Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Информационная безопасность**

Студент: Козека Е. М.

ФИТ 3 курс 4 группа

Преподаватель: Нистюк О. А.

Минск 2025

**Лабораторная работа №3. Основы теории чисел и их использование в криптографии**

**Цель:** приобретение практических навыков выполнения операций с числами для решения задач в области криптографии и разработка приложений для автоматизации этих операций.

**Задачи:**

1. Закрепить теоретические знания по высшей арифметике.
2. Научиться практически решать задачи с использованием простых и взаимно простых чисел, вычислений по правилам модулярной арифметики и нахождению обратных чисел по модулю.
3. Ознакомиться с особенностями реализации готового программного средства L\_PROST и особенностями выполнения с его помощью операций над простыми числами.
4. Разработать приложение для реализации указанных преподавателем операций с числами.
5. Результаты выполнения лабораторной работы оформить в видеописания разработанного приложения, методики выполнения эксперимента с использованием приложения и результатов эксперимента.

**Теоретические сведения**

Множество всех целых чисел (обозначим буквой Z) есть набор всех действительных чисел без дробной части: {..., –3, –2, –1, 0, 1, 2, 3, ...}.

Натуральные числа являются подмножеством целых чисел и образуют множество N: {1, 2, 3, ...}.

Делимость – одно из основных понятий теории чисел. Если для некоторого целого числа a и натурального числа b существует целое число q, при котором bq = a, то говорят, что число a делится на b. В этом случае b называется делителем числа a, а a называется кратным числу b.

Делитель a называется собственным делителем числа b, если 1 < |a| < |b|, и несобственным – в противном случае.

Всякое целое число а можно представить с помощью положительного целого числа b равенством вида а = bq + r, 0 ≤ r ≤ b. Число q называется неполным частным, а число r – остатком отделения а на b.

Натуральное число n называется простым, если n > 1 и не имеет положительных делителей, отличных от 1 и n.

Любое составное число представляется уникальным образом в виде произведения простых чисел; иначе еще говорят, что разложение числа на простые множители однозначно.

Всякое натуральное число n, кроме 1, можно представить как произведение простых множителей: n = p1p2p3...pz, z > 1.

Простых чисел бесконечно много, причем существует примерно n / ln(n) простых чисел, меньших числа n.

Наименьший простой делитель составного числа n не превышает √n, поэтому для проверки простоты числа достаточно проверить его делимость на 2 и на все нечетные (а еще лучше простые) числа, не превосходящие √n.

Любое четное число, большее 2, представимо в виде суммы двух простых чисел, а любое нечетное, большее 5, представимо в виде суммы трех простых чисел.

Для любого натурального n, большего 1, существует хотя бы одно простое число на интервале от n до 2n.

Натуральное число n называется составным, если n > 1 и имеет по крайней мере один положительный делитель, отличный от 1 и n.

Положительный наименьший собственный делитель составного числа n есть простое число.

Если два простых числа отличаются на 2, то их называют числами-близнецами.

Первый алгоритм нахождения простых чисел, не превышающих n, был придуман Эратосфеном во II в. до н. э. и известен сейчас как «решето Эратосфена». Его суть в последовательном исключении из списка целых чисел от 1 до n (или из сокращенного диапазона, например от m до n, 1 < m ≤ n) чисел, кратных 2, 3, 5 и другим простым числам, уже найденным «решетом».

Наибольшее целое число, которое делит без остатка числа a и b, называется наибольшим общим делителем этих чисел – НОД (a, b). Простым и эффективным средством вычисления НОД (a, b) является алгоритм Евклида.

Взаимно простыми являются целые числа, наибольший общий делитель которых равен 1.

Целые числа a и b взаимно просты тогда и только тогда, когда существуют такие целые числа u и v, что выполняется равенство:

аu + bv = 1.

Если НОД (a, b) = d, то справедливо следующее соотношение (соотношение Безу):

аu + bv = d.

Количество натуральных чисел, не превосходящих n и взаимно простых с n, называется функцией Эйлера и обозначается φ(n).

