МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ государственное БЮДЖЕТНОЕ

образовательное учреждение

высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра защиты информации



**ЛАБОРАТОРНАЯ РАБОТА №4**

**по дисциплине: «Программирование»**

**на тему: «Последовательные одномерные контейнеры»**

|  |  |
| --- | --- |
| Выполнил:  Студент гр. «АБ-420», «АВТФ»,  *Обметко Дмитрий Александрович*  «\_\_\_» \_\_\_\_\_\_ 2025г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | Проверил:  *Ассистент кафедры ЗИ*  *Исаев Г. А.*  «\_\_\_» \_\_\_\_\_\_ 2025г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

Новосибирск 2025

**Цели и задачи работы:** изучение алгоритмов формирования и обработки одномерных массивов и последовательных контейнеров, программирование и отладка программ формирования и обработки массивов.

**Задание к работе:** Написать программу решения задачи в соответствии с индивидуальным вариантом.

**Задание 1**

Отделите корни заданного уравнения, согласно варианту из табл.1, и уточните их одним из методов с точностью до ε=10-4. Решить уравнения методом половинного деления, методом Ньютона и методом простых итераций. Либо другими методами, перечисленными в теоретической части к заданию 1.

Вариант 6:

**Задание 2**

Вариант 2

1. Напишите программу, в которой определен массив 𝑎𝑟𝑟arr1 из n𝑛 чисел (15 ≤ n𝑛 ≤ 30) и инициализирован целыми случайными числами из диапазона [10, 30]. Определите массив 𝑎𝑟𝑟arr2 из 5 чисел и инициализируйте массив 𝑎𝑟𝑟arr2 целыми случайными числами из диапазона [10, 30].

2. Выведите количество чётных элементов массива arr𝑎𝑟𝑟1 на нечётных местах, а также сами элементы.

3. Определите, какие числа из массива 𝑎𝑟𝑟arr2 встречаются в массиве 𝑎𝑟𝑟arr1. Выведите эти числа и сколько раз они встречаются в массиве 𝑎𝑟𝑟arr1. Если ни одно число не встречается, то выведите соответствующее сообщение.

4. Напишите программу, модифицирующую массив случайных символов размера, как в примере:

[v,!,N,R,(,3] → [!,N,R,(,3,v].

5. Определите и инициализируйте массив, состоящий из случайных целых чисел, входящих в диапазон [1000, 9000]. Вычислите сумму элементов массива. Напишите программу, меняющую местами цифры в элементах массива в порядке возрастания (например, 8315 → 1358). Отсортируйте массив в порядке возрастания. Вычислите сумму элементов массива и сравните с суммой элементов исходного массива.

**Задание 3**

Генерация псевдослучайных чисел. Задание выполняется на языке C++. Для генерации чисел использовать алгоритм: Xorshift

**Задание 4**

На выборах участвуют n кандидатов и k избирателей. Каждый избиратель формирует полное ранжирование (цепочку) кандидатов от самого предпочтительного до наименее предпочтительного. Например, если кандидаты – Петя, Вася и Оля, то голос вида «Вася – Оля – Петя» означает, что для данного избирателя кандидат Вася имеет наивысший приоритет, а Петя – наименьший. Необходимо определить победителя выборов, используя два алгоритма: метод Борда и метод Кондорсе. Программа должна вывести победителя согласно каждому методу.

Если результаты разнятся – можно добавить комментарий о том, что разные алгоритмы могут давать разные результаты в зависимости от специфики голосования.

Протестируйте программу на различных наборах данных, в том числе на крайних случаях (например, когда все избиратели отдают одинаковые предпочтения или когда не существует кандидата, удовлетворяющего условию Кондорсе). В заключительной части лабораторной работы опишите, в каких случаях методы могут давать разные результаты и какие недостатки имеет каждый метод.

**Задание 5**

Требуется реализовать игру «Предать или сотрудничать» и реализовать 3 алгоритма поведения в игре.

Игра состоит из случайного кол-ва раундов от 100 до 200 (итоговое кол-во раундов при каждом запуске игры генерируется случайно). На протяжении игровой сессии сражаются 2 алгоритма. В каждом раунде каждый алгоритм выбирает, либо сотрудничество, либо предательство. Если алгоритм А выбирает предательство и алгоритм Б выбирает предательство они получают по 4 очка. Если алгоритм А выбирает сотрудничество, а алгоритм Б выбирает предательство - алгоритм А получает 0 очков, а алгоритм Б получает 20 очков. Если оба алгоритма выбирают сотрудничество оба получают 24 очка. Каждому алгоритму в каждом раунде известны результаты всех предыдуще раундов текущей игровой сессии, на основе этих данных алгоритм может выбирать будет он сотрудничать или предаст.

Каждый алгоритм должен представлять из себя функцию с сигнатурой: boolean func(int32 round\_number, array[boolean] self\_choices, array[boolean] enemy\_choices)

round\_number – номер текущего раунда

self\_choices – массив булевых значений, содержит информацию о собственных выборов (предать или сотрудничать) за все предыдущие раунды

enemy\_choices - массив булевых значений, содержит информацию о

выборов (предать или сотрудничать) противника за все предыдущие раунды

true – сотруднничество

false – предательство

**Выполнение задания 1**

Листинг программы на С++ (рисунок 1)

#include <iostream>

#include <vector>

#include <cmath>

#include <iomanip>

#define EPSILON 0.0001

using namespace std;

float function(float x)

{

return (x \* x - 3 \* sin(x));

}

float derivativeFunction(float x)

{

return (2 \* x - 3 \* cos(x));

}

float secondDerivativeFunction(float x)

{

return (2 + 3 \* sin(x));

}

void methodHalf(float a, float b)

{

// if (function(a) \* function(b) > 0) {

// cout << "На концах отрезка функция имеет одинаковые знаки" << endl;

// return;

// }

int step = 0;

float center;

cout << fixed << setprecision(5);

cout << left << setw(3) << "N" << setw(9) << "a" << setw(8) << "b" << "b - a" << endl;

cout << left << setw(3) << step << setw(9) << a << setw(8) << b << abs(b - a) << endl;

while (abs(b-a) >= EPSILON) {

center = (a+b)/2;

if (function(a) \* function(center) < 0) {

b = center;

}

if (function(center) \* function(b) < 0) {

a = center;

}

step++;

cout << left << setw(3) << step << setw(9) << a << setw(8) << b << abs(b - a) << endl;

}

cout << "Корень = " << a << endl << "Шагов всего: " << step << endl;

}

void methodNewton(float a) {

// if (function(a) \* secondDerivativeFunction(a) <= 0) {

// cout << "У функции и второй произвнодной разные знаки" << endl;

// return;

// }

int k = 0;

float x\_k1 = 0;

float x\_k = a;

cout << left << setw(3) << "N" << setw(9) << "x\_n" << setw(9) << "x\_n+1" << "x\_n1 - x\_n" << endl;

cout << left << setw(3) << k << setw(9) << x\_k << setw(9) << x\_k1 << x\_k1 - x\_k << endl;

while (true) {

k++;

x\_k1 = x\_k - function(x\_k)/derivativeFunction(x\_k);

cout << left << setw(3) << k << setw(9) << x\_k << setw(9) << x\_k1 << x\_k1 - x\_k << endl;

if (abs(x\_k1 - x\_k) < EPSILON) {

break;

}

x\_k = x\_k1;

}

cout << "Корень = " << x\_k1 << endl << "Шагов всего: " << k << endl;

}

float canonFunction(float x) {

return (sqrt(abs(3\*sin(x))));

