|  |  |
| --- | --- |
| https://lh7-us.googleusercontent.com/QuBaagabzZYLr7U3ZbYOtMusd5FbWWxIvMUTN8jrLHzSLMg534z9gXRTIG1Us4i_lOwmWlaBxKedNt-SQ26dm4WmyqwjGDmEO6z8GE3QrZosqvHM88J2EFeVf1u0GzyCZQlhWmp1Zeo85tKo4LJVXQ | МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)» |

Институт № 3 «Системы управления, информатика и электроэнергетика»

Кафедра 304 «Вычислительные машины, системы и сети»

**Отчет по лабораторной работе**

по учебной дисциплине «Информационные технологии»

на тему «Одномерные массивы»

Вариант №5

Руководители работы:

доц. каф.304, канд. физ.-мат. наук Чечиков Юрий Борисович

Секретарёв Виталий Евгеньевич

Исполнители:

обучающиеся группы М3О-119Б-24

Ефимов Ярослав Сергеевич

Тихонов Павел Сергеевич

Москва 2024

Содержание

[1. Задание 3](#_Toc182698085)

[2. Блок-схема 4](#_Toc182698086)

[3. Код программы 6](#_Toc182698087)

[4. Тесты программы 10](#_Toc182698088)

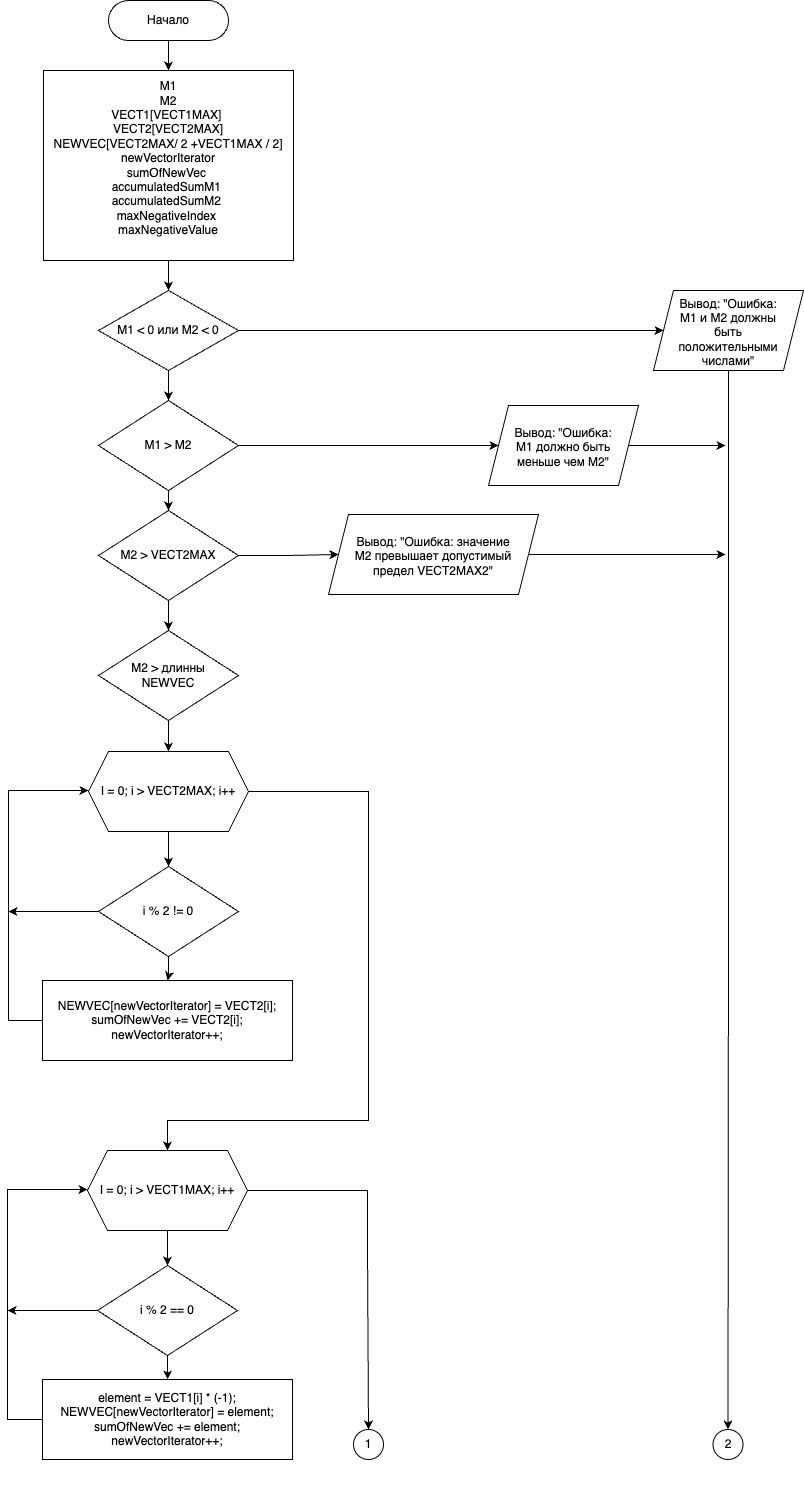
[4.1. Некорректные тесты 10](#_Toc182698089)

