|  |  |
| --- | --- |
|  | МИНОБРНАУКИ РОССИИ    федеральное государственное бюджетное образовательное учреждение  высшего образования  **«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова»**  **(БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова»)** |
| БГТУ.СМК-Ф-4.2-К5-01 |

Факультет И Информационные и управляющие системы

шифр наименование

Кафедра И5 Информационные системы и программные технологии

шифр наименование

Дисциплина Компьютерная практика

ЛАБОРАТОРНАЯ РАБОТА №1

на тему

Особенности использования набора компиляторов

и утилит GCC

Выполнил студент группы И582

Дубровский В. И.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Фамилия И.О. | | | | |
|  |  |  |  |  | **РУКОВОДИТЕЛЬ** | | |
| Вальштейн |  |  | К.В. |  |  |  |  |
| Фамилия И.О. | |  | |  | Подпись | |  |
| Оценка | |  | |  |  |  |  |
| «\_\_\_\_\_» |  |  |  |  |  | 20\_\_\_\_ г. | |

САНКТ-ПЕТЕРБУРГ

2019 г.

**Цель работы.**  
 Изучить особенности работы с набором компиляторов и утилит GNU Compiler Collection (GCC).

Индивидуальное задание:

1. Для натурального числа N получить все его натуральные делители.
2. Вычислить сумму элементов массива М (15), значения которых лежат в введенном с клавиатуры диапазоне [X, Y].
3. Удалить из массива А (20) первый отрицательный элемент.

Задания к лабораторной работе.  
Часть I  
1. Написать программу 1 в соответствии с вариантом при помощи любого текстового  
редактора.  
2. Провести поэтапную компиляцию исходного текста написанной программы,  
разобраться в результатах, полученных на каждом этапе компиляции.  
3. Провести оптимизацию кода написанной программы с помощью набора  
компиляторов GCC.  
Часть II  
4. Написать программы 2 и 3 в соответствии с вариантом при помощи любого  
текстового редактора. Функции для работы с массивом вынести в отдельные  
файлы: в одном файле описать функции для ввода/вывода массива, в другом - для  
обработки массива. В обеих программах должны использоваться одни и те же  
функции для ввода/вывода массивов, описанные в одном из этих файлах.  
5. Провести раздельную компиляцию написанных файлов.  
6. Скомпилировать обе программы, используя созданные объектные файлы, обе  
программы должны использовать один и тот же объектный файл с функциями для  
ввода/вывода массива. 7. Создать статическую библиотеку для ввода/вывода массива и продемонстрировать возможности по её подключению.

Часть I

* 1. Напишем программу согласно индивидуальному заданию в блокноте

Программа:

#include <iostream>

int main()

{

int x;

std::cout << "Input natural nubmer: ";

std::cin >> x;

while(std::cin.fail() || x < 0)

{

std::cout << "Wrong input!" << std::endl;

std::cin.clear();

std::cin.ignore(255, '\n');

std::cout << "Input natural nubmer: ";

std::cin >> x;

}

int temp = x / 2;

std::cout << "List of natural dividers:" << std::endl;

for(int i=1; i<temp;i++)

{

if(temp%i == 0)

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

}

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

return 0;

