ГУАП

КАФЕДРА № 42

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доцент |  |  |  | В. А. Кузнецов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 2.1 |
| ПРОСТОЙ АЛГОРИТМ ОБРАБОТКИ СТРОК |
| по курсу: |
| ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | 4326 |  |  |  | Г. С. Томчук |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2024

СОДЕРЖАНИЕ

[1 Постановка задачи 3](#__RefHeading___Toc801_316780651)

[2 Схема алгоритма решения 4](#__RefHeading___Toc809_316780651)

[3 Полное описание реализованной функции 5](#__RefHeading___Toc807_316780651)

[3.1 extract\_words\_with\_digits 5](#__RefHeading___Toc2172_3453699567)

[3.2 contains\_digit 5](#__RefHeading___Toc2178_3453699567)

[3.3 free\_words 6](#__RefHeading___Toc2176_3453699567)

[3.4 main 6](#__RefHeading___Toc2174_3453699567)

[4 Листинг программы 7](#__RefHeading___Toc805_316780651)

[5 Несколько тестов работы программы 8](#__RefHeading___Toc803_316780651)

1. Постановка задачи

Задача: реализовать программную функцию на языке C/С++, выполняющую поставленную задачу. Вариант задания, пример входных и выходных данных представлен в таблице 1. Глобальные параметры использовать запрещено; допустимо использование дополнительных функций. Обработку строк необходимо осуществлять как массив символов, без использования специальных библиотек для работы со строками.

Таблица 1 – Вариант

|  |  |  |  |
| --- | --- | --- | --- |
| N | Текст задания | Вход | Выход |
| 8 | Дана строка. Слово есть последовательность любых символов, разделенных пробелом. Сохранить отдельно все слова, содержащие цифры. | "К2у7 Smt g6 PH 4a" | "К2у7", "g6", "4a" |

1. Схема алгоритма решения

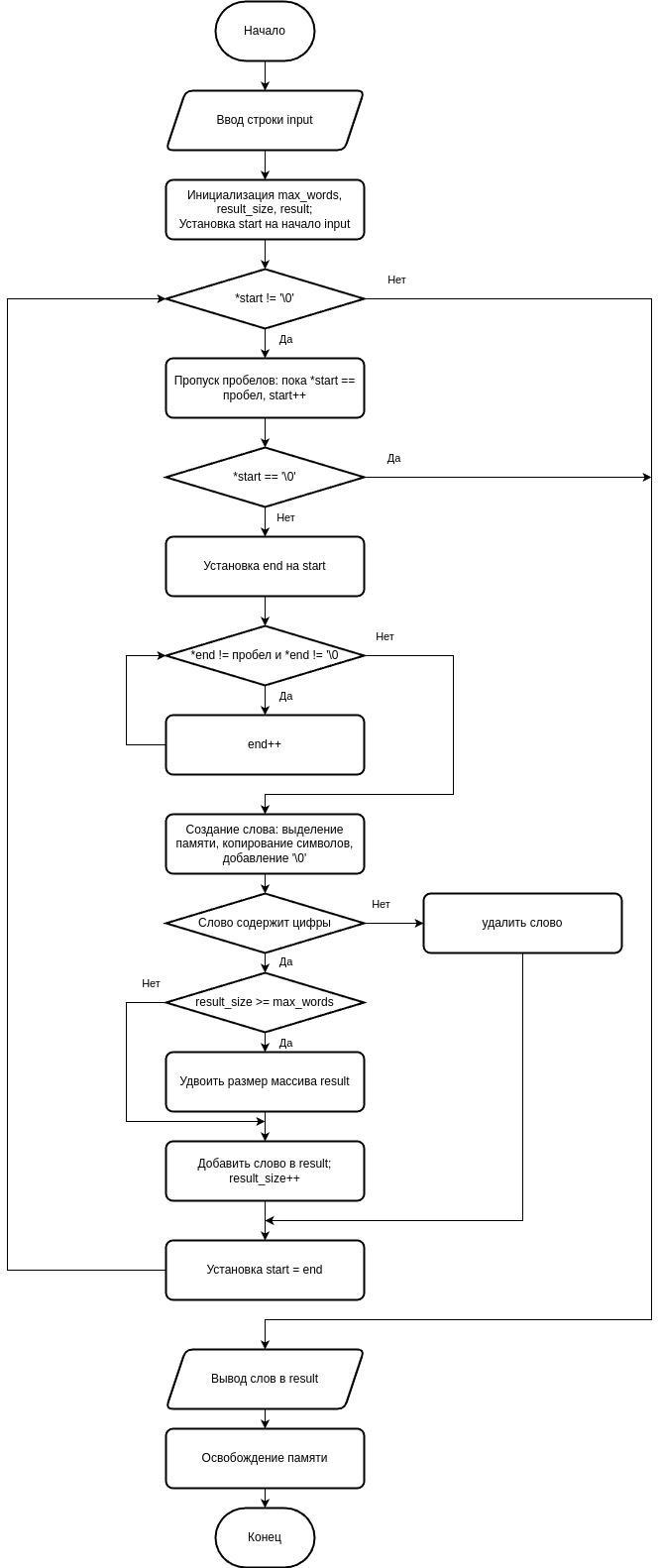


Рисунок 1 – Блок-схема алгоритма

1. Полное описание реализованной функции
   1. extract\_words\_with\_digits

Функция extract\_words\_with\_digits извлекает из строки все слова, содержащие цифры, и сохраняет их в массиве. Принимает следующие аргументы:

1. const char \*input: указатель на исходную строку.
2. char \*\*&result: ссылка на указатель массива строк, в который будут сохранены слова, содержащие цифры.
3. size\_t &result\_size: ссылка на переменную, в которую будет записано количество найденных слов.