Если p – простое число, то φ(p) = p – 1, если числа p и q являются простыми и p ≠ q, то φ(p) = (p – 1)(q – 1).

Модулярная арифметика так же коммутативна, ассоциативна и дистрибутивна, как и обычная арифметика. В силу этих свойств сравнения можно почленно складывать, вычитать, умножать, возводить в степень.

Малая теорема Ферма. Если n – простое число, а число а не кратно n, то справедливо:

an ≡ 1 mod n.

В соответствии с обобщением Эйлера приведенной теоремы, если НОД(а, n) = 1, то справедливо:

aφ(n) mod n ≡ 1

Последнее выражение можно переписать в следующем виде:

а–1 mod n ≡ aφ(n) – 1 mod n.

**Ход работы**

**Задание 1.** Используя L\_PROST, найти все простые числа в интервале [2, n]. Значение n соответствует варианту из табл. 1.2, указанному преподавателем. Подсчитать количество простых чисел в указанном интервале. Сравнить это число с n/ln(n).

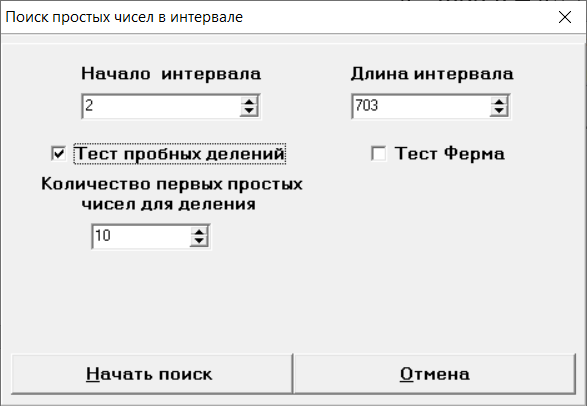


Рисунок 3.1 – Ввод данных для поиска в интервале [2, n]

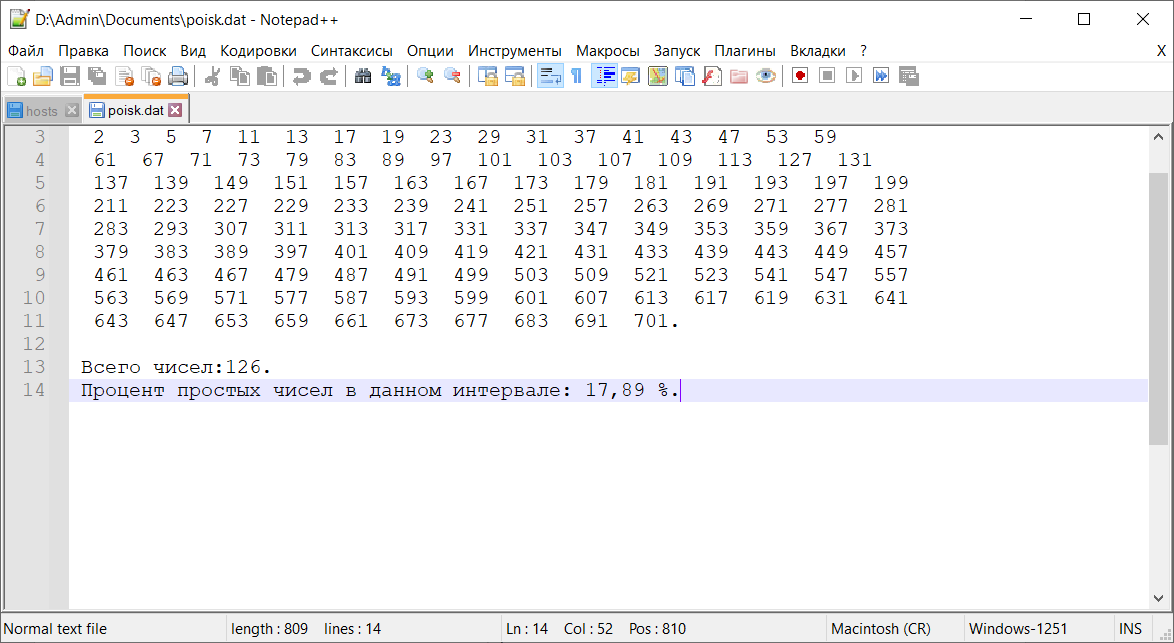


Рисунок 3.2 – Результат поиска простых чисел в интервале [2, n]

Предполагаемое количество простых чисел: n/ln(n) = 703/ln(703) = 107.24.