}

2.1 Создадим исполняемый файл файл .exe

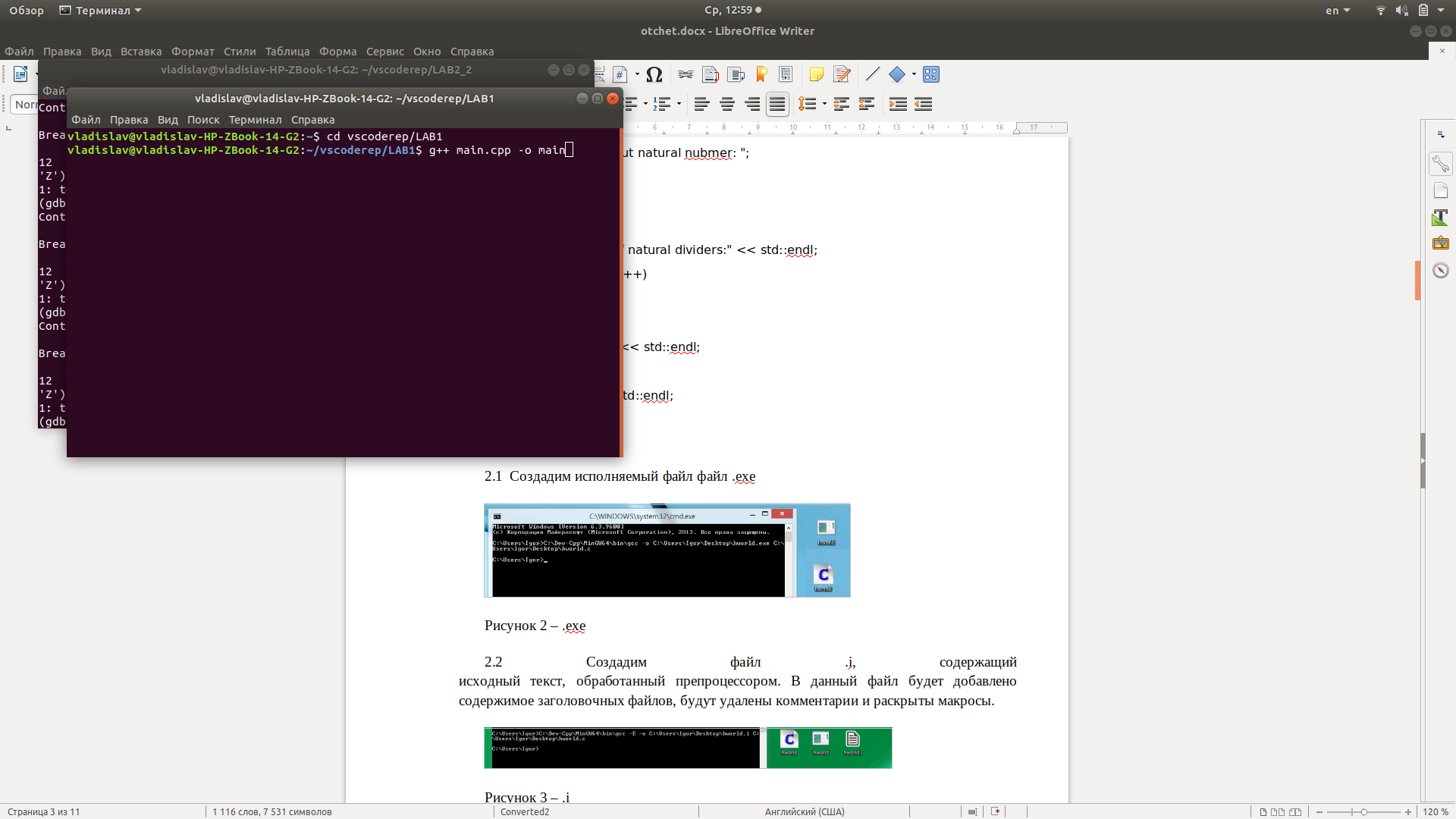


Рисунок 2 – .exe

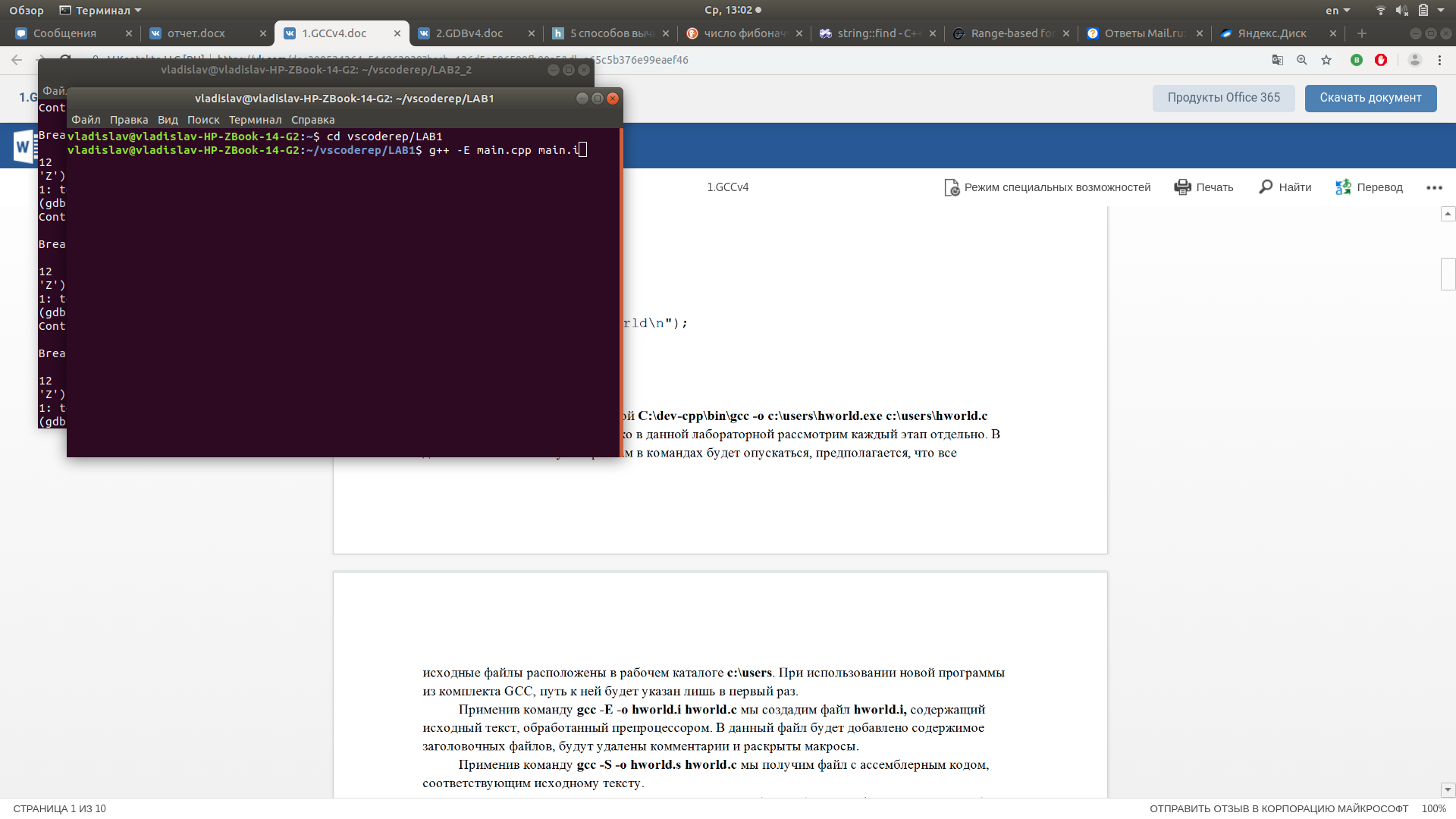
2.2 Создадим файл .i, содержащий  
исходный текст, обработанный препроцессором. В данный файл будет добавлено содержимое заголовочных файлов, будут удалены комментарии и раскрыты макросы.

