Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

**Лабораторная работа № 3**

« Основы теории чисел и их использование в криптографии»

Выполнил:

Студент: Круглик А.В.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

1. **Описание приложения**

Приложение написано на языке программирования C# и позволяет провести расчет НОД двух либо трех чисел, а также найти все простые числа в заданном промежутке.

Теория чисел или высшая арифметика – раздел математики, изучающий  
натуральные числа и иные похожие величины. В зависимости от используемых методов в теории чисел рассматривают несколько направлений. Нас будут интересовать вопросы делимости целых чисел, вычисления наибольшегообщего делителя (НОД).

1. **Методика выполнения расчетов**

В данной лабораторной работе была поставлена цель создания приложения, позволяющее произвести расчёт НОД двух либо трёх чисел, а также выполнить поиск простых чисел из диапазона. На листинге 2.1 представлен класс, реализующий данную функциональность.

static class NODCompute

{

public static int Compute(int x, int y)

{

while (x != 0 && y != 0)

{

if (x > y)

{

x -= y;

}

else

{

y -= x;

}

}

return Math.Max(x, y);

}

private static bool IsSimple(int x)

{

for (int i = 2; Math.Pow(i, 2) <= x; i++)

{

if (x % i == 0)

{

return false;

}

}

return true;

}

public static void FindSimple(int m, int n)

{

int counter = 0;

if (n < m)

{

Console.WriteLine("Неверный промежуток");

}

Console.Write($"Простые числа интервала [{m},{n}]: ");

for (int i = m; i <= n; i++)

{

if (IsSimple(i))

{

Console.Write(i.ToString() + " ");

counter++;

}

}

Console.WriteLine();

Console.WriteLine($"Количество простых чисел: {counter}");

}

}

public static void proctnumber(int num)

{

bool prost = true;

for (int i = 2; i <= num / 2; i++)

{

if (num % i == 0)

{

prost = false;

break;

}

}

if (prost)

{

Console.WriteLine("Число простое");

}

else

{

Console.WriteLine("Число не простое");

}

}

public static List<uint> SieveEratosthenes(uint n1, uint n)

{

var numbers = new List<uint>();

//заполнение списка числами от 2 до n-1

for (var i = 2u; i < n; i++)

{

numbers.Add(i);

}

for (var i = 0; i < numbers.Count; i++)

{

for (var j = 2u; j < n; j++)

{

//удаляем кратные числа из списка

numbers.Remove(numbers[i] \* j);

}

}

for (int i = 0; i < numbers.Count(); i++)

{

if (numbers[i] < n1)

{

numbers.RemoveAt(i);

i--;

}

}

return numbers;

}

public static bool IsPrimeNumber(int n)

{

var result = true;

if (n > 1)

{

for (int i = 2; i < n; i++)

{

if (n % i == 0)

{

result = false;

break;

}

else

result = true;

}

}

else

result = false;

return result;

}

public static List<uint> TrialDivision(uint n)

{

uint numd = n;

var divides = new List<uint>();

var div = 2u;

while (n > 1)

{

if (n % div == 0)

{

divides.Add(div);

n /= div;

}

else

{

div++;

}

}

int cout3 = 0;

int cout2 = 0;

int cout5 = 0;

int cout7 = 0;

string masseg = "";

foreach (var numbers in divides)

{

if (numbers == 3)

{

cout3++;

}

if (numbers == 2)

{

cout2++;

}

if (numbers == 5)

{

cout5++;

}

if (numbers == 7)

{

cout7++;

}

}

if (cout2 > 1)

{

masseg = masseg + "2 ^" + cout2 + " \* ";

}

if (cout3 > 1)

{

masseg = masseg + "3 ^" + cout3 + " \* ";

}

if (cout5 > 1)

{

masseg = masseg + "5 ^" + cout5 + " \* ";

}

if (cout7 > 1)

{

masseg = masseg + "5 ^" + cout7 + " \* ";

}

if (cout2 == 1)

{

masseg = masseg + "2 \* ";

}

if (cout3 == 1)

{

masseg = masseg + "3 \* ";

}

if (cout5 == 1)

{

masseg = masseg + "5 \* ";

}

if (cout7 == 1)

{

masseg = masseg + "7 \* ";

}

masseg = masseg.TrimEnd(' ');

masseg = masseg.TrimEnd('\*');

masseg = masseg + " = " + numd;

Console.WriteLine(masseg);

return divides;

}

}

Листинг 2.1 – класс, реализующий необходимый функционал

Метод Compute используется для получения НОД двух чисел путём исполнения алгоритма Евклида. Метод IsSimple осуществляет проверку на то, является ли переданное в него в качестве параметра число простым. Метод FindSimple находит все простые числа в диапазоне, границы которого переданы в него в качестве двух параметров.

