МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт (технический университет)»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| УГНС | | 09.00.00 | Информатика и вычислительная техника | | |
| Направление подготовки | | 09.03.01 | Информатика и вычислительная техника | | |
| Направленность (профиль) | |  | Системы автоматизированного проектирования | | |
| Форма обучения | |  | Очная | | |
|  | |  |  | | |
| Факультет | |  | Информационных технологий и управления | | |
| Кафедра | |  | Систем автоматизированного проектирования и управления | | |
| Учебная дисциплина | |  | Программирование | | |
| Курс | I | | | Группа | 404 |

**Отчёт по лабораторной работе № 2**

**Вариант № 3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исполнитель: |  |  |  |  |
| обучающийся группы 404 |  |  |  | Стариков Данил Евгеньевич |
|  |  | (дата, подпись) |  |  |
|  |  |  |  |  |
| Проверили: |  |  |  | Корниенко Иван Григорьевич |
|  |  | (дата, подпись) |  | Федин Алексей Константинович |

**Содержание**

[1 Постановка](#_heading=h.gjdgxs) задачи [3](#_heading=h.gjdgxs)

[2 Исходные](#_heading=h.gjdgxs) данные [3](#_heading=h.gjdgxs)

[3 Особые](#_heading=h.30j0zll) ситуации [3](#_heading=h.30j0zll)

[4 Математические](#_heading=h.1fob9te) методы и алгоритмы решения задачи [3](#_heading=h.1fob9te)

[5 Формат](#_heading=h.3znysh7) предоставления данных [3](#_heading=h.3znysh7)

[6 Блок](#_heading=h.2et92p0)-схема решения задачи6

[7 Описание](#_heading=h.tyjcwt) хода работы программы7

[8 Результат](#_heading=h.3dy6vkm) работы программы8

[9](#_heading=h.1t3h5sf) Исходный текст программы9

**1 Постановка задачи**

### Напишите программу для перестановки элементов массива так, чтобы все чётные элементы оказались в левой части массива, а все нечётные – в правой.

### 2 Исходные данные

Исходными данными является набор чисел произвольной длины.

### 3 Особые ситуации

Необходимо рассмотреть следующие особые ситуации:

* Последовательность из одного числа
* Последовательность из двух чисел, поставленных неверно.

### 4 Математические методы и алгоритмы решения задач

Для решения задачи был написан алгоритм на базе алгоритма «Сортировка пузырьком».

### 5 Форматы представления данных

|  |
| --- |
| Таблица 1 – Класс array<T>, представляющий массив типа T. |
| |  |  |  | | --- | --- | --- | | **Имя переменной** | **Тип** | **Описание** | | \_data | T[]\* | Внутренний динамический массив | | \_length | int | Длина массива | | **Имя метода** | **Тип** | **Описание** | | Array | - | Конструктор | | ~array | - | Деструктор | | erase | void | Очистить массив | | copy | array<T>\* | Копировать массив | | operator[] | T& | Перегрузка оператора [] | | get\_length | int | Получить длину массива | | print | void | Напечатать массив | | operator= | array<T>& | Перегрузка оператора = | | operator== | bool | Перегрузка оператора == | | operator!= | bool | Перегрузка оператора != | |

|  |
| --- |
| Таблица 2 – Класс tests, представляющий набор тестов. |
| |  |  |  | | --- | --- | --- | | **Имя переменной** | **Тип** | **Описание** | | length | int | Длина набора | | test\_data | array<int> | Последовательность исходных массивов | | results | array<int> | Последовательность массивов, ожидаемых на выходе | |