Рисунок 3 – .i

2.3 Создадим файл .s с ассемблерным кодом, соответствующим исходному тексту.

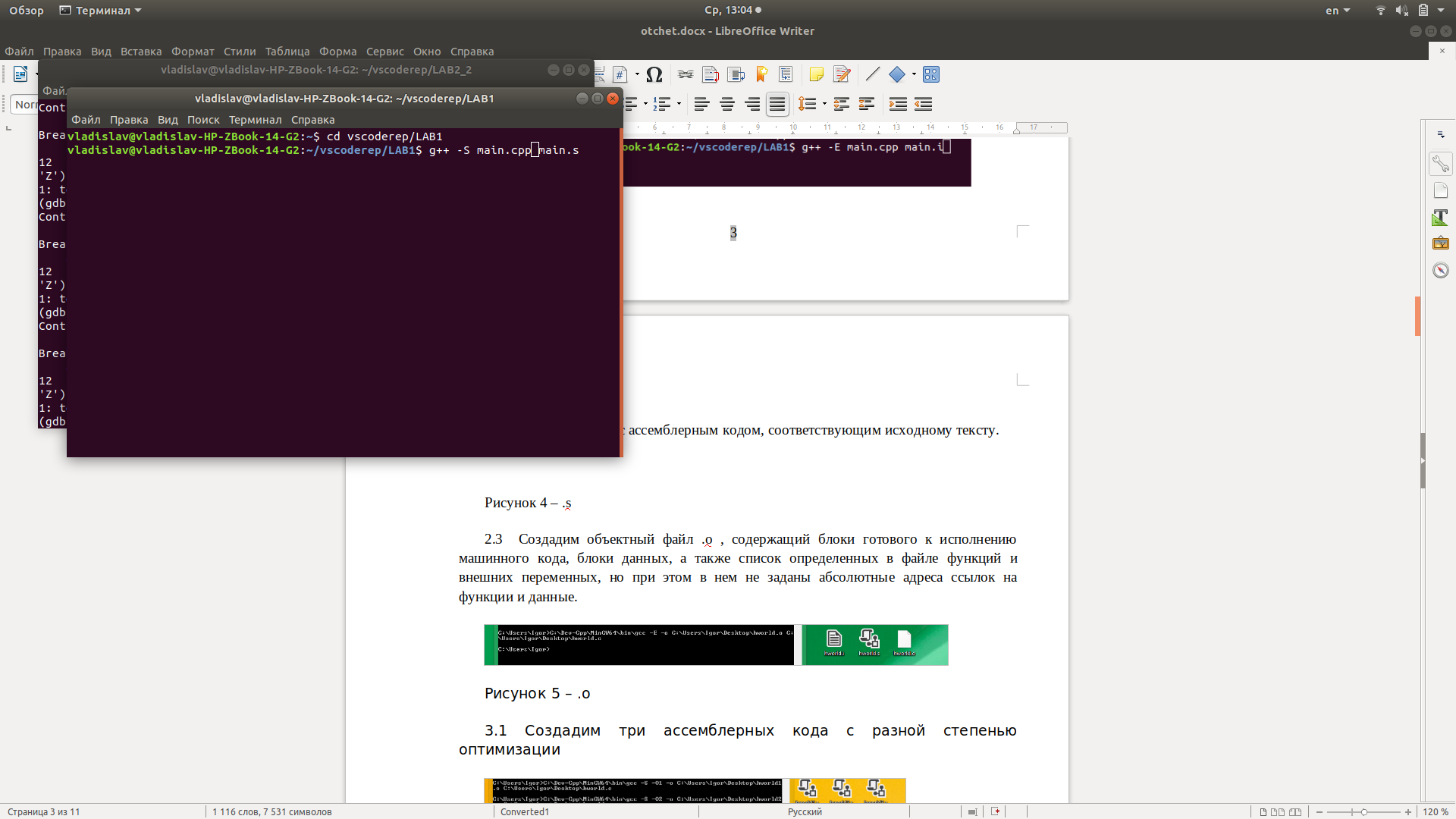


Рисунок 4 – .s

2.3 Создадим объектный файл .o , содержащий блоки готового к исполнению машинного кода, блоки данных, а также список определенных в файле функций и внешних переменных, но при этом в нем не заданы абсолютные адреса ссылок на функции и данные.