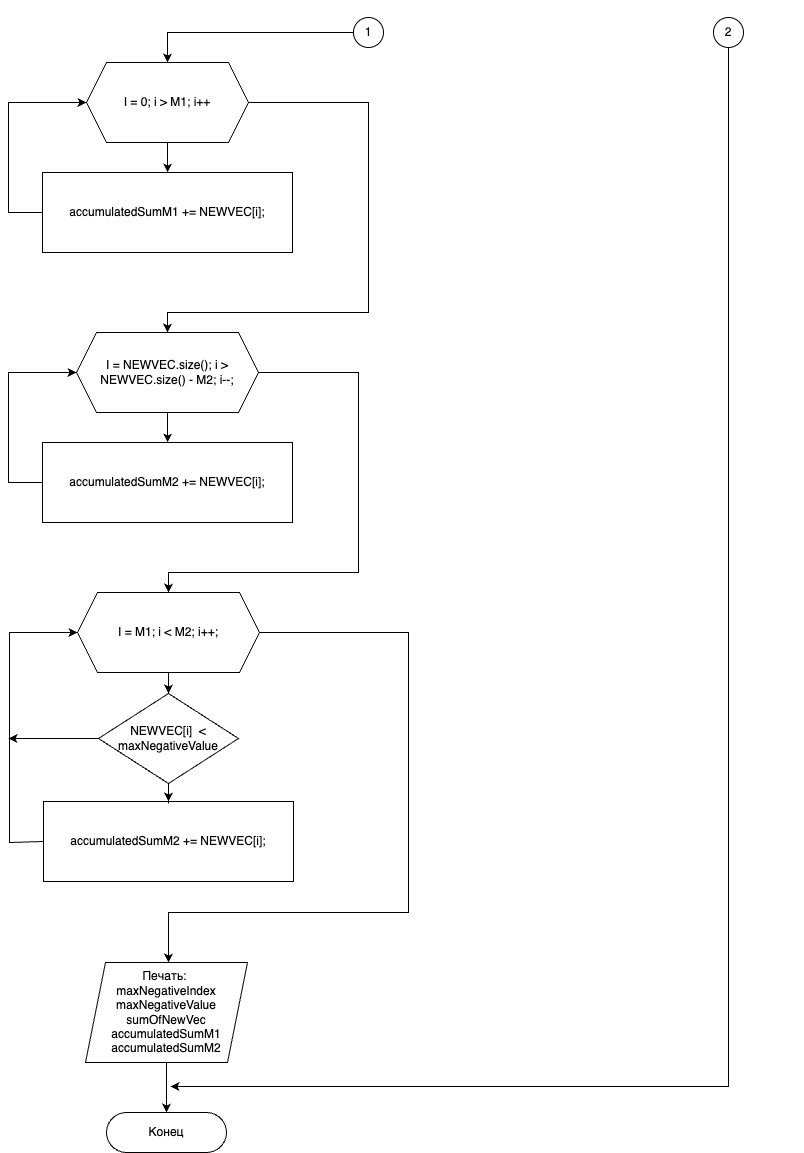
[4.2. Корректные тесты 14](#_Toc182698090)

[Вывод по работе 18](#_Toc182698091)

# Задание

# Блок-схема





# Код программы

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Курс КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ \*

\*----------------------------------------------------------------------------------\*

\* Project Type : Win64 Console Application \*

\* Project Name : Labs3 \*

\* File Name : Labs3.exe \*

\* Language : CPP, MSVS 2022 \*

\* Programmers : Тихонов Павел Сергеевич, Ефимов Ярослав Сергеевич, М3О-119БB-24 \*

