

}

int main()

{

system("chcp 1251");

int num;

char str[1024]{'\0'};

char symbol;

while (true)

{

cout << "-----------------------------------------------------------------------------------------------------------" << endl;

cout << " Меню\n";

cout << " 1. Ввести строку\n";

cout << " 2. Вывести строку\n";

cout << " 3. Преобразовать строку в соответсвии с управляющим символом\n";

cout << " 4. Завершить работу\n";

cin >> num;

switch (num)

{

case 1:

cout << " Введите строку\n";

cin.ignore();

gets\_s(str);

break;

case 2:

if (strlen(str)!=0)

{

cout << str << endl;

}

else

{

cout << " Пустая строка\n";

}

break;

case 3:

if (strlen(str) != 0)

{

cout << " Введите один из возможных символов управления - N (перенести все цифры текста в конец предложения так, чтобы первая встреченная в исходном тексте цифра была последней цифрой, а последняя первой и был сохранен порядок следования остальных цифр) или Y (вывод текста без входящих в него цифр)\n";

cin >> symbol;

if (symbol == 'N')

{

transform\_N(str);

cout << str << endl;

}

else if (symbol == 'Y')

{

transform\_Y(str);

strlen(str) == 0 ? cout << " В результате преобразования получилась пустая строка\n" : cout << str << endl;

}

else

{

cout << " Нет операции для введённого символа управления\n";

}

}

else

{

cout << " Пустая строка\n";

}

break;

case 4:

return 0;

default:

cout << " Нет такого пункта\n";

}

}

return 0;

}

## 3.3 ВРЕМЯ ВЫПОЛНЕНИЯ ФУНКЦИИ, РЕАЛИЗУЮЩЕЙ ЗАДАЧУ

С помощью функций библиотеки chrono была оценена практическая временная сложность работы алгоритмов функций transform\_N и transform\_Y. В качестве входных данных была сгенерирована произвольная строка из букв и цифр длиною 10000 символов.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1 – Оценка практической временной сложности transform\_N для строки, представленной как массив символов

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2 – Результат оценки времени работы transform\_N для строки, представленной как массив символов

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3 - Оценка практической временной сложности transform\_Y для строки, представленной как массив символов

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 4 - Результат оценки времени работы transform\_Y для строки, представленной как массив символов

## 3.4 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 5 – Результат теста 1

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 6 – Результат теста 2

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 7 – Результат теста 3

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 8 – Результат теста 4

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 9 – Результат теста 5

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 10 – Результат теста 6

## 4 РЕАЛИЗАЦИЯ ЗАДАНИЯ 2

## 4.1 ОПИСАНИЕ ПРИМЕНЯЕМЫХ ФУНКЦИЙ STRING

В ходе реализации программы были использованы следующие встроенные функции библиотеки «string»:

1. str\_length() - возвращает количество символов в строке str;
2. str.find() - ищет в данной строке str первое вхождение другой строки. Возвращается номер первого символа, начиная с которого далее идет подстрока, равная строке s. Если эта строка не найдена, то возвращается константа string::npos (которая равна -1, но при этом является беззнаковой, то есть на самом деле является большим беззнаковым положительным числом). Если задано значение pos, то поиск начинается с позиции pos, то есть возращаемое значение будет не меньше, чем pos. Если значение pos не указано, то считается, что оно равно 0 - поиск осуществляется с начала строки.
3. getline() - читает символы из входного потока и помещает их в строку.
4. Перегруженный оператор сложения – конкатенация (сложение) строк.

## 4.2 КОД РЕАЛИЗАЦИИ ЗАДАНИЯ 2

#include <iostream>

#include <string>

using namespace std;

string transform\_N(string &str)

{

string str\_N, str\_N\_num;

string numbers = "0123456789";

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

{

if (numbers.find(str[i]) == string::npos)

{

str\_N += str[i];

}

else

{

str\_N\_num += str[i];

}

}

if (str\_N\_num.length() != 0)

{

char temp = str\_N\_num[0];

str\_N\_num[0] = str\_N\_num[str\_N\_num.length() - 1];

str\_N\_num[str\_N\_num.length() - 1] = temp;

}

return str = str\_N + str\_N\_num;

}

string transform\_Y(string &str)

{

string str\_Y;

string numbers = "0123456789";

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

{

if (numbers.find(str[i]) == string::npos)

{

str\_Y += str[i];

}

}

return str = str\_Y;

}

int main()

{

system("chcp 1251");

int num;

string str;

char symbol;

while (true)

{

cout << "-----------------------------------------------------------------------------------------------------------" << endl;

cout << " Меню\n";

cout << " 1. Ввести строку\n";

cout << " 2. Вывести строку\n";

cout << " 3. Преобразовать строку в соответсвии с управляющим символом\n";

cout << " 4. Завершить работу\n";

cin >> num;

switch (num)

{

case 1:

cout << " Введите строку\n";

cin.ignore();

getline(cin, str);

break;

case 2:

if (str.length() != 0)

{

cout << str << endl;

}

else

{

cout << " Пустая строка\n";

}

break;

case 3:

if (str.length() != 0)

{

cout << " Введите один из возможных символов управления - N (перенести все цифры текста в конец предложения так, чтобы первая встреченная в исходном тексте цифра была последней цифрой, а последняя первой и был сохранен порядок следования остальных цифр) или Y (вывод текста без входящих в него цифр)\n";

cin >> symbol;

if (symbol == 'N')

{

transform\_N(str);

cout << str << endl;

}

else if (symbol == 'Y')

{

transform\_Y(str);

str.length() == 0 ? cout << " В результате преобразования получилась пустая строка\n" : cout << str << endl;

}

else

{

cout << " Нет операции для введённого символа управления\n";

}

}

else

{

cout << " Пустая строка\n";

}

break;

case 4:

return 0;

default:

cout << " Нет такого пункта\n";

}

}

return 0;

}

## 4.3 ВРЕМЯ ВЫПОЛНЕНИЯ ФУНКЦИИ, РЕАЛИЗУЮЩЕЙ ЗАДАЧУ

С помощью функций библиотеки chrono была оценена практическая временная сложность работы алгоритмов функций transform\_N и transform\_Y. В качестве входных данных была сгенерирована произвольная строка из букв и цифр длиною 10000 символов. Для класса string были использованы часы с наименьшим временем тика high\_resolution\_clock.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 11- Оценка практической временной сложности transform\_N для класса string

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 12 - Результат оценки времени работы transform\_N для класса string

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 13 - Оценка практической временной сложности transform\_Y для класса string

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 14 - Результат оценки времени работы transform\_Y для класса string

## 4.4 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 15 – Результат теста 1.2

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 16 – Результат теста 2.2

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 17 – Результат теста 2.3

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 18 – Результат теста 2.4

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 19 – Результат теста 2.5

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 20 – Результат теста 2.6

# 5 ВЫВОДЫ

В ходе выполнения практической работы были получены навыки в использовании средств языка C и C++ для реализации алгоритмов обработки текстовых данных. Был изучен класс <string>, а также функции библиотеки <string.h>, которая работает с нуль – терминированными строками.

# 6 ИСПОЛЬЗУЕМЫЕ ИСТОЧНИКИ

1. Учебно-методическое пособие СиАОД (часть 1)
2. Приложение к практическим работам – СДО (online-edu.mirea.ru)