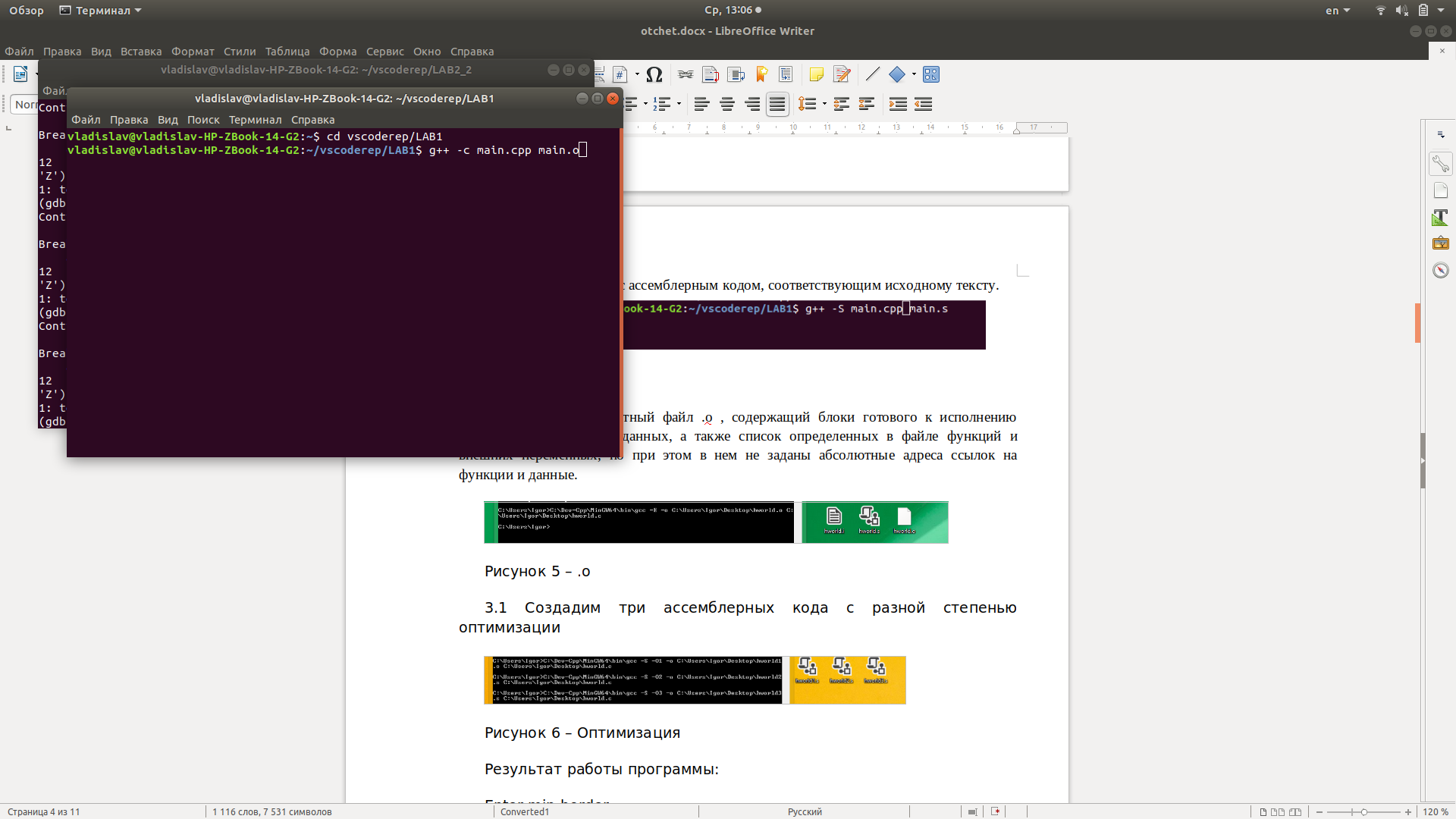


Рисунок 5 – .o

3.1 Создадим три ассемблерных кода с разной