\* Modifyed By : \*

\* Created : 19.11.2024 \*

\* Last Revision : 21.11.2024 \*

\* Comment(s) : Одномерные массивы \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <fstream>

#include <unistd.h>

#include <sstream>

struct Filenames {

std::string correct\_1;

std::string correct\_2;

std::string correct\_3;

std::string error\_empty;

std::string error\_invalid\_format;

std::string error\_negative\_params;

std::string uncorrected\_M1M2\_1;

std::string uncorrected\_M1M2\_2;

std::string uncorrected\_M1M2\_3;

std::string uncorrected\_VEC;

std::string uncorrected\_VEC1;

std::string uncorrected\_VEC2;

};

// Имена файлов для различных тестов

Filenames filenames = {

"test\_correct\_01.txt",

"test\_correct\_02.txt",

"test\_correct\_03.txt",

"test\_error\_empty.txt",

"test\_error\_invalid\_format.txt",

"test\_error\_negative\_params.txt",

"test\_uncorrected\_M1M2\_1.txt",

"test\_uncorrected\_M1M2\_2.txt",

"test\_uncorrected\_M1M2\_3.txt",

"test\_uncorrected\_VEC.txt",

"test\_uncorrected\_VEC1.txt",

"test\_uncorrected\_VEC2.txt"

};

// Имя файла для текущего теста

auto FNAME = "\_\_tests\_\_/" + filenames.correct\_3;

// Максимальные размеры массивов

const int VECT1MAX = 8;

const int VECT2MAX = 10;

// Коды ошибок

constexpr int NOT\_FOUND = 1;

constexpr int INVALID\_DATA = 2;

constexpr int ACCESS\_PERMISSION = 3;

int main() {

// Получение текущего рабочего каталога

char cwd[1024];

if (getcwd(cwd, sizeof(cwd)) != NULL) {

std::cout << "Текущий рабочий каталог: " << cwd << std::endl;

} else {

std::cerr << "Ошибка при получении текущего рабочего каталога" << std::endl;

return ACCESS\_PERMISSION;

}

// ------------------------

// ЧТЕНИЕ ДАННЫХ ИЗ ФАЙЛА

// ------------------------

// Установка локали

std::locale::global(std::locale("en\_US.UTF-8"));

std::wcout.imbue(std::locale("en\_US.UTF-8"));

const std::string CWDFN = std::string(cwd) + "/" + FNAME; // Путь к файлу

int M1, M2;

// Векторы для хранения данных

std::vector<double> VECT1(VECT1MAX);

std::vector<double> VECT2(VECT2MAX);

std::vector<double> NEWVEC(floor(VECT2MAX / 2) + ceil(VECT1MAX / 2));

// Итератор для нового вектора

size\_t newVectorIterator = 0;

// Сумма элементов нового вектора

double sumOfNewVec = 0;

// Накопленные суммы для M1 и M2

double accumulatedSumM1 = 0;

double accumulatedSumM2 = 0;

// Индекс и значение максимального отрицательного числа

int maxNegativeIndex = -1;

double maxNegativeValue = 0;

std::cout << "Чтение данных из файла: " << CWDFN << std::endl;

// Открытие файла для чтения std::ios::in указывает режим открытия файла(только чтение)

std::fstream fin(FNAME, std::ios::in);

if (!fin.is\_open())

{

// Файл не найден

std::cerr << "Файл \"" << CWDFN << "\"" << " не найден" << std::endl;

fin.close(); // закрыть файл

return NOT\_FOUND;

}

if (fin.peek() == EOF) // файл пуст

{

std::cerr << "Файл \"" << CWDFN << "\" пустой\n";

fin.close(); // закрыть файл

return INVALID\_DATA; // выход по ошибке

}

if(!(fin >> M1 >> M2)) {

std::cerr << "Ошибка: не удалось заполнить параметры M1 и M2" << std::endl;

return INVALID\_DATA;

}

/\*\*

\* Проверки необходимы для выполнения условий, указанных в пункте 5, где сказано, что M1 и M2

\* будут использоваться как индексы массивов. Следовательно, все положительные значения M1 и M2

\* допустимы, а отрицательные - нет.