}

void methodSimpleIterations(float a) {

int k = 0;

float x\_k1 = 0;

float x\_k = a;

cout << left << setw(3) << "N" << setw(9) << "x\_n" << setw(9) << "x\_n+1" << "x\_n1 - x\_n" << endl;

cout << left << setw(3) << k << setw(9) << x\_k << setw(9) << x\_k1 << x\_k1 - x\_k << endl;

while (true) {

k++;

x\_k1 = canonFunction(x\_k);

cout << left << setw(3) << k << setw(9) << x\_k << setw(9) << x\_k1 << x\_k1 - x\_k << endl;

if (abs(x\_k1 - x\_k) <= EPSILON) {

break;

}

x\_k = x\_k1;

}

cout << "Корень = " << x\_k1 << endl << "Шагов всего: " << k << endl;

}

int main()

{

cout << "Метод половинного деления:" << endl;

methodHalf(-2,1);

cout << endl;

methodHalf(1, 2);

cout << endl;

cout << "Метод Ньютона:" << endl;

methodNewton(-0.5);

cout << endl;

methodNewton(1);

cout << endl;

cout << "Метод простых итераций:" << endl;

methodSimpleIterations(-0.5);

cout << endl;

methodSimpleIterations(1);

cout << endl;

return 0;

}

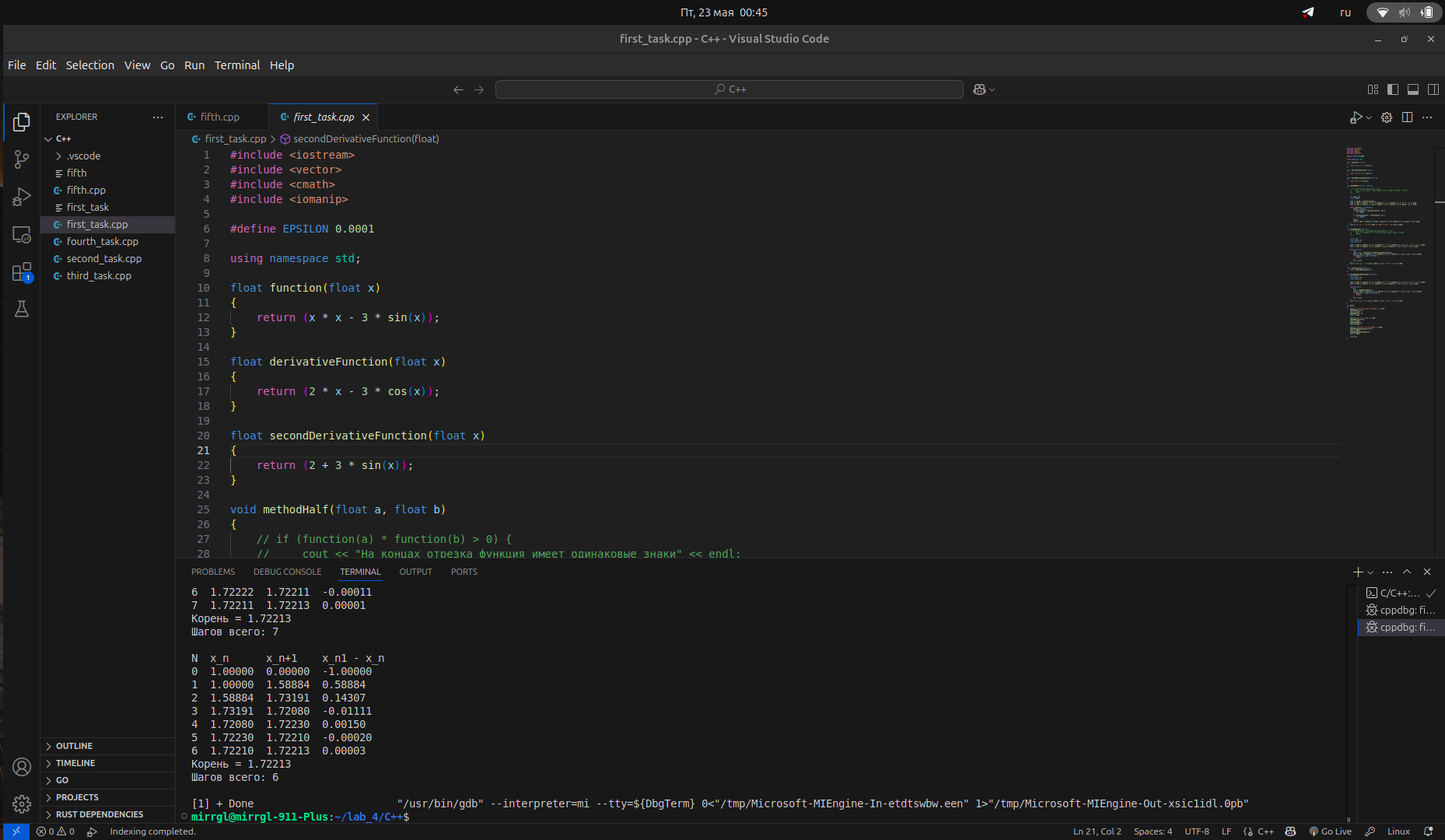


Рисунок 1 — Демнострация кода программы на C++

Листинг программы на Python (рисунок 2)

import math

EPSILON = 0.0001

def function(x):

return x\*\*2 - 3 \* math.sin(x)

def derivativeFunction(x):

return 2 \* x - 3 \* math.cos(x)

def secondDerivativeFunction(x):

return 2 + 3 \* math.sin(x)

def methodHalf(a, b):

step = 0

center = 0.0

print(f"{'N':<3}{'a':<9}{'b':<8}b - a")

print(f"{step:<3}{a:<9.5f}{b:<8.5f}{abs(b - a):.5f}")

while abs(b - a) >= EPSILON:

center = (a + b) / 2

if function(a) \* function(center) < 0:

b = center

if function(center) \* function(b) < 0:

a = center

step += 1

print(f"{step:<3}{a:<9.5f}{b:<8.5f}{abs(b - a):.5f}")

print(f"Корень = {a:.5f}\nШагов всего: {step}\n")

def methodNewton(a):

k = 0

x\_k1 = 0.0

x\_k = a

print(f"{'N':<3}{'x\_n':<9}{'x\_n+1':<9}x\_n1 - x\_n")

print(f"{k:<3}{x\_k:<9.5f}{x\_k1:<9.5f}{x\_k1 - x\_k:.5f}")

while True:

k += 1

x\_k1 = x\_k - function(x\_k) / derivativeFunction(x\_k)

print(f"{k:<3}{x\_k:<9.5f}{x\_k1:<9.5f}{x\_k1 - x\_k:.5f}")

if abs(x\_k1 - x\_k) < EPSILON:

break

x\_k = x\_k1

print(f"Корень = {x\_k1:.5f}\nШагов всего: {k}\n")

def canonFunction(x):

return math.sqrt(abs(3 \* math.sin(x)))

def methodSimpleIterations(a):

k = 0

x\_k1 = 0.0

x\_k = a

print(f"{'N':<3}{'x\_n':<9}{'x\_n+1':<9}x\_n1 - x\_n")

print(f"{k:<3}{x\_k:<9.5f}{x\_k1:<9.5f}{x\_k1 - x\_k:.5f}")

while True:

k += 1

x\_k1 = canonFunction(x\_k)

print(f"{k:<3}{x\_k:<9.5f}{x\_k1:<9.5f}{x\_k1 - x\_k:.5f}")

if abs(x\_k1 - x\_k) <= EPSILON:

break

x\_k = x\_k1

print(f"Корень = {x\_k1:.5f}\nШагов всего: {k}\n")

print("Метод половинного деления:")

methodHalf(-2, 1)

methodHalf(1, 2)

print("\nМетод Ньютона:")

methodNewton(-0.5)

methodNewton(1)

print("\nМетод простых итераций:")

methodSimpleIterations(-0.5)

methodSimpleIterations(1)