|  |
| --- |
| Таблица 3 – Собственные классы исключений. Наследуют от класса exception и отличаются только контекстом вызова. |
| |  |  | | --- | --- | | **Имя** | **Описание** | | ReadNumError | Ошибка при чтении числа | | FileNotExistError | Ошибка: файл не существует | | FileReadingError | Ошибка при чтении файла | | FileNotOpenError | Ошибка: не удалось открыть файл | | EOFError | Исключение: конец файла | | WriteError | Ошибка записи | | FileAlreadyExistError | Ошибка: файл уже существует | | Invalid\_array\_size | Ошибка: неверный размер массива | |
| Таблица 4 – Класс IInput – интерфейс, реализующий ввод. |
| |  |  |  | | --- | --- | --- | | **Имя переменной** | **Тип** | **Описание** | | stream | istream\* | Поток ввода | | **Имя метода** | **Тип** | **Описание** | | \_get\_string | string | Получить строку | | \_get\_num | T | Получение числа типа T | | get\_array | array<T> | Получение объекта класса array<T> | |

|  |
| --- |
| Таблица 5 – Класс ConsoleInput, реализующий интерфейс IInput, отвечает за ввод из консоли. |
| |  |  |  | | --- | --- | --- | | **Имя метода** | **Тип** | **Описание** | | \_get\_num | T | Получение числа типа T в диапазоне от min до max | | get\_num | T | Получение числа типа T | | TestSet | - | Конструктор | | ~TestSet | - | Деструктор | |

|  |
| --- |
| Таблица 6 – Класс FileInput, реализующий интерфейс IInput, отвечает за ввод из файла |
| |  |  |  | | --- | --- | --- | | **Имя переменной** | **Тип** | **Описание** | | \_filepath | string | Путь к файлу | | **Имя метода** | **Тип** | **Описание** | | \_eof\_exc | void | Бросить исключение EOFError | | \_is\_regular\_file | bool | Является ли файл «обычным» | | \_get\_num | T | Получение числа типа T | | is\_file\_exist | bool | Существует ли файл | | FileInput | - | Конструктор | | get\_string | string | Получение строки | | get\_num | T | Получение числа типа T | | get\_num | T | Получение числа типа T в диапазоне от min до max | | ~FileInput | - | Деструктор | |

|  |
| --- |
| Таблица 7 – Класс IOutput – интерфейс, отвечающий за вывод данных . |
| |  |  |  | | --- | --- | --- | | **Имя переменной** | **Тип** | **Описание** | | stream | ostream\* | Поток вывода | | **Имя метода** | **Тип** | **Описание** | | \_print | void | Вывод информации | | print\_arr | void | Вывод массива | |

|  |
| --- |
| Таблица 8 – Класс ConsoleOutput, реализующий интерфейс IOutput, отвечает за вывод в консоль. |
| |  |  |  | | --- | --- | --- | | **Имя метода** | **Тип** | **Описание** | | ConsoleOutput | - | Конструктор | | print | bool | Вывод информации | |

|  |
| --- |
| Таблица 9 – Класс FileOutput, реализующий интерфейс IOutput, отвечает за вывод в файл. |
| |  |  |  | | --- | --- | --- | | **Имя переменной** | **Тип** | **Описание** | | \_filepath | string | Путь к файлу | | **Имя метода** | **Тип** | **Описание** | | is\_file\_exist | bool | Существует ли файл | | FileOutput | - | Конструктор | | try\_to\_overwrite | bool | Попытаться перезаписать существующийфайл | | print | bool | Вывод информации | | ~FileOutput | - | Деструктор | |
| Таблица 9 – пространство имён program, объединяющее методы работы программы. |
| |  |  |  | | --- | --- | --- | | **Имя метода** | **Тип** | **Описание** | | greeting | void | Приветствие | | read\_array\_from\_file | bool | Прочитать массив из файла | | read\_array\_from\_console | bool | Прочитать массив из консоли | | sort\_array | int | Сортировать массив | | ask\_for\_input\_save | void | Спросить, нужно ли сохранять ввод | | print\_array\_to\_file | void | Печать массива в файл | | save\_output | void | Сохранить вывод | | ask\_for\_output\_save | void | Спросить, нужно ли сохранять вывод | | print\_arrays\_to\_console | void | Вывод массивов в консоль | | fill\_arr\_with\_random\_nums | void | Заполнить массив случайными числами | | ask\_for\_tests | void | Спросить, нужно ли проводить тесты | | loop | void | Итерация программы | | run | void | Запустить программу | |

### 6 Блок-схема алгоритма определения наличия у прямоугольников общей области

Ниже представлена блок схема решения задачи определения наличия у прямоугольников общей области.



Рисунок 1 – Блок-схема определения наличия у прямоугольников общей

области

### 7 Описание хода выполнения лабораторной работы

1. В ходе лабораторной работы было создано решение (solution) в интегрированной среде разработки Microsoft Visual Studio C++ 2019. В нем был создан проект.
2. В ходе выполнения программы была выявлена проблема: при создании файла с именем «con» программа зависла. После изучения данной проблемы было обнаружено, что существуют такие слова, которые нельзя использовать для именования файлов. Для устранения этой проблемы было принято решение использовать функцию is\_regular\_file() из библиотеки filesystem.
3. На базе «сортировки пузырьком» был разработал алгоритм, меняющий порядок элементов.
4. В ходе выполнения программы было выявлено, что программа при сохранении данных в файл перезаписывала его, не предупредив пользователя. Был написан метод try\_overwrite\_file(), который уточняет у пользователя, стоит ли перезаписывать файл. Если пользователь соглашается, то файл перезаписывается, если нет, то пользователь вводит новое название файла. Процесс продолжается до тех пор, пока пользователь не согласится с перезаписью, или не введет новый файл.

### 8 Результаты работы программы

В результате вычислений программа выводит, существует ли у заданных прямоугольников общая область. Результат выводится в консоль, но также есть возможность сохранить его в файл. Ниже представлены примеры результатов работы программы.

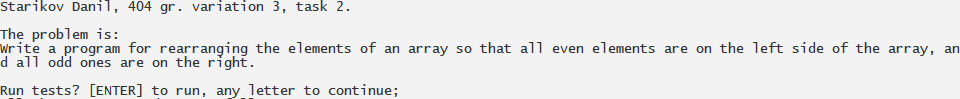


Рисунок 2 – Приветствие программы

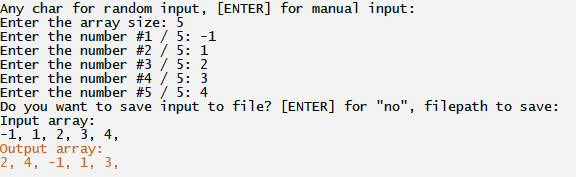


Рисунок 3 – Пример работы программы, ввод из консоли

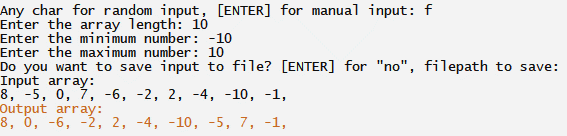


Рисунок 4 – Пример работы программы, случайный ввод



Рисунок 5 – Пример работы программы, когда файл нельзя создать

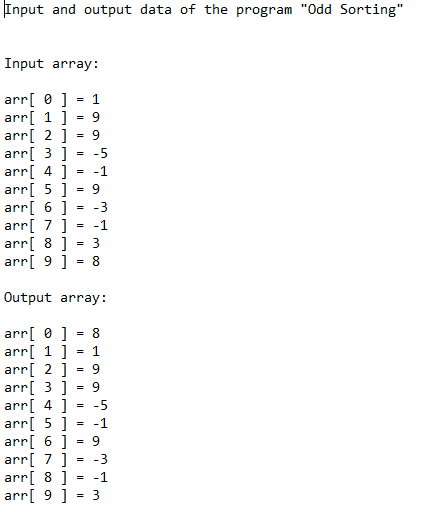


Рисунок 6 – Пример сохранения результата работы программы в файле

### 9 Исходный текст программы

[--- Начало main.cpp ---]