\*

\* Согласно пункту 5, где говорится: "только в диапазоне индексов от M1 до M2", M1 должно быть

\* меньше M2. В противном случае длина диапазона будет отрицательной, что делает условие

\* нерелевантным.

\*

\* Также необходимо проверить, что M1 и M2 не превышают размер наибольшего массива (или индекс

\* последнего элемента массива). В противном случае индексы M1 и M2 могут выйти за пределы массива,

\* что приведет к критической ошибке.

\*\*/

if(M1 < 0 || M2 < 0) {

std::cerr << "Ошибка: M1 и M2 должны быть положительными числами" << std::endl;

return INVALID\_DATA;

}

if(M1 > M2) {

std::cerr << "Ошибка: M1 должно быть меньше чем M2" << std::endl;

return INVALID\_DATA;

}

/\*\*

\* VECT2MAX по услоивю задачи всегда больше чем VECT1MAX, поэтому учитывать только VECT2MAX.

\*\*/

if (M2 > VECT2MAX) {

std::cerr << "Ошибка: значение M2 превышает допустимый предел VECT2MAX" << std::endl;

return INVALID\_DATA;

}

/\*\*

\* Сумма четных и нечетный индексов масивов VECT1 и VECT2 равна.

\* Целой части от деления ( VECT2MAX / 2) + результат деления (VECT1MAX / 2), округлённый в большую сторону.

\* Из этого следует, что M1 и M2 не должны превышать размер массива NEWVEC.

\*\*/

if(M2 > NEWVEC.size()) {

std::cerr << "Ошибка: M1 и M2 должны быть меньше чем размер массива NEWVEC" << std::endl;

return INVALID\_DATA;

}

// Заполнение массивов VECT1 и VECT2 из файла CWDFN

for (auto& value : VECT1) {

if (!(fin >> value)) {

std::cerr << "Ошибка: не удалось заполнить VECT1" << std::endl;

fin.close();

return INVALID\_DATA;

}

}

for (auto& value : VECT2) {

if (!(fin >> value)) {

std::cerr << "Ошибка: не удалось заполнить VECT2" << std::endl;

fin.close();

return INVALID\_DATA;

}

}

// Проверка, что после считывания значений в fin что-то еще осталось

if (fin.peek() != EOF) {

std::cerr << "Ошибка: после считывания значений остались данные в файле: " << fin.peek() << std::endl;

fin.close();

return INVALID\_DATA;

}

fin.close();

// -----------------------------

// ОСНОВНОЙ АЛГОРИТМ ПРОГРАММЫ

// -----------------------------

// Формирование нового массива NEWVEC

for (size\_t i = 0; i < VECT2.size(); ++i) {

// Оставляем только нечетные элементы

if(i % 2 != 0) {

/\*\* Добавляем элемент в новый массив NEWVEC

\* и считаем сумму элементов \*\*/

NEWVEC[newVectorIterator] = VECT2[i];

sumOfNewVec += VECT2[i];

newVectorIterator++;

}

}

for (size\_t i = VECT1.size(); i-- > 0;) {

// Оставляем только четные элементы

if(i % 2 == 0) {

double element = VECT1[i] \* (-1);

/\*\* Добавляем элемент в новый массив NEWVEC

\* и считаем сумму элементов \*\*/

NEWVEC[newVectorIterator] = element;

// По условию задачи берем противоположный знак

sumOfNewVec += element;

newVectorIterator++;

}

}

// начиная с 0 индекса считаем сумму M1 элементов(инными словами сумму элементов от 0 до M1)

for(size\_t i = 0; i < M1; ++i) {

accumulatedSumM1 += NEWVEC[i];

}

// начиная с последнего индекса считаем сумму M2 элементов(инными словами сумму элементов от последнего индекса до M2)

for (size\_t i = NEWVEC.size(); i-- > NEWVEC.size() - M2;) {

accumulatedSumM2 += NEWVEC[i];

}

std::cout << "Сумма всех элементов массива [NEWVEC]: " << sumOfNewVec << "\n"

<< "Сумма первых " << M1 << " элементов: " << accumulatedSumM1 << "\n"

<< "Сумма последних " << M2 << " элементов: " << accumulatedSumM2 << std::endl;

// Поиск максимального отрицательного числа на промежутке индексов [M1, M2]

for (size\_t i = static\_cast<size\_t>(M1); i < M2; ++i) {

if(NEWVEC[i] < maxNegativeValue) {

maxNegativeValue = NEWVEC[i];

maxNegativeIndex = i;