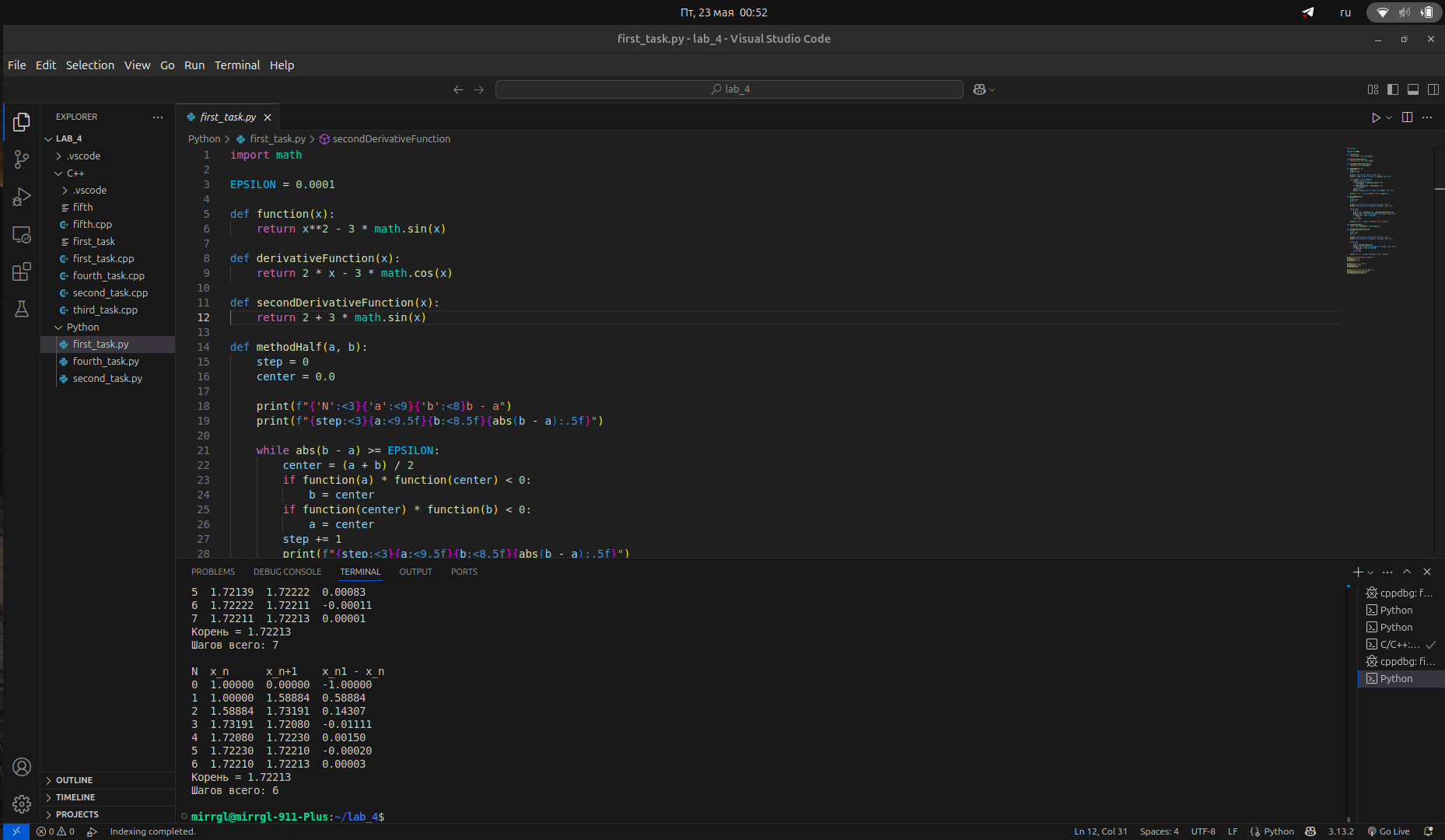


Рисунок 2 — Демонстрация кода на Python

Вывод по скорости сходимости: Метод Ньютона находит корни за 4 и 6 итераций из-за квадратичной сходимости. Метод простых итераций находит корни за 6 и 7 итераций. Метод половинного деления из-за линейной сходимости является самым долгим: 14 и 15 итераций для нахождения корней.

**Выполнение задания 2**

Листинг программы на С++ (рисунок 3)

#include <vector>

#include <iostream>

#include <random>

#include <tuple>

#include <string>

#include <ctime>

#include <sstream>

#include <algorithm>

using namespace std;

template<typename T>

ostream& operator<<(ostream& os, vector<T> vec) {

for (int i = 0; i < vec.size(); i++) {

os << vec[i] << " ";

}

return os;

}

template<typename T>

istringstream& operator>>(istringstream& is, vector<T>& vec) {

char i;

while (is >> i) {

vec.push\_back(i);

}

return is;

}

template<typename T>

void inputArray(vector<T>& vec) {

string input;

getline(cin, input);

istringstream iss (input);

iss >> vec;

}

void countEvenNumbers(vector<int> arr1) {

int count = 0;

for(int i = 0; i < arr1.size(); i += 2) {

if (arr1[i] % 2 == 0) {

cout << arr1[i] << " ";

count++;

}

}

cout << endl << "Всего " << count << endl;

}

void arr2IntoArr1(vector<int> arr1, vector<int> arr2) {

int count = 0;

bool isThereCommonNumbers = 0;

for (int numberFromArr2 : arr2) {

for (int numberFromArr1 : arr1) {

if (numberFromArr2 == numberFromArr1) {

count++;

isThereCommonNumbers = 1;

}

}

if (isThereCommonNumbers) {

cout << "Число " << numberFromArr2 << " повторяется " << count << endl;

count = 0;

}

}

if (!isThereCommonNumbers) {

cout << "В первом и втором массивах нет общих чисел" << endl;

}

}

vector<char> sortArrayOfSymbols(vector<char> array) {

vector<char> sortedArray;

vector<char> signArray;

vector<char> upperArray;

vector<char> bracketArray;

vector<char> digitArray;

vector<char> lowerArray;

for (auto element : array) {

if (int(element) == 33 || int(element) == 39 || int(element) == 46 || int(element) == 58 || int(element) == 59 || int(element) == 63) {

signArray.push\_back(element);

}

else if (int(element) >= 65 && int(element) <= 90) {

upperArray.push\_back(element);

}

else if (int(element) == 34 || int(element) == 39 || int(element) == 60 || int(element) == 62 || int(element) == 40 || int(element) == 41 || int(element) == 91 || int(element) == 93 || int(element) == 123 || int(element) == 125) {

bracketArray.push\_back(element);

}

else if (int(element) >= 48 && int(element) <= 57) {

digitArray.push\_back(element);

}

else if (int(element) >= 97 && int(element) <= 122) {

lowerArray.push\_back(element);

}

}

sortedArray.insert(sortedArray.end(), signArray.begin(), signArray.end());

sortedArray.insert(sortedArray.end(), upperArray.begin(), upperArray.end());

sortedArray.insert(sortedArray.end(), bracketArray.begin(), bracketArray.end());

sortedArray.insert(sortedArray.end(), digitArray.begin(), digitArray.end());

sortedArray.insert(sortedArray.end(), lowerArray.begin(), lowerArray.end());

return sortedArray;

}

vector<int> sortArrayOfNumbers(vector<int> array) {

for (auto& number : array) {

string str\_number = to\_string(number);

sort(str\_number.begin(), str\_number.end());

number = stoi(str\_number);

}

sort(array.begin(), array.end());

return array;

}

int main() {

knuth\_b engine;

engine.seed(time(nullptr));

uniform\_int\_distribution<> distribution(15, 30);

int length = distribution(engine);

vector<int> arr1;

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

uniform\_int\_distribution<> distribution(10, 30);

arr1.push\_back(distribution(engine));

}

cout << arr1 << endl;

vector<int> arr2;

for (int i = 0; i < 5; i++) {

uniform\_int\_distribution<> distribution(10, 30);

arr2.push\_back(distribution(engine));

}

cout << arr2 << endl;

countEvenNumbers(arr1);

arr2IntoArr1(arr1, arr2);

cout << endl;

vector<char> notsortedArray;

inputArray(notsortedArray);

cout << sortArrayOfSymbols(notsortedArray) << endl;

uniform\_int\_distribution<> distribution2(5, 10);

length = distribution2(engine);

vector<int> arr3;

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

uniform\_int\_distribution<> distribution2(1000, 9000);

arr3.push\_back(distribution2(engine));

}

cout << arr3 << endl;

cout << sortArrayOfNumbers(arr3) << endl;

return 0;

}

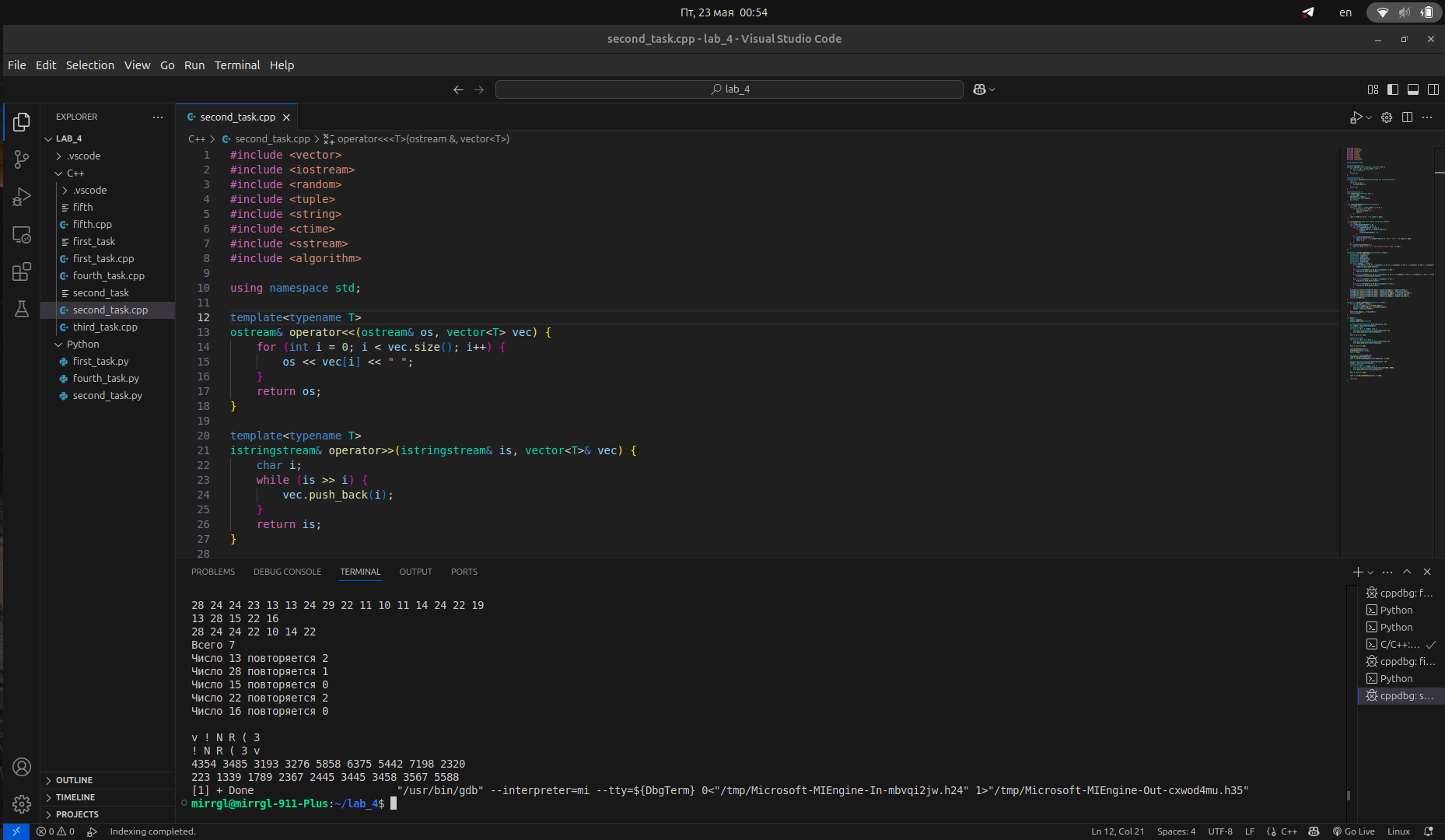


Рисунок 3 — Демонстрация кода на C++

Листинг программы на Python (рисунок 4)

import random

import time

def print\_vector(vec):

print(' '.join(map(str, vec)))

def count\_even\_numbers(arr1):

count = 0

for i in range(0, len(arr1), 2):

if arr1[i] % 2 == 0:

print(arr1[i], end=' ')

count += 1

print(f"\nВсего {count}")

def arr2\_into\_arr1(arr1, arr2):

count = 0

common\_exists = False

for num2 in arr2:

count = 0

for num1 in arr1:

if num2 == num1:

count += 1

common\_exists = True

if count > 0:

print(f"Число {num2} повторяется {count}")

if not common\_exists:

print("В первом и втором массивах нет общих чисел")

def sort\_symbols(array):

signs = []

upper = []

brackets = []

digits = []

lower = []

for char in array:

code = ord(char)

if code in {33, 34, 39, 46, 58, 59, 60, 62, 63}:

signs.append(char)

elif code in range(65, 91):

upper.append(char)

elif code in {34, 39, 40, 41, 60, 62, 91, 93, 123, 125}:

brackets.append(char)

elif code in range(48, 58):

digits.append(char)

elif code in range(97, 123):

lower.append(char)

return (signs + upper + brackets + digits + lower)

def sort\_numbers(array):

return sorted([int(''.join(sorted(str(i)))) for i in array])

random.seed(time.time())

arr1 = [random.randint(10, 30) for \_ in range(random.randint(15, 30))]

print\_vector(arr1)

arr2 = [random.randint(10, 30) for \_ in range(5)]

print\_vector(arr2)

count\_even\_numbers(arr1)

arr2\_into\_arr1(arr1, arr2)

print()

symbols = list(input().replace(' ', ''))

print(' '.join(sort\_symbols(symbols)))

arr3 = [random.randint(1000, 9000) for \_ in range(random.randint(5, 10))]

print\_vector(arr3)

print\_vector(sort\_numbers(arr3))

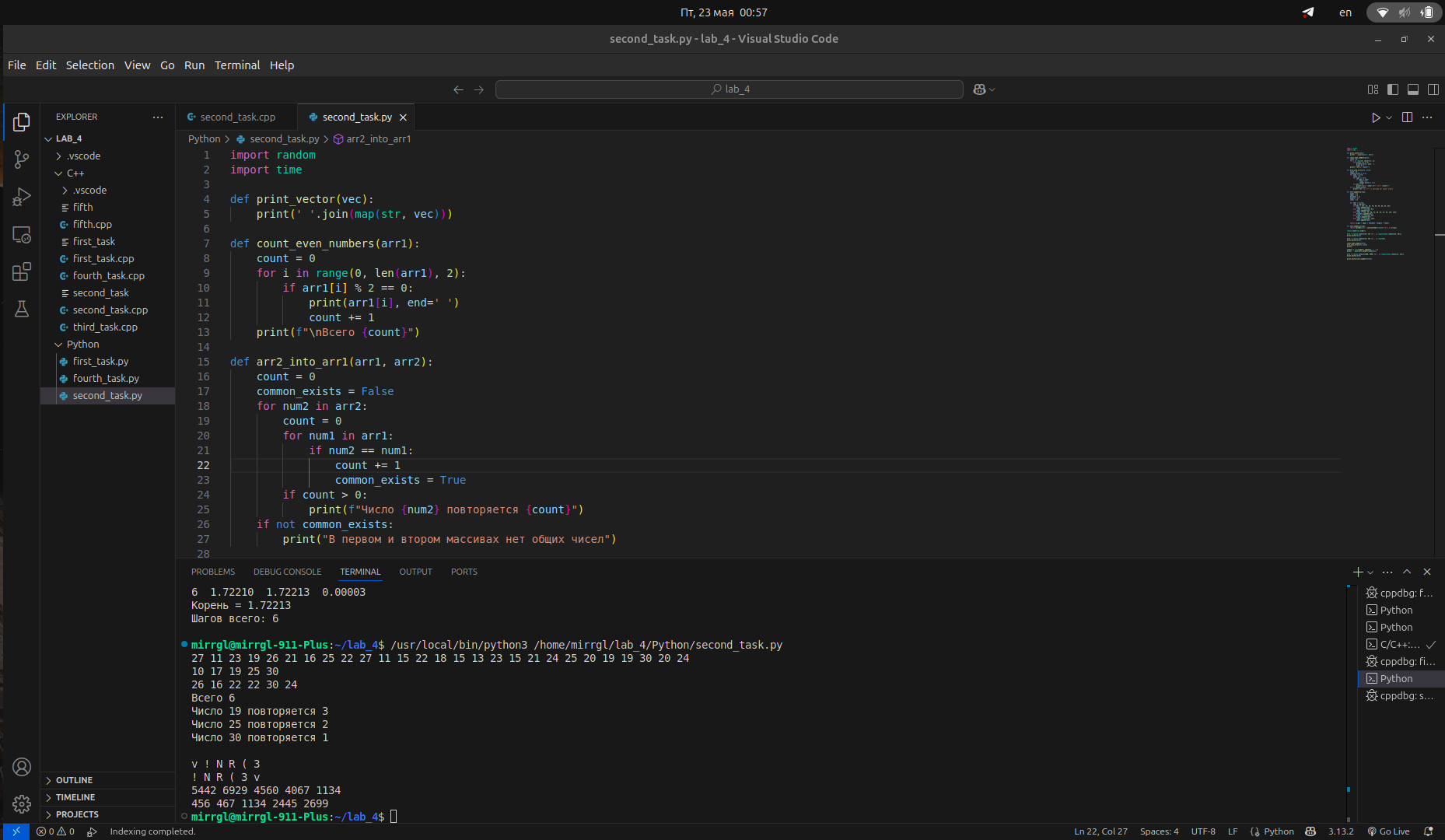


Рисунок 4 -—Демонстрация кода на Python

**Выполнение задания 3**

Листинг программы на С++ (рисунок 5)

#include <iostream>

#include <ctime>

using namespace std;

long XORShift(int step) {

long seed = time(nullptr) + step;

seed ^= (seed << 21);

seed ^= (seed >> 35);

seed ^= (seed << 4);

return seed;

}

int main() {

for (int i = 0; i < 10; i++) {

cout << XORShift(i) << " ";

}

return 0;

}

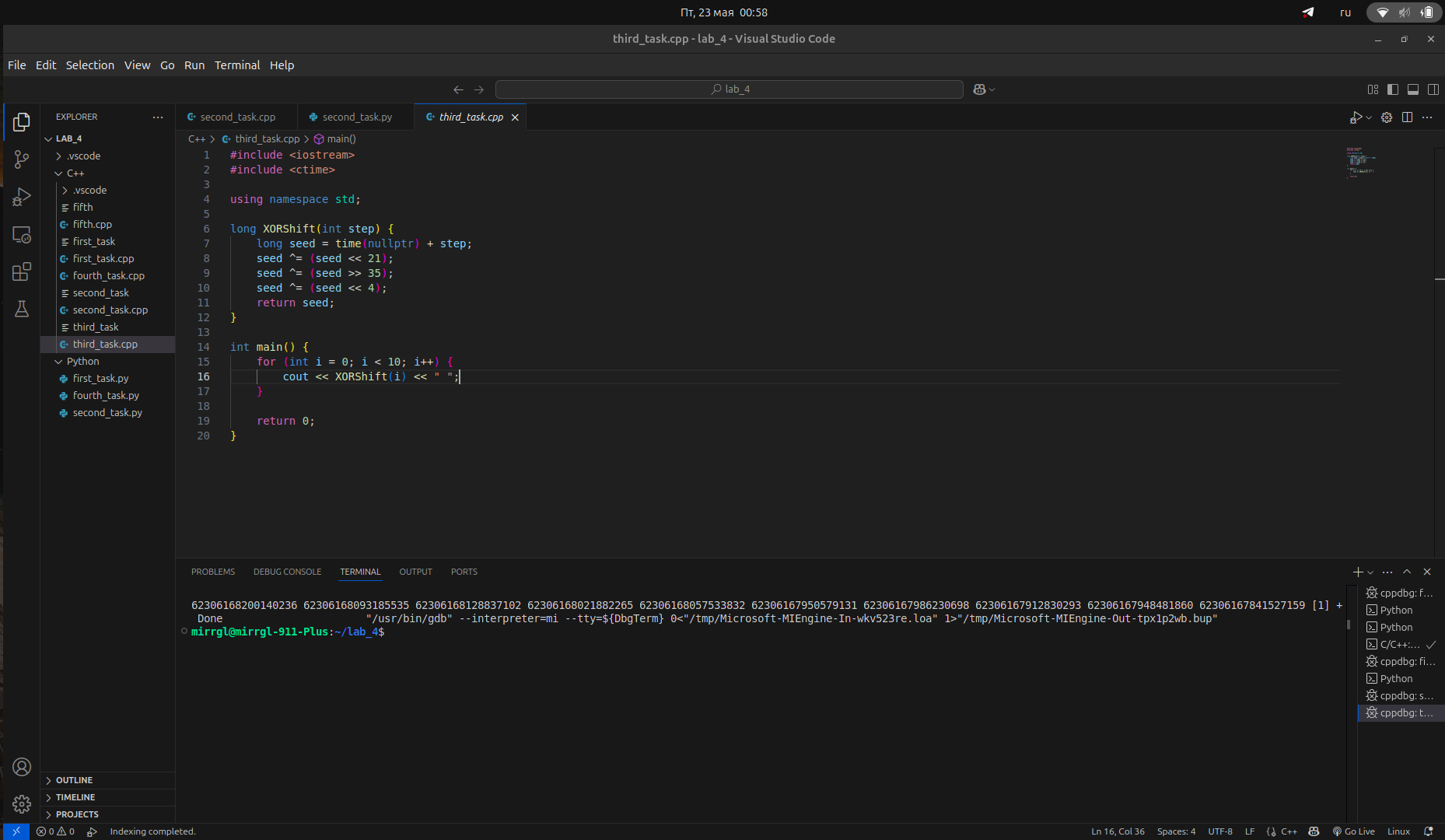


Рисунок 5 — Демонстрация кода на C++

**Выполнение задания 4**

Листинг программы на С++ (рисунок 6)

#include <iostream>

#include <vector>

#include <map>

using namespace std;

char methodBord(vector<vector<char>> chains) {

map<char,int> scores;

for(int i = 0; i < chains[0].size(); i++) {

scores[chains[0][i]] = 0;

}

for (auto& chain : chains) {

for (int i = 0; i < chain.size(); i++) {

scores[chain[i]] += chain.size() - 1 - i;

}

}

int max = 0;

char max\_key = ' ';

for (auto& [a, b] : scores) {

if (b > max) {

max\_key = a;

max = b;

}

}

return max\_key;

}

char methodCondorse(vector<vector<char>> chains) {

map<char,int> scores;

for(int i = 0; i < chains[0].size(); i++) {

scores[chains[0][i]] = 0;

}

for (auto& chain : chains) {

scores[chain[0]]++;

}

int max = 0;

char max\_key = ' ';

for (auto& [a, b] : scores) {

if (b > max) {

max\_key = a;

max = b;

}

}

return max\_key;

}

int main() {

int n, k;

cin >> n >> k;

vector<vector<char>> chains;

chains.resize(k);

for (int i = 0; i < k; i++) {

chains[i].resize(n);

}

for (int i = 0; i < k; i++) {

cout << "Цепь для " << i+1 << " избирателя:" << endl;

for (int j = 0; j < n; j++) {

char letter;

cin >> letter;

chains[i][j] = letter;

}

cout << endl;

}

cout << "По методу Борда выигрывает кандидат " << methodBord(chains) << endl;

cout << "По методу Кондорсе выигрывает кандидат " << methodCondorse(chains) << endl;

return 0;

}

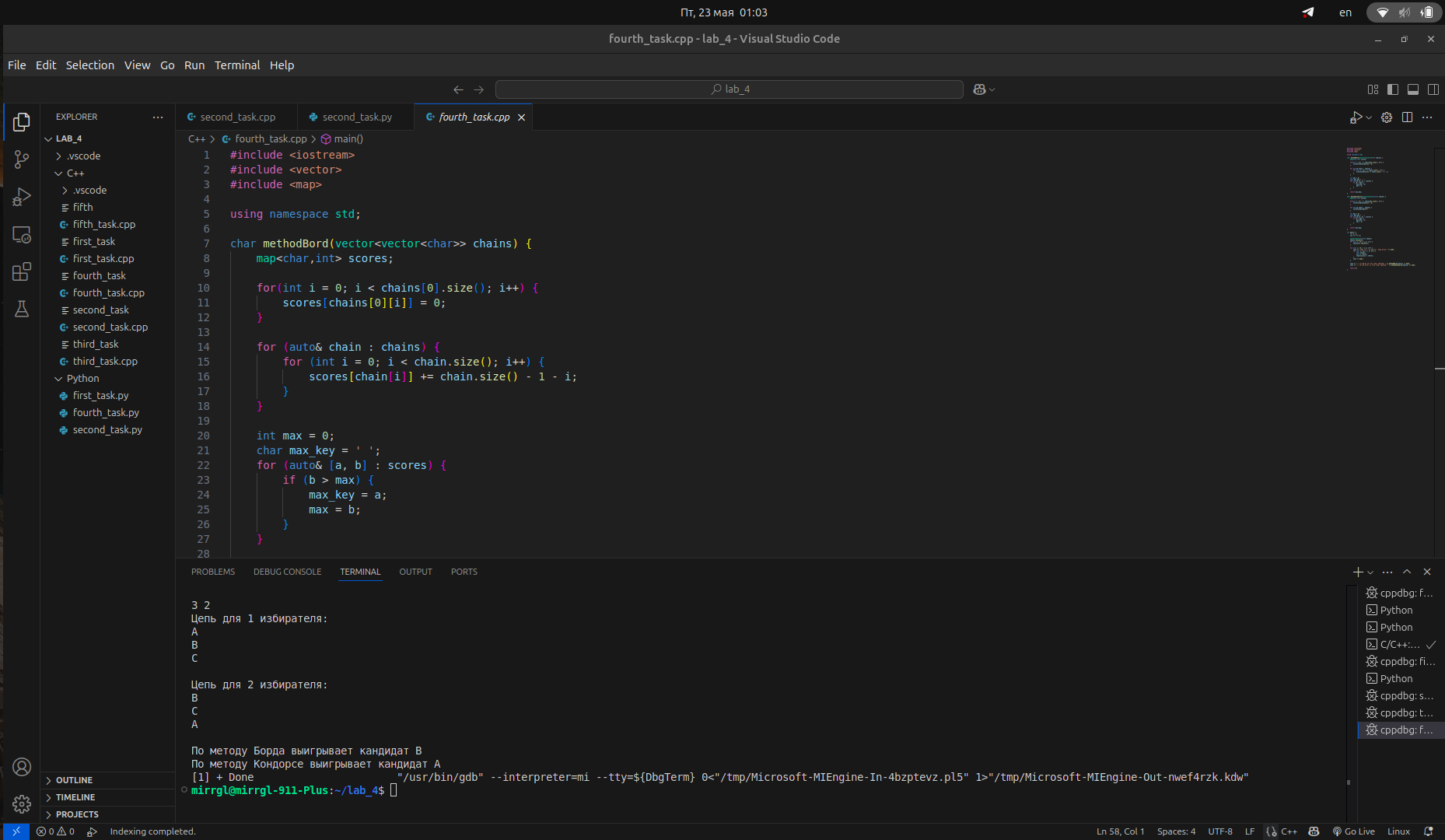


Рисунок 6 — Демонстрация кода на C++

Листинг программы на Python (рисунок 7)

def method\_bord(chains):

scores = {c: 0 for c in chains[0]}

for chain in chains:

for i in range(len(chain)):

scores[chain[i]] += len(chain) - 1 - i

max = -1

max\_key = ' '

for candidate, score in scores.items():

if score > max:

max = score

max\_key = candidate

return max\_key

def method\_condorse(chains):

scores = {c: 0 for c in chains[0]}

for chain in chains:

scores[chain[0]] += 1

max\_votes = -1

winner = ' '

for candidate, votes in scores.items():

if votes > max\_votes:

max\_votes = votes

winner = candidate

return winner

n, k = map(int, input().split())

chains = []

for i in range(k):

print(f"Цепь для {i+1} избирателя:")

chains.append(input().split())

print(f"По методу Борда выигрывает кандидат {method\_bord(chains)}")

print(f"По методу Кондорсе выигрывает кандидат {method\_condorse(chains)}")

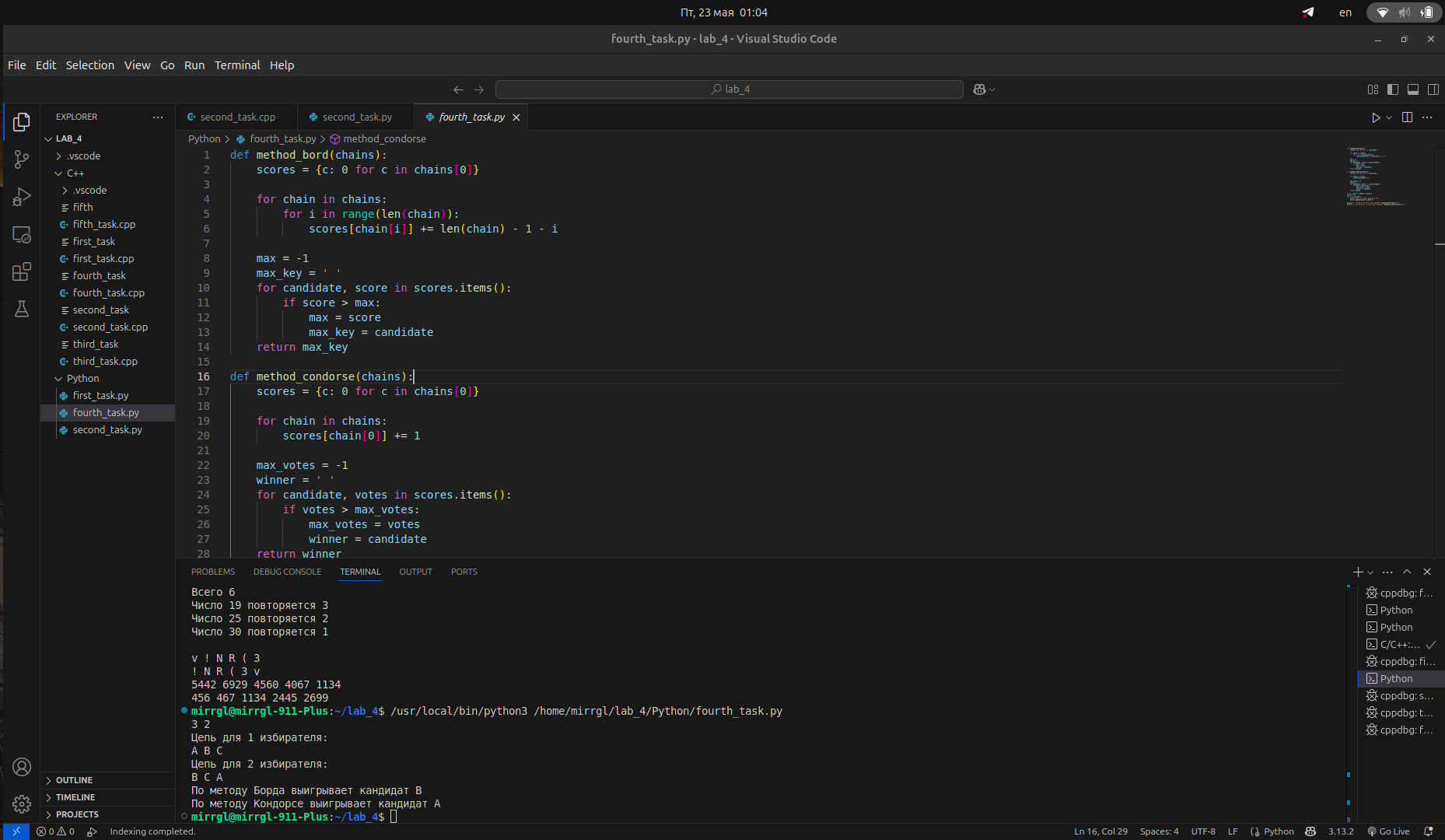


Рисунок 7 — Демонстрация кода на Python

Анализ методов: методы могут давать разные результаты, например, при случае, когда идёт сравнение A — B и B — A у двух избирателей. В таком случае по методу Борда выигрывает B, так как у него самая большая сумма баллов. А по методу Кондорсе нельзя точно установить победителя.

Недостатки метода Борда: чувствителен к добавлению «клона» кандидата; не всегда выбирает кандидата, который побеждает всех в парных сравнениях.

Недостатки метода Кондорсе: может не иметь победителя; не учитывает степень предпочтения, только факт победы в парном сравнении.

**Выполнение задания 5**

Листинг программы на С++ (рисунок 8)

#include <iostream>

#include <vector>

#include <random>

#include <ctime>

using namespace std;

bool alwaysCoop(int32\_t& round\_number, vector<bool>& self\_choices, vector<bool>& enemy\_choices) {

return 1;

}

bool mirror(int32\_t& round\_number, vector<bool>& self\_choices, vector<bool>& enemy\_choices) {

if(round\_number == 0) {return 1;}

return (enemy\_choices[round\_number-1]);

}

bool reverse(int32\_t& round\_number, vector<bool>& self\_choices, vector<bool>& enemy\_choices) {

if(round\_number == 0) {return 1;}

return ((self\_choices[round\_number-1]+1)%2);

}

void scoreCount(pair<int,int>& scores, bool A, bool B) {

if (A == 0 && B == 0) {

scores.first+=4;

scores.second+=4;

}

else if (A == 1 && B == 1) {

scores.first+=24;

scores.second+=24;

}

else if (A == 1 && B == 0) {

scores.first+=0;

scores.second+=20;

}

else if (A == 0 && B == 1) {

scores.first+=20;

scores.second+=0;

}

}

int main() {

mt19937\_64 engine;

engine.seed(time(nullptr));

int32\_t round\_number = -1;

vector<bool> AChoices;

vector<bool> BChoices;

pair<int,int> scores;

uniform\_int\_distribution<> distribution(100,200);

int max\_round = distribution(engine);

cout << "Всегда сотрудничество" << endl;

while (round\_number < max\_round) {

round\_number++;

AChoices.push\_back(alwaysCoop(round\_number, AChoices, BChoices));

BChoices.push\_back(alwaysCoop(round\_number, BChoices, AChoices));

scoreCount(scores, AChoices[round\_number], BChoices[round\_number]);

}

cout << "A: " << scores.first << endl << "B: " << scores.second << endl;

round\_number = -1;

AChoices = {};

BChoices = {};

scores = {};

cout << "Зерклальное против обратного" << endl;

while (round\_number < max\_round) {

round\_number++;

AChoices.push\_back(mirror(round\_number, AChoices, BChoices));

BChoices.push\_back(reverse(round\_number, BChoices, AChoices));

scoreCount(scores, AChoices[round\_number], BChoices[round\_number]);

}

cout << "A: " << scores.first << endl << "B: " << scores.second << endl;

cout << "Сотрудничество против обратного" << endl;

while (round\_number < max\_round) {

round\_number++;

AChoices.push\_back(alwaysCoop(round\_number, AChoices, BChoices));

BChoices.push\_back(reverse(round\_number, BChoices, AChoices));

scoreCount(scores, AChoices[round\_number], BChoices[round\_number]);

}

cout << "A: " << scores.first << endl << "B: " << scores.second << endl;

return 0;

