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 exclude\_max 5](#__RefHeading___Toc2172_3453699567)

[3.2 read\_array 5](#__RefHeading___Toc2178_3453699567)

[3.3 write\_array 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

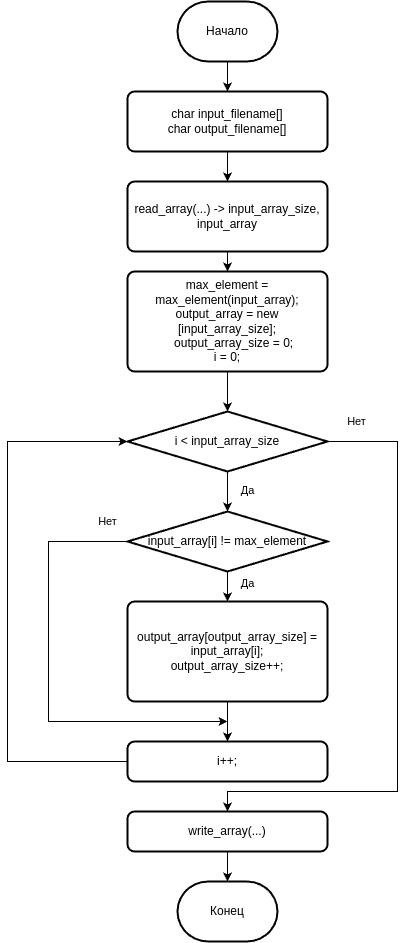


Рисунок 1 – Блок-схема алгоритма

1. Полное описание реализованной функции
   1. exclude\_max

Функция exclude\_max формирует новый массив, исключив все элементы с максимальным значением из входного массива. Принимает следующие аргументы:

1. const long \*input\_array: указатель на входной массив.
2. size\_t input\_array\_size: размер входного массива.
3. long \*&output\_array: ссылка на указатель, который будет указывать на выделенный динамический массив, содержащий элементы, исключая максимальные.
4. size\_t &output\_array\_size: ссылка на переменную, в которую будет записан размер нового массива.

Работа функции происходит следующим образом:

1. Находит максимальный элемент в массиве input\_array.
2. Выделяет память для массива output\_array размером input\_array\_size.
3. Инициализирует output\_array\_size значением 0.
4. Копирует элементы из массива input\_array в output\_array, исключая элементы с максимальным значением.
5. Увеличивает output\_array\_size при каждом добавлении элемента в output\_array.
   1. read\_array

Функция read\_array считывает массив из текстового файла. Принимает следующие аргументы:

1. const char \*filename: путь к файлу, из которого будет считываться массив.
2. long \*&array: ссылка на указатель, который будет указывать на выделенный динамический массив, содержащий считанные элементы.
3. size\_t &size: ссылка на переменную, в которую будет записан размер массива.

Работа функции происходит следующим образом:

1. Открывает файл с именем filename для чтения.
2. Если файл не удается открыть, выводит сообщение об ошибке и завершает программу.
3. Считывает размер массива из файла и записывает его в size.
4. Выделяет память для массива array размером size.
5. Считывает элементы массива из файла в array.
6. Закрывает файл.
   1. write\_array

Функция write\_array записывает массив в текстовый файл. Принимает следующие аргументы:

1. const char \*filename: путь к файлу, в который будет записан массив.
2. long \*array: указатель на массив, который будет записан в файл.
3. size\_t &size: ссылка на переменную, содержащую размер массива.

Работа функции происходит следующим образом:

1. Открывает файл с именем filename для записи.
2. Если файл не удается открыть, выводит сообщение об ошибке и завершает программу.
3. Записывает размер массива size в файл.
4. Записывает элементы массива array в файл.
5. Закрывает файл.
   1. main
6. Задает пути к входному и выходному файлам.
7. Объявляет указатели и переменные для входного и выходного массивов.
8. Считывает входной массив из файла с помощью read\_array.
9. Формирует выходной массив, исключая максимальные элементы, с помощью exclude\_max.
10. Записывает выходной массив в файл с помощью write\_array.
11. Освобождает выделенную память для массивов.
12. Возвращает 0, сигнализируя об успешном завершении программы.
13. Листинг программы

Листинг 1

#include <iostream>  
#include <fstream>  
#include <algorithm>  
  
void read\_array(const char \**filename*, long \*&*array*, size\_t &*size*) {  
 std::ifstream input\_file(*filename*);  
 if (!input\_file.is\_open()) {  
 std::**cerr** << "Ошибка открытия файла!" << std::endl;  
 exit(1);  
 }  
  
 input\_file >> *size*; // Читаем размерность массива  
  
 *array* = new long[*size*];  
  
 // Считываем элементы массива из файла  
 for (size\_t i = 0; i < *size*; i++)  
 input\_file >> *array*[i];  
  
 input\_file.close();  
}  
  
void write\_array(const char \**filename*, long \**array*, size\_t &*size*) {  
 std::ofstream output\_file(*filename*);  
 if (!output\_file.is\_open()) {  
 std::**cerr** << "Ошибка открытия файла!" << std::endl;  
 exit(1);  
 }  
  
 // Записываем размерность и элементы выходного массива в файл  
 output\_file << *size* << std::endl;  
 for (size\_t i = 0; i < *size*; i++)  
 output\_file << *array*[i] << " ";  
  
 output\_file.close();  
}  
  
void exclude\_max(const long \**input\_array*, size\_t *input\_array\_size*, long \*&*output\_array*, size\_t &*output\_array\_size*) {  
 // Находим максимальный элемент в массиве  
 long max\_element = \*std::max\_element(*input\_array*, *input\_array* + *input\_array\_size*);  
  
 *output\_array* = new long[*input\_array\_size*];  
 *output\_array\_size* = 0;  
  
 // Проходим по массиву и добавляем элементы в output\_array, исключая максимальный элемент  
 for (size\_t i = 0; i < *input\_array\_size*; i++)  
 if (*input\_array*[i] != max\_element)  
 *output\_array*[*output\_array\_size*++] = *input\_array*[i];  
}

Продолжение листинга 1

int main() {  
 char input\_filename[100] = "/home/grigorijtomczuk/Desktop/suai/op/lab1.3/input.txt";  
 char output\_filename[100] = "/home/grigorijtomczuk/Desktop/suai/op/lab1.3/output.txt";  
  
 long \*input\_array;  
 size\_t input\_array\_size;  
  
 long \*output\_array;  
 size\_t output\_array\_size;  
  
 read\_array(input\_filename, input\_array, input\_array\_size);  
 exclude\_max(input\_array, input\_array\_size, output\_array, output\_array\_size);  
 write\_array(output\_filename, output\_array, output\_array\_size);  
  
 delete[] input\_array;  
 delete[] output\_array;  
  
 return 0;  
};

1. Несколько тестов работы программы

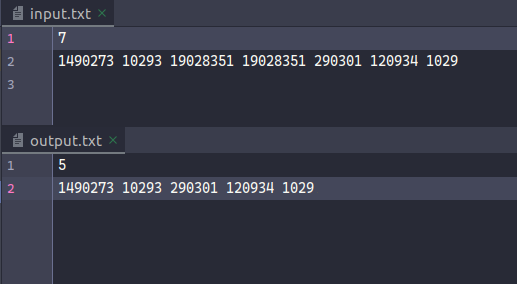


Рисунок 2

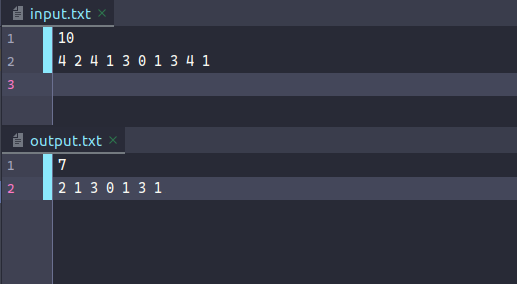


Рисунок 3

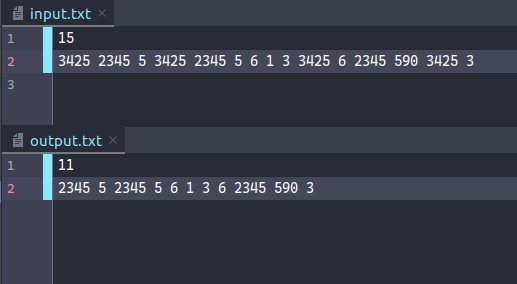


Рисунок 4