}

}

std::cout << "Максимальное отрицательное число на промежутке индексов [" <<

M1 << " , " << M2 << "]: " <<

maxNegativeValue << "\n" << "Индекс максимального отрицательного числа в массиве [NEWVECT]: " <<

maxNegativeIndex << std::endl;

return 0;

}

# Тесты программы

## Некорректные тесты

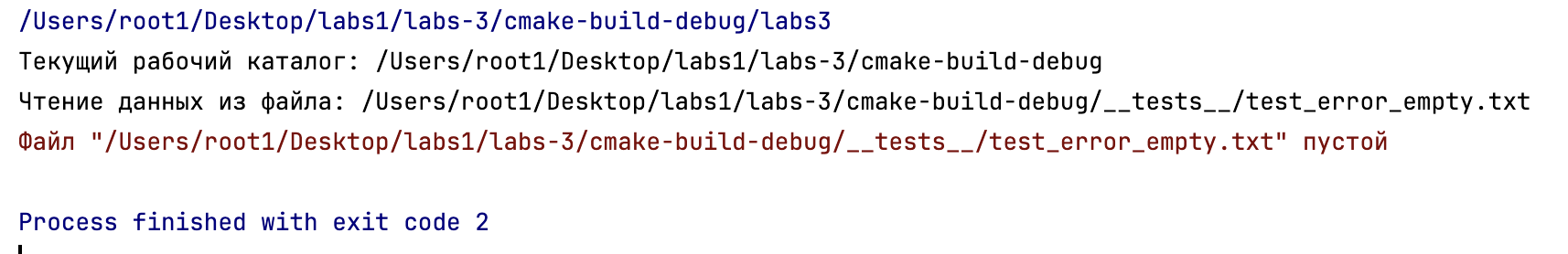
Тест 1

Цель теста: проверить корректность работы программы при обработке пустого файла.

Исходные данные:

Ожидаемый результат: “ Файл <абсолютный\_путь> пустой”

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибку не обнаружил.

Тест 2

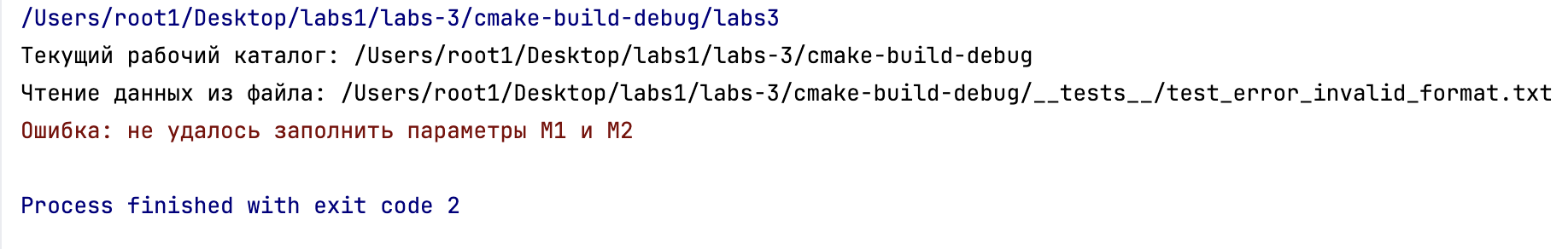
Цель теста: проверить корректность работы программы при указании файла с произвольным содержимым.

Исходные данные: M1=[1] M2=[2] VECT1=[1 2 3 4 5 6 7]

VECT2=[1 2 3 4 5 6 7 8 9]

Ожидаемый результат: “Ошибка: не удалось заполнить параметры M1 и M2”

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибку не обнаружил.

Тест 3

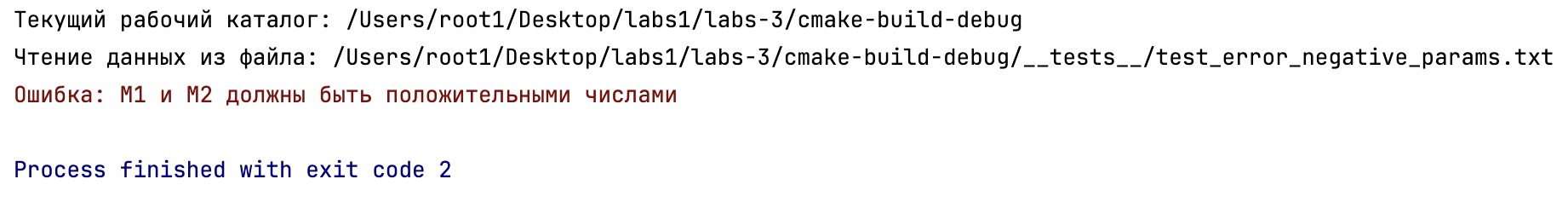
Цель теста: проверить корректность работы программы при указании аргументов M1 и M2 как отрицательных чисел.