﻿#include "program.h"

int main()

{

srand(static\_cast<unsigned int>(time(nullptr)));

program::run();

return 0;

}

[--- Конец main.cpp ---]

[--- Начало program.cpp ---]

﻿#include "program.h"

#include "array.h"

#include "Output.h"

#include "tests.h"

ConsoleOutput console\_out{};

const std::string greeting\_string =

"Starikov Danil, 404 gr. variation 3, task 2.\n\n"

"The problem is: \n"

"Write a program for rearranging the elements of an array so that all even elements are on the left side of the array, and all odd ones are on the right.\n";

// Приветствие

void program::greeting() {

std::cout << greeting\_string << std::endl;

}

// Прочитать массив из файла

bool program::read\_array\_from\_file(array<int>& arr, FileInput& input) {

try {

const int length = input.get\_num(1, INT\_MAX);

arr = \*input.get\_array<int>(length);

}

catch (ReadNumError&) {

return false;

}

catch (EOFError&) {

return false;

}

catch (std::invalid\_argument&) {

return false;

}

catch (FileNotOpenError&) {

return false;

}

catch (std::range\_error&) {

return false;

}

return true;

}

bool program::read\_array\_from\_console(array<int>& arr) {

ConsoleInput input{};

while (true)

{

std::cout << "Enter the array size: ";

int length;

try

{

length = input.get\_num<int>(1, INT\_MAX);

}

catch (std::range\_error&)

{

continue;

}

catch (ReadNumError&)

{

continue;

}

arr = array<int>(length);

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

{

std::cout << "Enter the number #" << i + 1 << " / " << length << ": ";

arr[i] = input.get\_num<int>();

}

return true;

}

}

int program::sort\_array(array<int>& arr)

{

int counter = 0;

for (int \_ = 0; \_ < arr.get\_length(); \_++)

{

for (int i = 0; i < arr.get\_length() - 1; i++)

{

if (arr[i] % 2 != 0 && arr[i + 1] % 2 == 0)

{

const auto temp = arr[i];

arr[i] = arr[i + 1];

arr[i + 1] = temp;

counter++;

}

}

}

return counter;

}

void program::ask\_for\_input\_save(array<int>& arr)

{

std::cout << "Do you want to save input to file? [ENTER] for \"no\", filepath to save: ";

std::string choice;

getline(std::cin, choice);

if (!choice.empty())

{

FileOutput file = open\_file\_for\_output(choice);

file.print(arr.get\_length());

file.print\_array(arr);

std::cout << "Successful save." << std::endl;

}

}

void program::print\_array\_to\_file(array<int>& arr, FileOutput& file)

{

for (int i = 0; i < arr.get\_length(); i++)

{

file.print("arr[ ", "");

file.print(i, "");

file.print(" ] = ", "");

file.print(arr[i]);

}

}

// Сохранить вывод программы

void program::save\_output(const std::string& filepath, array<int>& input\_arr, array<int>& output\_arr) {

FileOutput file = open\_file\_for\_output(filepath);

file.print("Input and output data of the program \"Odd Sorting\"", "\n\n\n");

file.print("Input array: \n");

print\_array\_to\_file(input\_arr, file);

file.print("\nOutput array: \n");

print\_array\_to\_file(output\_arr, file);

std::cout << "Successful save." << std::endl;

}

// Спросить, нужно ли сохранить вывод программы в файл, и сохранить, если нужно

void program::ask\_for\_output\_save(array<int>& input\_arr, array<int>& output\_arr) {

std::string filepath;

std::cout << "Do you want to save output? Filepath to save, [ENTER] to continue: ";

getline(std::cin, filepath);

if (!filepath.empty()) save\_output(filepath, input\_arr, output\_arr);