степенью оптимизации

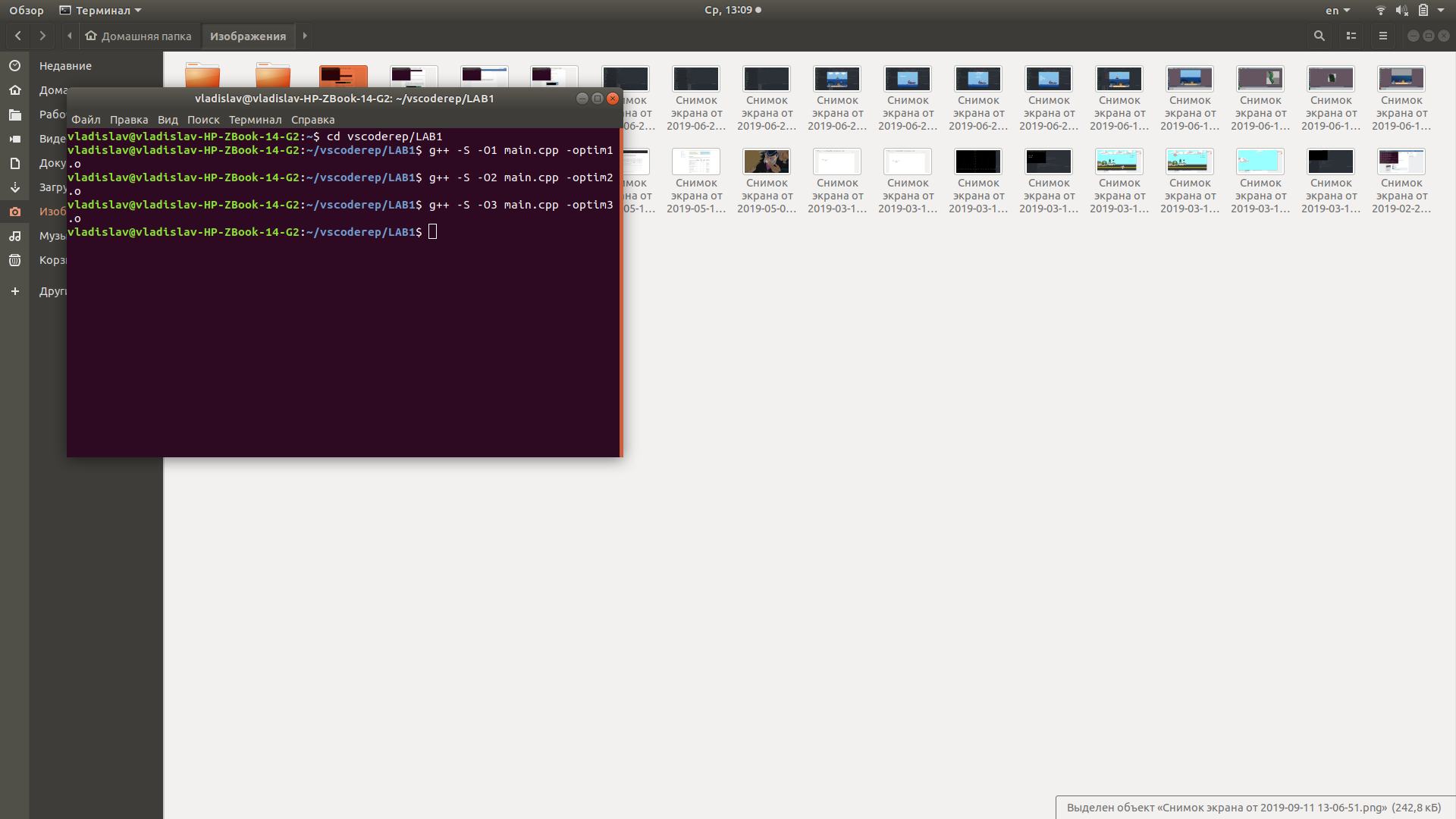
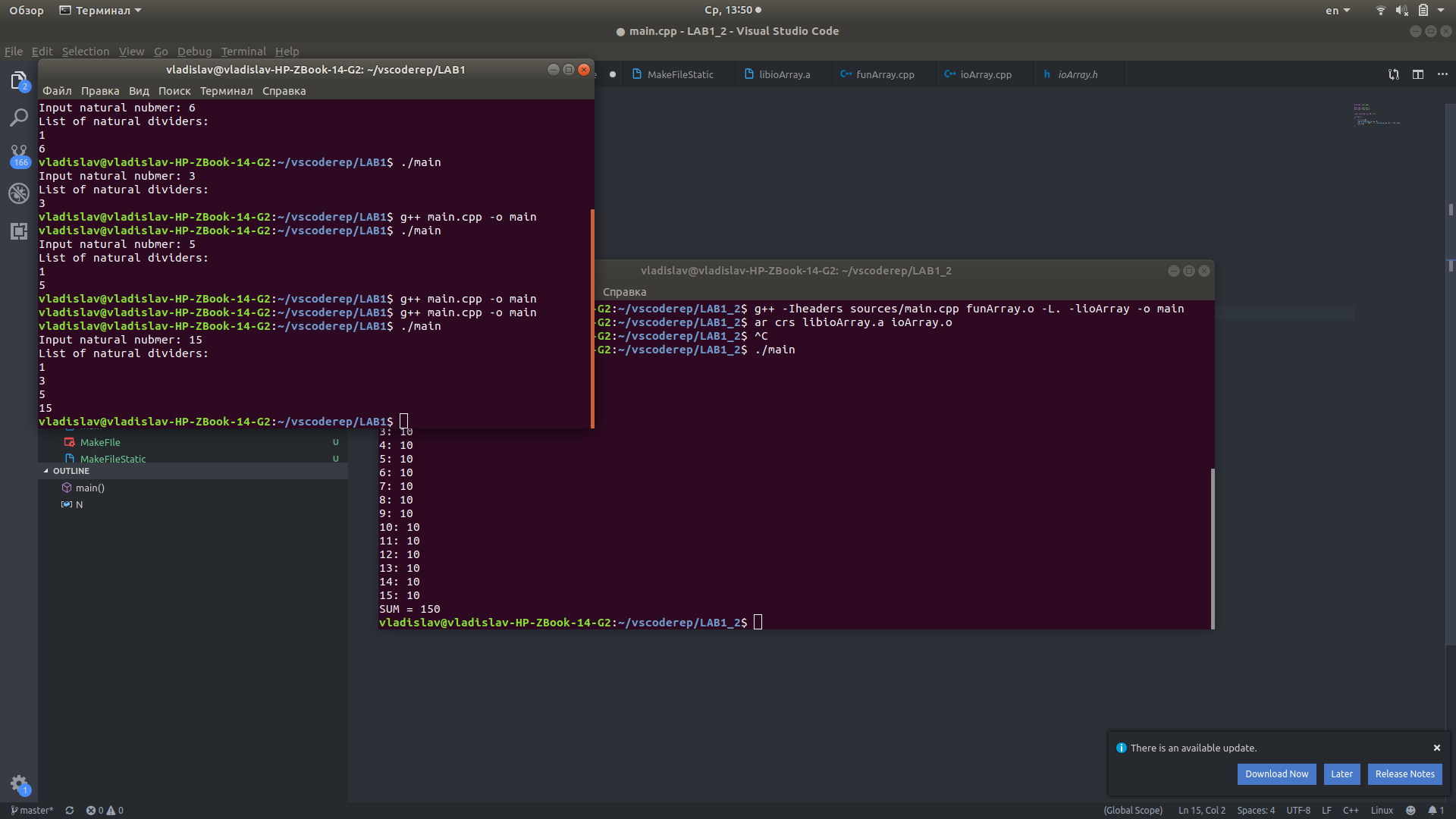