Исходные данные: M1 = -5 M2 = -1 VECT1 = 1 2 3 4 5 6 7 8

VECT2 = 1 2 3 4 5 6 7 8 9 10

Ожидаемый результат: " Ошибка: M1 и M2 должны быть положительными числами"

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибку не обнаружил.

Тест 4

Цель теста: проверить корректность работы программы при указании аргументов M1 и M2, превышающих размер массива NEWVEC.

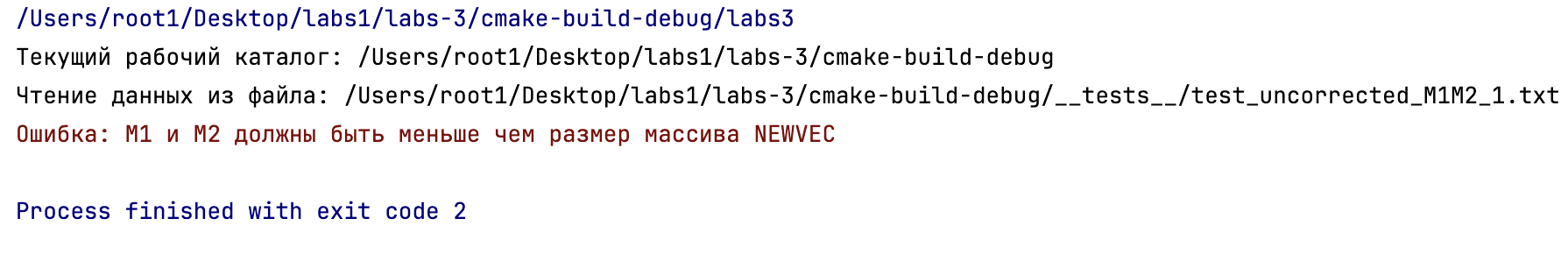
Исходные данные: M1 = 1 M2 = 10 VECT1 = 1 2 3 4 5 6 7 8

VECT2 = 1 2 3 4 5 6 7 8 9 10

Размер NEWVEC = 9

Ожидаемый результат: "Ошибка: M1 и M2 должны быть меньше, чем размер массива NEWVEC"

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибку не обнаружил.

Тест 5

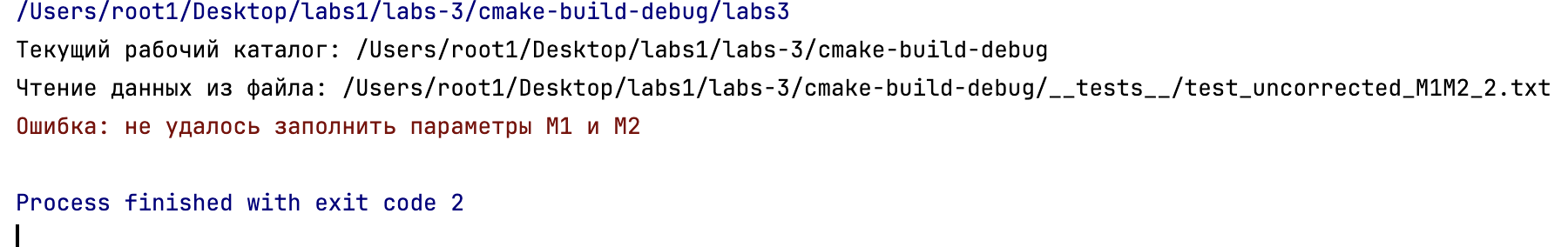
Цель теста: проверить корректность работы программы при указании аргумента M1 в виде буквенного значения вместо ожидаемого целого числа.

Исходные данные: M1 = a M2 = 9 VECT1 = 1 2 3 4 5 6 7 8

VECT2 = 1 2 3 4 5 6 7 8 9 10

Ожидаемый результат: "Ошибка: не удалось заполнить параметры M1 и M2"

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибку не обнаружил.