}

void program::print\_arrays\_to\_console(array<int>& input\_arr, array<int>& output\_arr)

{

std::cout << "Input array: " << std::endl;

console\_out.print\_array(input\_arr, ", ");

std::cout << std::endl;

change\_color(console\_color::cyan, console\_color::black);

std::cout << "Output array: " << std::endl;

console\_out.print\_array(output\_arr, ", ");

std::cout << std::endl << std::endl;

change\_color();

}

void program::fill\_arr\_with\_random\_nums(array<int>& arr)

{

ConsoleInput input{};

std::cout << "Enter the array length: ";

const int length = input.get\_num(1, INT\_MAX);

std::cout << "Enter the minimum number: ";

const int min = input.get\_num(-1000, 1000);

std::cout << "Enter the maximum number: ";

const int max = input.get\_num(min, 1000);

auto new\_arr = array<int>(length);

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

{

new\_arr[i] = min + rand() % (max - min + 1);

}

arr = new\_arr;

}

void program::ask\_for\_tests()

{

std::string choice;

bool are\_tests\_failed = false;

std::cout << "Run tests? [ENTER] to run, any letter to continue; ";

getline(std::cin, choice);

if (choice.empty())

{

auto [length, test\_data, results] = tests();

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

{

array<int>\* temp\_arr = test\_data[i].copy();

sort\_array(\*temp\_arr);

if (results[i] != \*temp\_arr)

{

std::cout << "Its an ERROR occurred while passing the tests!" << std::endl;

std::cout << "Test #" << i + 1 << " failed!" << std::endl << std::endl;

std::cout << "Input array: ";

console\_out.print\_array(\*temp\_arr, ", ");

change\_color(console\_color::red, console\_color::black);

std::cout << std::endl << std::endl << "Output array: ";

console\_out.print\_array(test\_data[i], ", ");

change\_color(console\_color::green, console\_color::black);

std::cout << std::endl << std::endl << "Expected array: ";

console\_out.print\_array(results[i], ", ");

change\_color();

std::cout << std::endl;

are\_tests\_failed = true;

}

}

if (!are\_tests\_failed) {

std::cout << "All the tests passed successfully!" << std::endl;

}

else

{

std::cout << "Be careful using the program!" << std::endl << std::endl;

}

}

}

void program::loop()

{

array<int> input\_arr{};

std::cout << "Choose input method. [ENTER] for console input, filepath for input from file: ";

std::string choice;

getline(std::cin, choice);

std::cout << std::endl;

if (choice.empty())

{

std::cout << "Any char for random input, [ENTER] for manual input: ";

getline(std::cin, choice);

if (choice.empty())

{

read\_array\_from\_console(input\_arr);

}

else

{

fill\_arr\_with\_random\_nums(input\_arr);

}

array<int>\* output\_arr = input\_arr.copy();

ask\_for\_input\_save(input\_arr);

sort\_array(\*output\_arr);

print\_arrays\_to\_console(input\_arr, \*output\_arr);

ask\_for\_output\_save(input\_arr, \*output\_arr);

delete output\_arr;

}

else

{

if (FileInput input = open\_file\_for\_input(choice); !read\_array\_from\_file(input\_arr, input)) {

std::cout << "Invalid file. Please choose another one." << std::endl;

return;

}

array<int>\* output\_arr = input\_arr.copy();

sort\_array(\*output\_arr);

print\_arrays\_to\_console(input\_arr, \*output\_arr);

ask\_for\_output\_save(input\_arr, \*output\_arr);

delete output\_arr;

}

}

void program::run() {

greeting();

ask\_for\_tests();

std::string choice;

do {

std::cout << std::endl;

loop();

std::cout << "[ENTER] to exit, any key to repeat;";

getline(std::cin, choice);

} while (!choice.empty());

std::cout << "Bye!" << std::endl;

}

[--- Конец program.cpp ---]