Работа функции происходит следующим образом:

1. Инициализация переменных: начальный размер массива слов (max\_words), указатель на массив строк (result), и размер массива (result\_size).
2. Указатель start устанавливается на начало входной строки.
3. Цикл проходит по всей строке, пропуская начальные пробелы.
4. Нахождение конца текущего слова (до первого пробела).
5. Копирование слова в новый массив символов.
6. Проверка, содержит ли слово цифры с помощью contains\_digit.
7. Если слово содержит цифры, добавляем его в массив result.
8. При необходимости увеличиваем размер массива result.
9. Переход к следующему слову.
   1. contains\_digit

Функция contains\_digit проверяет, содержит ли переданное слово хотя бы одну цифру. Принимает следующие аргументы:

1. const char \*word: указатель на строку (слово), которую нужно проверить.

Работа функции происходит следующим образом:

1. Инициализация указателя на начало строки.
2. Цикл проходит по каждому символу строки.
3. Проверка, является ли текущий символ цифрой.
4. Если символ - цифра, возвращает true.
5. Если цикл завершился без нахождения цифры, возвращает false.
   1. free\_words

Функция free\_words освобождает память, выделенную под массив строк. Принимает следующие аргументы:

1. char \*\*words: указатель на массив строк.
2. size\_t size: количество строк в массиве.

Работа функции происходит следующим образом:

1. Цикл проходит по каждому элементу массива words.
2. Освобождает память, выделенную под каждое слово.
3. Освобождает память, выделенную под массив указателей.
   1. main
4. Объявление массива input для ввода строки.
5. Ввод строки от пользователя.
6. Объявление указателя words\_with\_digits и переменной words\_with\_digits\_size.
7. Вызов extract\_words\_with\_digits для извлечения слов, содержащих цифры.
8. Вывод слов, содержащих цифры, на экран.
9. Освобождение памяти, выделенной под массив слов с цифрами, с помощью free\_words.
10. Возвращение 0 для обозначения успешного завершения программы.
11. Листинг программы

Листинг 1

#include <iostream>  
  
bool contains\_digit(const char \*word) {  
 while (\*word) {  
 if (std::isdigit(\*word))  
 return true;  
 ++word;  
 }  
 return false;  
}  
  
void extract\_words\_with\_digits(const char \*input, char \*\*&result, size\_t &result\_size) {  
 size\_t max\_words = 8; // Начальный размер массива слов  
 result = new char \*[max\_words];  
 result\_size = 0;  
  
 const char \*start = input;  
 const char \*end;  
  
 while (\*start) {  
 // Пропуск пробелов в начале  
 while (\*start && std::isspace(\*start))  
 ++start;  
  
 if (!\*start) break;  
  
 // Находим конец слова  
 end = start;  
 while (\*end && !std::isspace(\*end)) {  
 ++end;  
 }  
  
 // Копирование слова в отдельный массив  
 long length = end - start;  
 char \*word = new char[length + 1];  
 for (size\_t i = 0; i < length; ++i)  
 word[i] = start[i];  
 word[length] = '\0';  
  
 // Проверка, содержит ли слово цифры  
 if (contains\_digit(word)) {  
 if (result\_size >= max\_words) {  
 // Увеличиваем размер массива  
 max\_words \*= 2;  
 char \*\*new\_result = new char \*[max\_words];  
 for (size\_t i = 0; i < result\_size; ++i)  
 new\_result[i] = result[i];  
 delete[] result;  
 result = new\_result;  
 }  
 result[result\_size++] = word;  
 } else {  
 delete[] word;

Продолжение листинга 1

}  
  
 start = end;  
 }  
}  
  
void free\_words(char \*\*words, size\_t size) {  
 for (size\_t i = 0; i < size; ++i)  
 delete[] words[i];  
 delete[] words;  
}  
  
int main() {  
 char input[256];  
 std::cout << "Введите строку: ";  
 std::cin.getline(input, 256);  
  
 char \*\*words\_with\_digits;  
 size\_t words\_with\_digits\_size;  
  
 extract\_words\_with\_digits(input, words\_with\_digits, words\_with\_digits\_size);  
  
 for (size\_t i = 0; i < words\_with\_digits\_size; ++i)  
 std::cout << "\"" << words\_with\_digits[i] << "\" ";  
  
 // Освобождаем память  
 free\_words(words\_with\_digits, words\_with\_digits\_size);  
  
 return 0;  
}

1. Несколько тестов работы программы

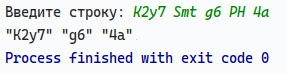


Рисунок 2

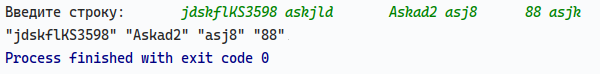


Рисунок 3

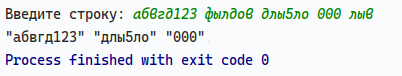


Рисунок 4