Тест 6

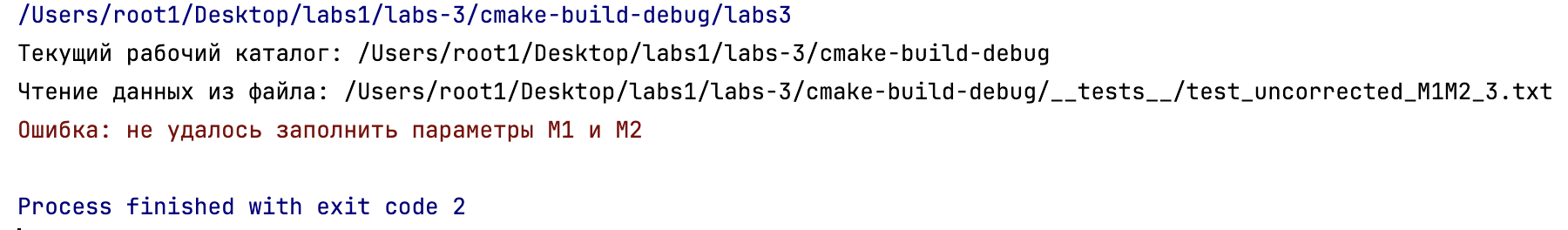
Цель теста: проверить корректность работы программы при указании аргумента M2 в виде буквенного значения вместо ожидаемого целого числа.

Исходные данные: M1 = 1 M2 = a VECT1 = 1 2 3 4 5 6 7 8

VECT2 = 1 2 3 4 5 6 7 8 9 10

Ожидаемый результат: "Ошибка: не удалось заполнить параметры M1 и M2"

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибку не обнаружил.

Тест 7

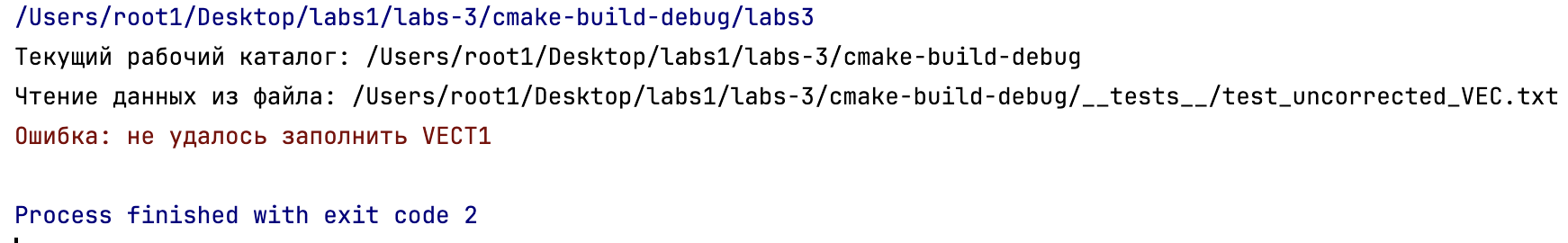
Цель теста: проверить корректность работы программы при недостаточном или некорректном содержимом файла, что приводит к невозможности заполнения массива VECT1.

Исходные данные: M1 = 4 M2 = 7 VECT1 = 1 2 3 4 5 b a x

VECT2 = 1 2 3 4 5 6 7 8 9 10

Ожидаемый результат: "Ошибка: не удалось заполнить VECT1"

Полученный результат:



Вывод: полученный результат совпал с ожидаемым. Тест ошибку не обнаружил.

Тест 8

Цель теста: проверить корректность работы программы при недостаточном или некорректном содержимом файла, что приводит к невозможности заполнения массива VECT2.

Исходные данные: M1 = 1 M2 = 6 VECT1 = 1 2 3 4 5 6 7 8

VECT2 = 1 2 3 4 5 6 7 8 9

Ожидаемый результат: "Ошибка: не удалось заполнить VECT2"

Полученный результат:

Изображение выглядит как текст, Шрифт, снимок экрана, алгебра

Автоматически созданное описание

Вывод: полученный результат совпал с ожидаемым. Тест ошибку не обнаружил.