}

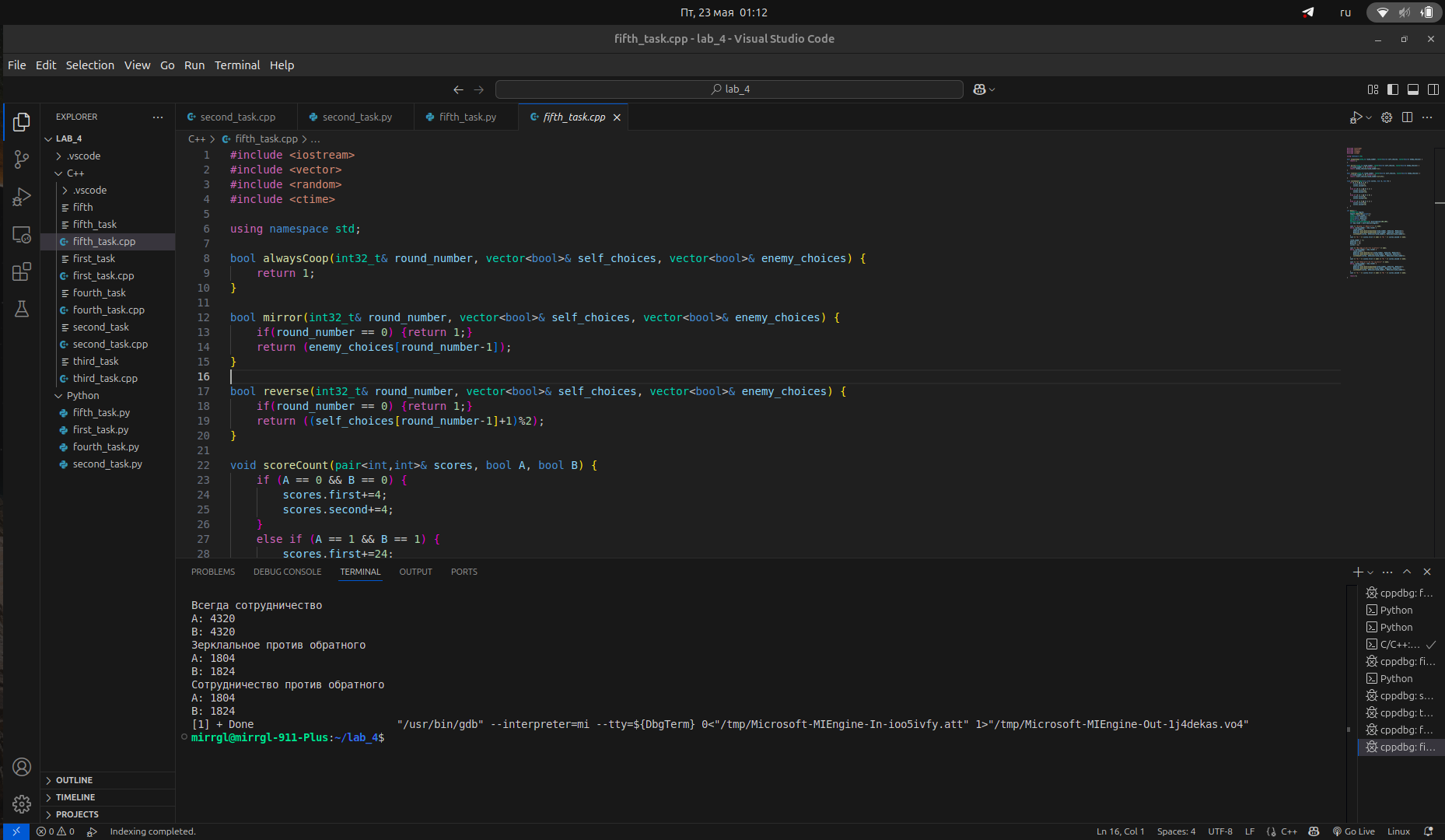


Рисунок 8 — Демонстрация кола на C++

Листинг программы на Python (рисунок 9)

import random

import time

def always\_coop(round\_number, self\_choices, enemy\_choices):

return 1

def mirror(round\_number, self\_choices, enemy\_choices):

if round\_number == 0:

return 1

return enemy\_choices[round\_num-1]

def reverse(round\_number, self\_choices, enemy\_choices):

if round\_number == 0:

return 1

return (self\_choices[round\_num-1] + 1) % 2

def score\_count(a\_score, b\_score, a\_choice, b\_choice):

if a\_choice == 0 and b\_choice == 0:

return a\_score + 4, b\_score + 4

elif a\_choice == 1 and b\_choice == 1:

return a\_score + 24, b\_score + 24

elif a\_choice == 1 and b\_choice == 0:

return a\_score, b\_score + 20

elif a\_choice == 0 and b\_choice == 1:

return a\_score + 20, b\_score

random.seed(time.time())

max\_round = random.randint(100, 200)

print("Всегда сотрудничество")

a\_score, b\_score = 0, 0

a\_choices, b\_choices = [], []

for round\_num in range(max\_round + 1):

a\_choice = always\_coop(round\_num, a\_choices, b\_choices)

b\_choice = always\_coop(round\_num, b\_choices, a\_choices)

a\_choices.append(a\_choice)

b\_choices.append(b\_choice)

a\_score, b\_score = score\_count(a\_score, b\_score, a\_choice, b\_choice)

print(f"A: {a\_score}\nB: {b\_score}\n")

print("Зеркальное против обратного")

a\_score, b\_score = 0, 0

a\_choices, b\_choices = [], []

for round\_num in range(max\_round + 1):

a\_choice = mirror(round\_num, a\_choices, b\_choices)

b\_choice = reverse(round\_num, b\_choices, a\_choices)

a\_choices.append(a\_choice)

b\_choices.append(b\_choice)

a\_score, b\_score = score\_count(a\_score, b\_score, a\_choice, b\_choice)

print(f"A: {a\_score}\nB: {b\_score}\n")

print("Сотрудничество против обратного")

a\_score, b\_score = 0, 0

a\_choices, b\_choices = [], []

for round\_num in range(max\_round + 1):

a\_choice = always\_coop(round\_num, a\_choices, b\_choices)

b\_choice = reverse(round\_num, b\_choices, a\_choices)

a\_choices.append(a\_choice)

b\_choices.append(b\_choice)

a\_score, b\_score = score\_count(a\_score, b\_score, a\_choice, b\_choice)

print(f"A: {a\_score}\nB: {b\_score}")

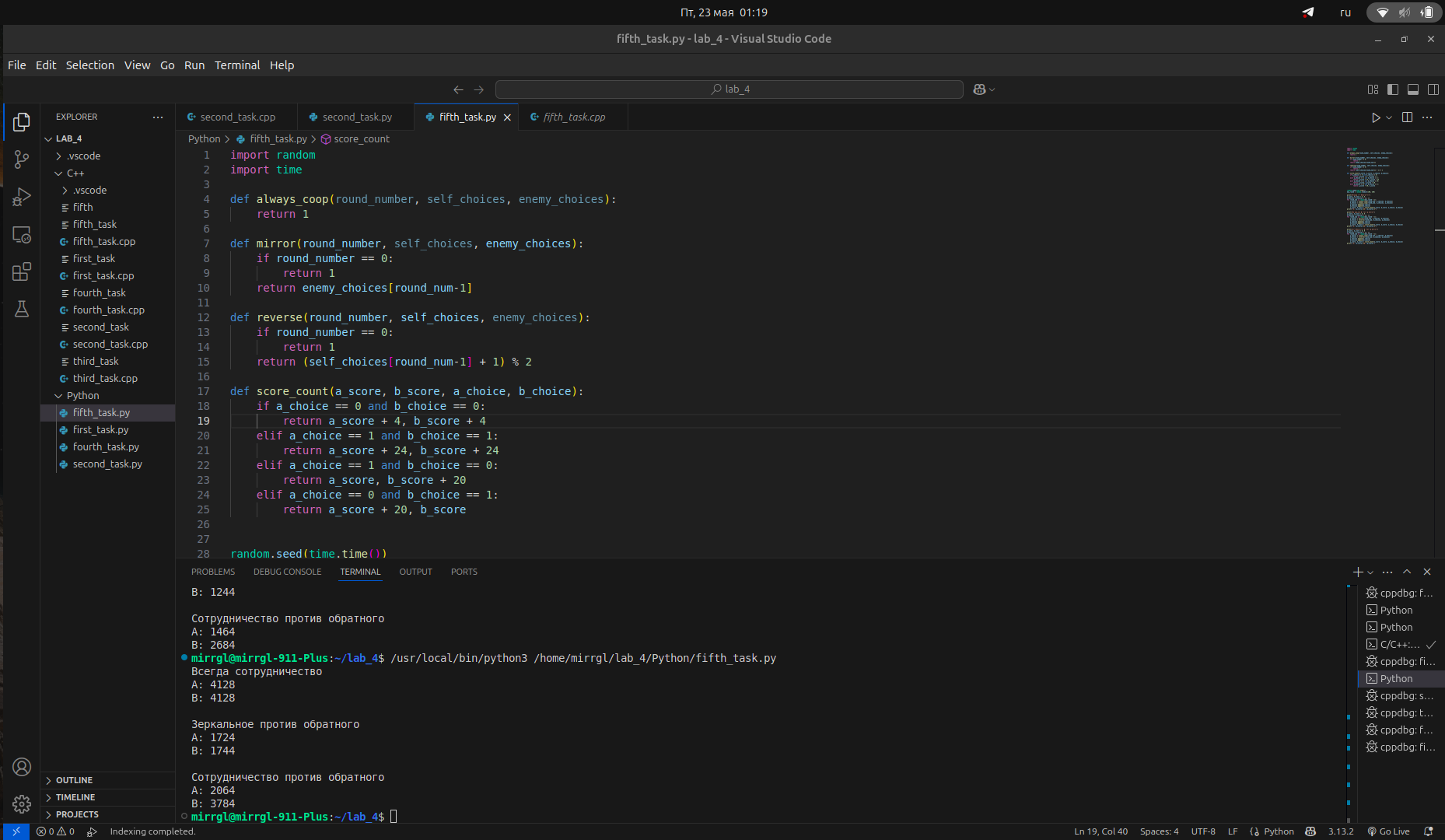


Рисунок 9 — Демонстрация кода на Python