Рисунок 6 – Оптимизация

Результат работы программы:



Часть II

4.1 Первым делом пишем исходный код программы в блокноте, сохраняя файлы с разрешениями .сpp или .h

Программы:

main.cpp:

#include <iostream>

#include "ioArray.h"

#include "funArray.h"

const constexpr auto N = 15;

int main()

{

int array[N];

inputArrayBorders(array, N);

std::cout << "SUM = " << countSum(array, N) << std::endl;

return 0;

}

В программе main.cpp можно изменить функцию обработки массива для проверки корректности работы, что я и сделаю позже.

ioArray.h:

#pragma once

void inputArray(int \*arr, int N);

void inputArrayBorders(int \*arr, int N);

void outputArray(int \*arr, int N);

funArray.h:

#pragma once

int countSum(int\* arr, int N);

ioArray.cpp:

#include "ioArray.h"

#include <iostream>

void inputArray(int \*arr, int N, int X, int Y)

{

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

{

std::cout << i+1 << ": ";

std::cin >> arr[i];

}

}

void inputArrayBorders(int \*arr, int N)

{

int X=0, Y=10;

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

std::cout << "X = "; std::cin >> X;

std::cout << "Y = "; std::cin >> Y;

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

{

std::cout << i+1 << ": ";

int temp;

std::cin >> temp;

while(std::cin.fail() || temp < X || temp > Y)

{

std::cin.clear();

std::cin.ignore(255, '\n');

std::cin >> temp;

}

arr[i] = temp;

}

}

void outputArray(int \*arr, int N)