## Корректные тесты

Тест 9

Цель теста: проверить корректность работы программы при корректных входных данных

Исходные данные: M1 = 1 M2 = 6 VECT1 = 1 2 3 4 5 6 7 8

VECT2 = 1 2 3 4 5 6 7 8 9 10

Ожидаемый результат:

Размер NEWVEC = 5 + 4 = 9

Нечетные индексы в VECT2: 1, 3, 5, 7, 9

VECT2[1] = 2

VECT2[3] = 4

VECT2[5] = 6

VECT2[7] = 8

VECT2[9] = 10

Четные индексы VECT1 в обратном порядке: 6, 4, 2, 0

-VECT1[6] = -7

-VECT1[4] = -5

-VECT1[2] = -3

-VECT1[0] = -1

NEWVEC = [2, 4, 6, 8, 10, -7, -5, -3, -1]

sumOfNewVec: 2 + 4 + 6 + 8 + 10 + (-7) + (-5) + (-3) + (-1) = 14

Сумма первых M1(1) элементов массива NEWVEC = 2

Cумма последних M2(6) элементов: 8 + 10 + (-7) + (-5) + (-3) + (-1) = 2

Поиск максимального отрицательного числа на промежутке индексов от M1(1) до M2(6): NEWVEC[5] = -7

Полученный результат:

Изображение выглядит как текст, снимок экрана, Шрифт, алгебра

Автоматически созданное описание

Вывод: полученный результат совпал с ожидаемым.

Тест 10

Цель теста: проверить корректность работы программы при частичной замене данных

Исходные данные: M1 = 4 M2 = 7 VECT1 = 1 2 3 4 5 6 7 8

VECT2 = 1 2 3 4 5 6 7 8 9 10

Ожидаемый результат:

Размер NEWVEC = 5 + 4 = 9

Нечетные индексы в VECT2: 1, 3, 5, 7, 9

VECT2[1] = 2

VECT2[3] = 4

VECT2[5] = 6

VECT2[7] = 8

VECT2[9] = 10

Четные индексы VECT1 в обратном порядке: 6, 4, 2, 0

-VECT1[6] = -7

-VECT1[4] = -5

-VECT1[2] = -3

-VECT1[0] = -1

NEWVEC = [2, 4, 6, 8, 10, -7, -5, -3, -1]

sumOfNewVec: 2 + 4 + 6 + 8 + 10 + (-7) + (-5) + (-3) + (-1) = 14

Сумма первых M1(4) элементов массива NEWVEC: 2 + 4 + 6 + 8 = 20

Cумма последних M2(7) элементов: 6 + 8 + 10 + (-7) + (-5) + (-3) + (-1) = 8

Поиск максимального отрицательного числа на промежутке индексов от M1(4) до M2(7): NEWVEC[5] = -7

Полученный результат:

Изображение выглядит как текст, снимок экрана, Шрифт, алгебра

Автоматически созданное описание

Вывод: полученный результат совпал с ожидаемым.

Тест 11

Цель теста: проверить корректность работы программы при значениях M1 и М2 равных 0

Исходные данные: M1 = 0 M2 = 0 VECT1 = 1 2 3 4 5 6 7 8

VECT2 = 1 2 3 4 5 6 7 8 9 10

Ожидаемый результат:

Размер NEWVEC = 5 + 4 = 9

Нечетные индексы в VECT2: 1, 3, 5, 7, 9

VECT2[1] = 2

VECT2[3] = 4

VECT2[5] = 6

VECT2[7] = 8

VECT2[9] = 10

Четные индексы VECT1 в обратном порядке: 6, 4, 2, 0

-VECT1[6] = -7

-VECT1[4] = -5

-VECT1[2] = -3

-VECT1[0] = -1

NEWVEC = [2, 4, 6, 8, 10, -7, -5, -3, -1]

sumOfNewVec: 2 + 4 + 6 + 8 + 10 + (-7) + (-5) + (-3) + (-1) = 14

Сумма первых M1 элементов массива NEWVEC: 0

Cумма последних M2 элементов: 0

Поиск максимального отрицательного числа на промежутке индексов от M1(0) до M2(0): Нет

Полученный результат:

Изображение выглядит как текст, снимок экрана, Шрифт, алгебра

Автоматически созданное описание

Вывод: полученный результат совпал с ожидаемым.

# Вывод по работе

Разработка программы завершена на том основании, что:

1. Полученные результаты совпали с ожидаемыми;
2. Считаем набор тестов полными.