**3. Результаты работы приложения**

Для выполнения расчетов достаточно необходимо запустить приложение, выбрать необходимый пункт задания и ввести числа. Рисунки 3.1, 3.2, 3.3 показывают необходимые расчеты и вызовы методов, требуемые в данной лабораторной работе.

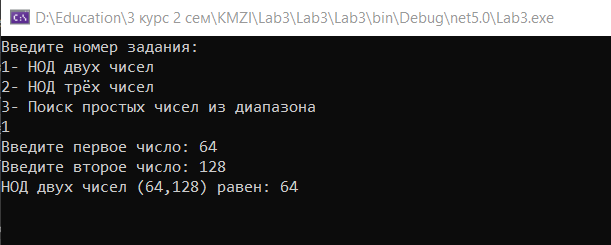


Рисунок 3.1 – Результат работы методе Compute с двумя числами

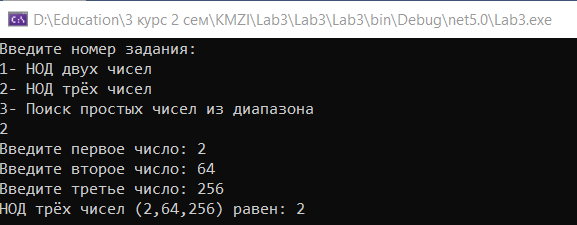


Рисунок 3.2 – Результат работы методе Compute с тремя числами

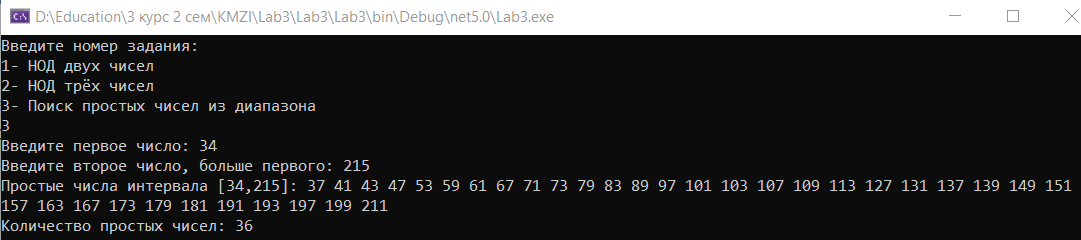


Рисунок 3.3 – Результат работы методе FindSimple

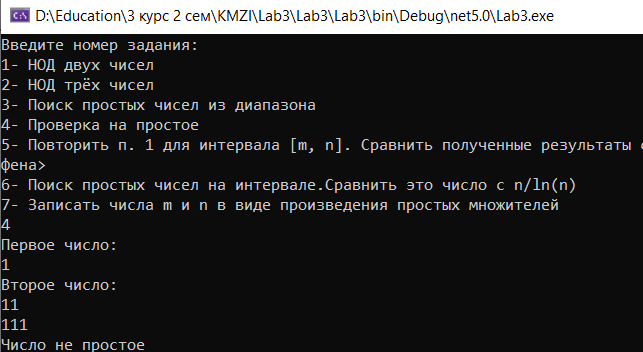


Рисунок 3.4 – Результат работы методе proctnumber

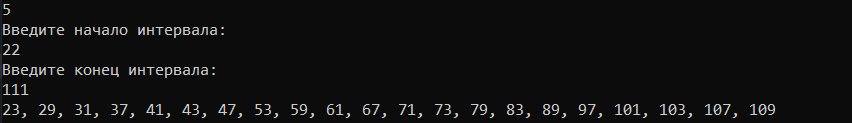


Рисунок 3.5 – Результат работы методе SieveEratosthenes

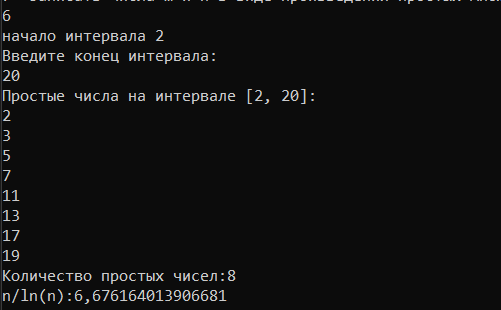


Рисунок 3.6 – Результат работы методе IsPrimeNumber



Рисунок 3.7 – Результат работы методе TrialDivision

Решето Эратосфена от 1 до 20 :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 |
| 16 | 17 | 18 | 19 | 20 |

8- количество простых чисел до 20

Все простые числа до 20: 2, 3, 5, 7, 11, 13, 17, 19.

Зелёные – простые числа.

Красные – составные числа.

**Вывод**

В ходе изучения теоретических материалов лабораторной работы и выполнения её практической части были изучены основы теории чисел, используемые в криптографии. Изучены основные свойства простых и составных чисел, взаимно простых чисел, а также признаки делимости чисел. На практике полученные знания были закреплены через создание собственного приложения, реализующего функционал вычисления НОД двух и трёх чисел и получения списка простых чисел в заданном диапазоне.