{

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

{

std::cout << "Array[" << i << "] = " << arr[i] << std::endl;

}

}

funArray.cpp:

#include "funArray.h"

#include <malloc.h>

int countSum(int\* arr, int N)

{

int sum = 0;

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

{

sum += arr[i];

}

return sum;

}

int deleteFirstNegative(int\* arr, int& N)

{

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

{

if(arr[i] < 0)

{

int re = arr[i];

for(int j=N-1; j > i; j--)

arr[j-1] = arr[j];

N--;

arr = (int\*)realloc(arr,sizeof(int)\*(N));

if(arr == nullptr) return -1;

return re;

}

}

return 0;

}

5. Теперь сформируем объектные файлы всех созданных файлов с расширением .cpp и создадим .exe для программы, состоящей из созданных объектных файлов:

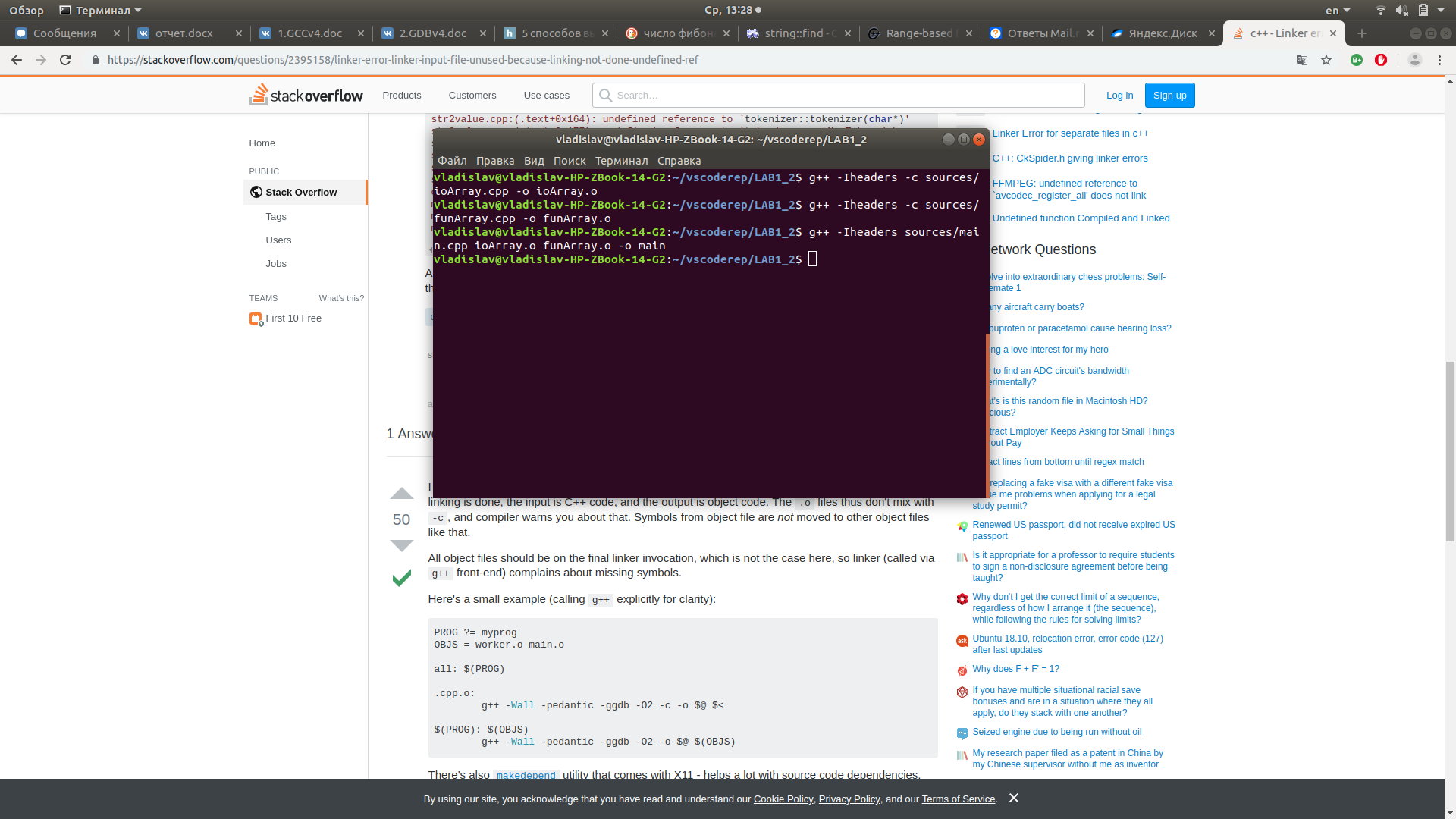


Рисунок 7 – Формирование объектных файлов и исполняемого

6.1 Cформируем объектный файл библиотеки:

ar crs libioArray.a ioArray.o

Рисунок 7 – библиотека из двух файлов с функциями

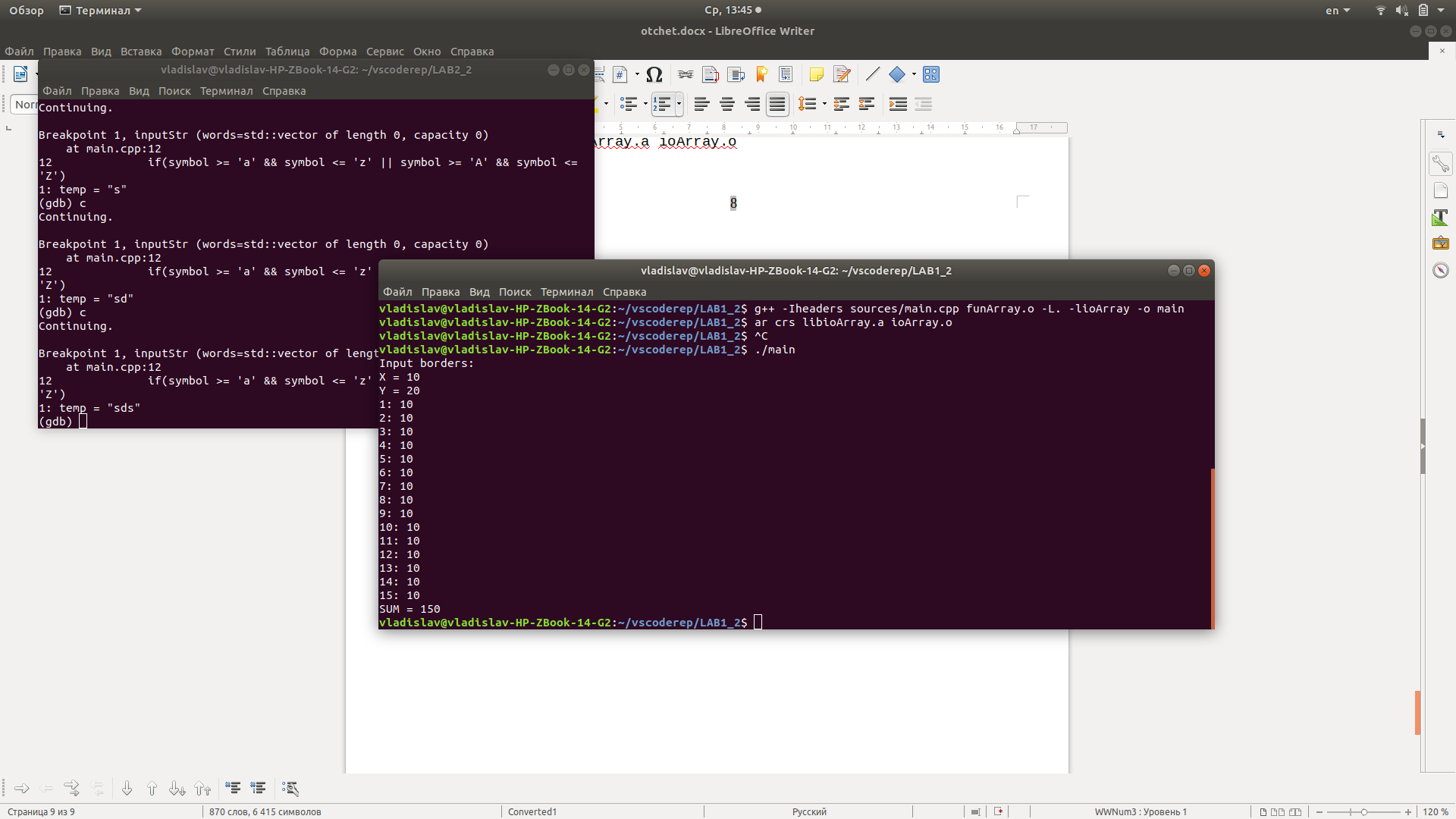
6.2 Теперь создадим исполняемый файл уже с использованием созданной статической библиотеки.

g++ -Iheaders sources/main.cpp funArray.o -L. -lioArray -o main

Рисунок 8 – создан исполняемый файл при помощи библиотеки

Теперь проверим корректность работы программы:

Результат работы программы:



8.1 Задание 3. Проверим корректность работы программы используя статическую библиотеку.

main.cpp

#include <iostream>

#include "ioArray.h"

#include "funArray.h"

auto N = 15;

int main()

{

int\* array = new int(N);

inputArray(array, N);

int result = deleteFirstNegative(array, N);

std::cout << (result == 0 ? "Not found" : "First negative: " + std::to\_string(result)) << std::endl;

outputArray(array, N);

delete array;

return 0;

}

Результат работы программы:

