ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ "ВЫСШАЯ ШКОЛА ЭКОНОМИКИ"»

ДОМАШНЕЕ ЗАДАНИЕ №4

Работа студента 2 курса бакалавриата группы БПИ-195 по предмету «Архитектура вычислительных систем»

Выполнил:

Карабаш Радимир Сейранович,

БПИ195-1

Преподаватель:

Доктор технических наук,

Профессор

Легалов А. И

Задание:

Определить множество индексов і, для которых (A[i] - B[i]) или (A[i] + B[i]) являются простыми числами. Входные данные: массивы целых положительных чисел A и B, произвольной длины ≥ 1000. Количество потоков является входным параметром.

Решение:

Так как в программе используется метод, внутри которого находится цикл для проверки выполнения условия, и все потоки принадлежат данному методу для решения общей задачи, было принято решение использовать **итеративный параллелизм** в качестве модели построения приложения.

При запуске программы пользователь видит приветствуещее сообщение с информацией об исполнителе и задании, а затем программа просит ввести пользоваетеля длину двух массивов. После получения информации о длине массивов программа просит пользователя ввести количество потоков для выполнения задачи. Два массива генерируются с помощью генератора псевдослучайных чисел, и логируются в файл input.txt для дальнейшей проверки корректности алгоритма.

Далее вызывается метод **get indexes**, внутри которого происходит распараллеливание программы с помощью директив ОрепМР, который принимает на вход соответствующие два массива, и ссылку на массив с индексами простых чисел. Далее после отработки указанного выше метода список индексов сортируется, логирутся в файл output.txt для дальнейшей проверки корректности алгоритма, и показывается пользователю.

Вывод данных осуществляется в консоль.

Для проверки правильности работы алгоритма реализована логирование входных и выходных данных! (файлы input.txt и output.txt).

Входным параметром для количества потоков является целое число в диапазоне от 1 до 10 включительно.

Описание методов:

- vector<int> generate_array(int size) генерация массива длиной size
- void is_prime(int n, bool& flag) проверка числа n на простое
- bool try_parse(string n) возможно ли привести строку n в целочисленный тип
- int introduce(int& thr_count) получение входных параметров и вывод информативного сообщения
- void log(vector<int> a, vector<int> b) логирование входных данных
- void log(vector<int> a) логирование выходных данных
- void get_indexes(vector<int> a, vector<int> b, vector<int>&
 indexes) получение индексов простых чисел

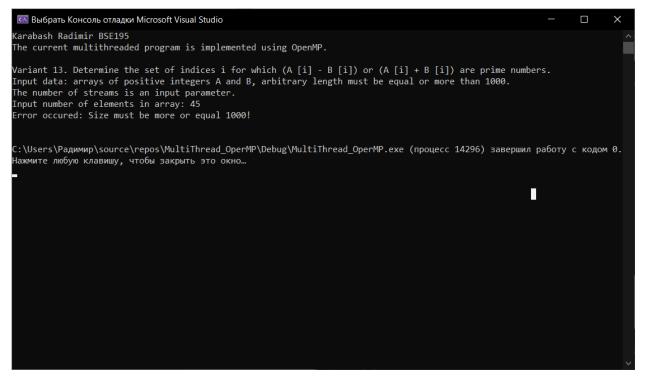
Текст программы приведен ниже:

```
// c++ MultiThread OperMP.cpp -fopenmp
#include <iostream>
#include<vector>
#include<string>
#include<omp.h>
#include<fstream>
#include<algorithm>
using namespace std;
//Генерация массива
vector<int> generate array(int size) {
    vector<int> v1(size);
    for (size t i = 0; i < v1.size(); i++)
        v1[i] = 1 + rand();
    return v1;
// Проверка на простое число
void is prime(int n, bool& flag) {
    for (int i = 2; i \le sqrt(n); i++) {
        if (n % i == 0) {
            flag = false;
            return;
        }
    flag = true;
// Возможно ли записать значение в int
bool try parse(string n) {
    try
```

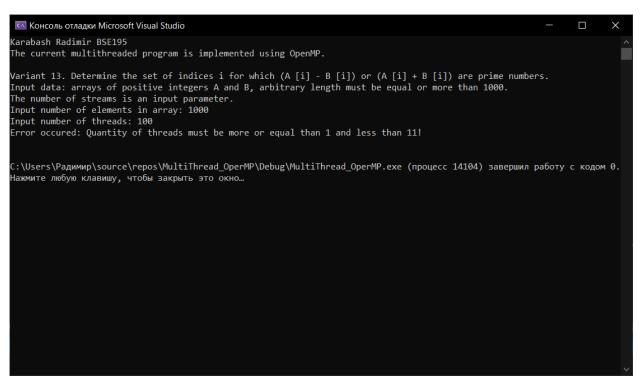
```
{
        int a = stoi(n);
        return true;
    }
    catch (const std::exception&)
        return false;
    }
// Ознакомительные сообщения и получение данных от пользователя
int introduce(int& thr count) {
    std::cout << "Karabash Radimir BSE195\nThe current multithreaded program
is implemented using OpenMP.\n\n";
    std::cout << "Variant 13. Determine the set of indices i for which (A [i]</pre>
- B [i]) or (A [i] + B [i]) are prime numbers.\nInput data: arrays of
positive integers A and B, arbitrary length must be equal or more than 1000.
\nThe number of streams is an input parameter. \n";
    cout << "Input number of elements in array: ";</pre>
    string n;
    cin >> n;
    int size;
    if (try_parse(n))
        size = stoi(n);
        if (size < 1000)
            throw new exception("Size must be more or equal 1000!\n");
    }
    else
        throw new exception("Invalid size of array!\n");
    cout << "Input number of threads: ";</pre>
    n = "";
    cin >> n;
    if (try_parse(n))
        thr count = stoi(n);
        if (thr count < 1 \mid \mid thr count > 10)
            throw new exception("Quantity of threads must be more or equal
than 1 and less than 11!\n");
        }
    }
    else
        throw new exception("Invalid quantity of threads!\n");
    return size;
// Логирование входных и выходных данных
void log(vector<int> a, vector<int> b) {
    fstream file("../input.txt", fstream::out);
    file << "Array A: ";</pre>
    for (size t i = 0; i < a.size(); i++)
        file << to string(a[i]) + " ";</pre>
    file << "\n\nArray B: ";</pre>
    for (size t i = 0; i < b.size(); i++)
        file << to string(b[i]) + " ";
    }
```

```
void log(vector<int> a) {
    fstream file("../output.txt", fstream::out);
    file << "Indexes: ";</pre>
    for (size_t i = 0; i < a.size(); i++)</pre>
        file << to string(a[i]) + " ";
    }
// Получение индексов простых чисел
void get indexes(vector<int> a, vector<int> b, vector<int>& indexes) {
    bool left, right;
    #pragma omp parallel
        #pragma omp for
        for (int i = 0; i < a.size(); i++)
            is prime(abs(a[i] - b[i]), left);
            is prime(abs(a[i] + b[i]), right);
            if (left || right)
            #pragma omp critical
                     indexes.push back(i);
                     //cout << "Thread(" << omp get thread num() << ") Index:"</pre>
<< to string(i) << endl;
        }
int main()
    try {
        int thr count;
        int size = introduce(thr count);
        vector<int> a = generate array(size);
        vector<int> b = generate array(size);
        log(a, b);
        vector<int> indexes;
        omp set num threads(thr count);
        get indexes(a, b, indexes);
        sort(indexes.begin(), indexes.end());
        log(indexes);
        for (size t i = 0; i < indexes.size(); i++)</pre>
            cout << indexes[i] << endl;</pre>
        }
        cout << "Logging of input and output data is implemented to check the
correctness of the algorithm! (files input.txt and output.txt)";
    catch (exception* e) {
        cout << "Error occured: " << (*e).what() << endl;</pre>
}
```

Результаты разработки:



TEST 1



TEST 2

```
Кагаbash Radimir BSE195
The current multithreaded program is implemented using OpenMP.

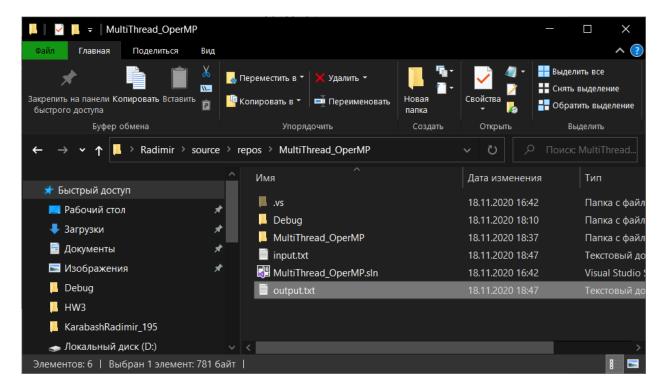
Variant 13. Determine the set of indices i for which (A [i] - B [i]) or (A [i] + B [i]) are prime numbers. Input data: arrays of positive integers A and B, arbitrary length must be equal or more than 1000. The number of streams is an input parameter.

Input number of elements in array: 1000
Input number of threads: 10
```

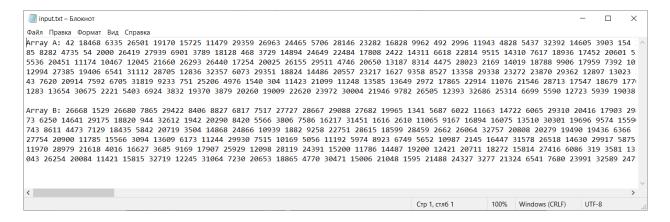
TEST 3.1



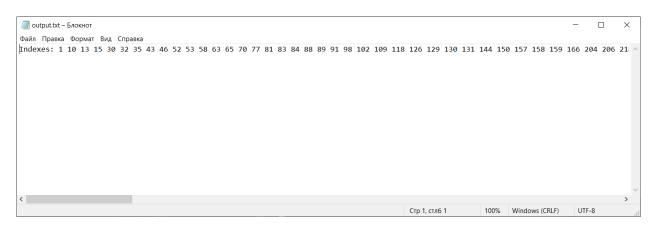
TEST 3.2



TEST 3.3



TEST 3.4



TEST 3.5

```
Кагаbash Radimir BSE195
The current multithreaded program is implemented using OpenMP.

Variant 13. Determine the set of indices i for which (A [i] - B [i]) or (A [i] + B [i]) are prime numbers.

Input data: arrays of positive integers A and B, arbitrary length must be equal or more than 1000.

The number of streams is an input parameter.

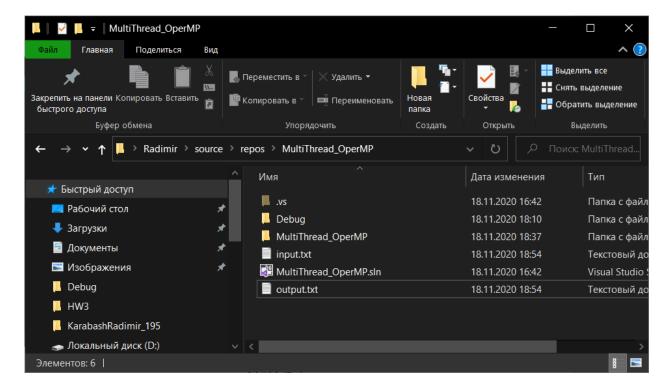
Input number of elements in array: 100000

Input number of threads: 5
```

TEST 4.1

```
🐼 Консоль отладки Microsoft Visual Studio
                                                                                                                                 499867
499868
499869
499870
499871
499877
499884
499899
499908
499909
499917
499928
499929
499937
499947
499948
499953
499956
499960
499963
499965
499970
499978
499989
499997
Logging of input and output data is implemented to check the correctness of the algorithm! (files input.txt and output.t
xt)
....
C:\Users\Pадимир\source\repos\MultiThread_OperMP\Debug\MultiThread_OperMP.exe (процесс 5000) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно…
```

TEST 4.2



TEST 4.3

Файл Правка Формат Вид Справка
Аггау А: 42 18468 6335 26501 19170 15725 11479 29359 26963 24465 5706 28146 23282 16828 9962 492 2996 11943 4828 5437 32392 14605 3903 154 №
85 8282 4735 54 2000 26419 27939 6901 3789 18128 468 3729 14894 24649 22484 17808 2422 14311 6618 22814 9515 14310 7617 18936 17452 20601 5
536 20451 11174 10467 12045 21660 26293 26440 17254 20025 26155 29511 4746 20650 13187 8314 4475 28023 2169 14019 18788 9906 17959 7392 10
12994 27385 19406 6541 31112 28705 12836 32357 6073 29351 18824 14486 20557 23217 1627 9358 8527 13358 29338 23272 23870 29362 12897 13023 .
43 7620 20914 7592 6705 31819 9233 751 25206 4976 1540 304 11423 21099 11248 13585 13649 2972 17865 22914 11076 21546 28713 17547 18679 1771
1283 13654 30675 2221 5403 6924 3832 19370 3879 20260 19009 22620 23972 30004 21946 9782 26505 12333 32686 25314 6699 5590 12723 5939 19038 4099 27831 29293 14601 7279 20800 18853 20449 13883 541 28316 4576 8763 9568 22337 18398 31419 19898 25829 13852 26817 24231 4450 16926 659 30975 3448 5022 18284 26223 29332 2377 28584 26949 12724 17983 22019 14777 9221 31112 32183 23857 19491 8926 4325 6487 19678 15970 11644 75 5012 1561 15835 1841 31498 2786 18530 31541 18806 28792 13393 13211 13550 21579 26980 30972 9278 74 20194 1621 21498 13827 31277 19791 6583 30423 27667 8487 3456 2029 29615 4861 29254 11778 31349 12504 10862 22432 29083 12456 14198 22107 8753 15822 17297 26282 26022 24456 15948 27217 12080 28769 17041 31532 12934 23780 20664 12260 26654 27937 2096 24366 11875 7721 26836 25681 8977 18456 5726 4072 24809 19157 51 313183 29152 23016 8056 3394 8739 15280 19883 1609 12655 3823 32708 24246 1339 145 22291 31340 23155 24605 4624 22226 20079 21725 31982 23 459 25968 8457 9406 11532 29963 26820 7976 10557 7532 31496 8045 30592 26804 3389 18916 7451 12320 6273 25792 29384 10134 1776 24643 23570 12081 12082 12082 24774 25135 27252 22676 13337 16335 21002 2738 9311 5975 7591 20357 1072 32081 10936 32146 17283 1826 18719 13574 4713 13038 4999 12561 12082 14747 25135 27252 22676 13337 16335 21

TEST 4.4

```
output.txt – Блокнот
                                                                                                                                                       Файл Правка Формат Вид Справка
Indexes: 0 1 16 19 31 32 40 45 47 53 70 73 96 99 104 106 107 109 111 112 115 119 127 132 133 135 140 141 155 178 181 198 204 210 213 226 22:
1319 1329 1339 1342 1348 1351 1356 1366 1365 1366 1369 1374 1386 1387 1390 1393 1396 1400 1401 1403 1405 1407 1415 1416 1420 1421 1426 1428
2477 2482 2485 2495 2500 2501 2504 2507 2508 2515 2519 2523 2524 2525 2528 2531 2540 2550 2565 2574 2583 2596 2600 2611 2615 2617 2618 2620
3551 3555 3562 3564 3565 3576 3583 3587 3593 3598 3692 3605 3602 3622 3625 3639 3640 3641 3642 3643 3644 3646 3655 3656 3661 3663 3671 3673 4560 4563 4577 4579 4598 4602 4608 4611 4613 4614 4616 4617 4622 4628 4631 4652 4657 4659 4663 4668 4670 4672 4673 4675 4678 4682 4686 4690
5694 5695 5698 5699 5705 5708 5710 5715 5718 5723 5724 5732 5733 5736 5740 5748 5748 5756 5758 5761 5771 5787 5789 5790 5791 5792 5795 5805
6581 6589 6606 6607 6610 6616 6621 6625 6632 6635 6641 6643 6646 6647 6657 6658 6661 6662 6669 6674 6676 6680 6688 6691 6696 6698 6702 6707
7615 7617 7619 7621 7626 7631 7634 7644 7646 7654 7661 7663 7667 7669 7673 7688 7697 7702 7712 7717 7720 7725 7744 7751 7754 7755 7756 7758
8732 8733 8738 8744 8750 8761 8766 8771 8775 8779 8781 8782 8784 8790 8792 8795 8796 8805 8814 8817 8819 8820 8822 8823 8826 8828 8840 8842
9828 9831 9845 9851 9854 9855 9857 9859 9867 9892 9896 9901 9903 9904 9910 9912 9921 9925 9930 9936 9938 9956 9957 9962 9964 9966 9978 9980
10760 10763 10777 10785 10788 10793 10798 10801 10803 10812 10814 10816 10819 10823 10828 10829 10840 10841 10846 10847 10849 10853 10862 10 11631 11635 11641 11642 11645 11653 11655 11667 11669 11675 11678 11679 11685 11688 11690 11695 11702 11712 11718 11737 11739 11742 11749
530 12533 12536 12539 12544 12545 12549 12550 12551 12553 12563 12571 12578 12582 12583 12591 12596 12597 12598 12602 12603 12607 12611 126
13324 13331 13344 13348 13371 13372 13378 13391 13407 13411 13412 13415 13417 13419 13423 13438 13440 13446 13451 13453 13467 13473 13481 1
3 14274 14283 14289 14291 14294 14298 14303 14309 14318 14320 14321 14325 14326 14338 14342 14347 14349 14350 14355 14360 14362 14366 14368
                         15177 15176 15100 15105 15107 15201 15201 15200 15200 15200 15200 15240 15272 15272 15272 15200 15200 15200 15200 15205 152
                                                                                                      Стр 1, стлб 1
                                                                                                                          100% Windows (CRLF)
```

TEST 4.5