**Задание 2.** Повторить п. 1 для интервала [m, n]. Сравнить полученные результаты с «ручными» вычислениями, используя «решето Эратосфена».

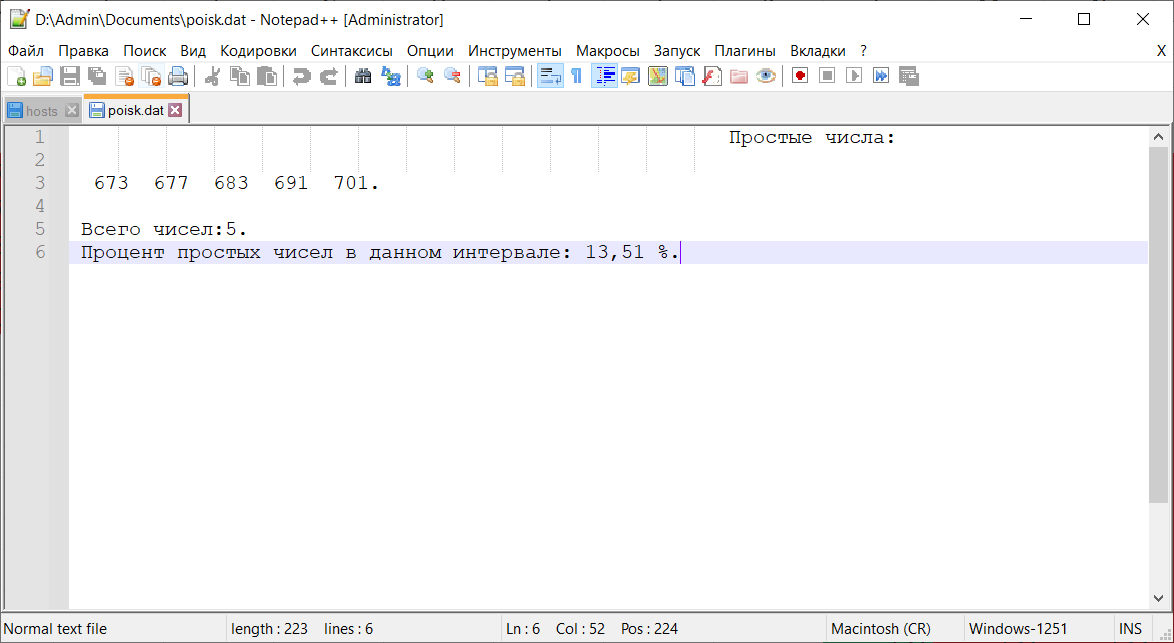


Рисунок 3.3 – Результат поиска простых чисел в интервале [m, n]

На рисунке 3.4 изображено вручную высчитанное решето Эратосфена, где красным выделены вычеркнутые в ходе алгоритма числа, а зелёным – оставшиеся простые числа в диапазоне [*m*, *n*].

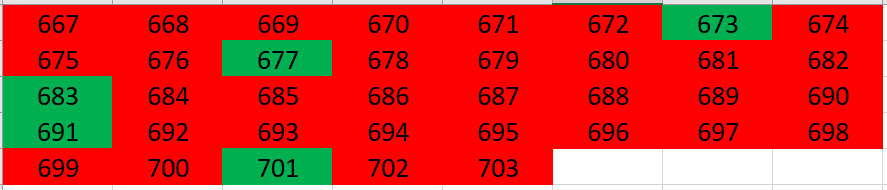


Рисунок 3.4 – Решето Эратосфена в диапазоне [*m*, *n*]

Аналогично высчитывается решето Эратосфена для чисел в диапазоне [2, *n*], что отражено на рисунке 3.5.

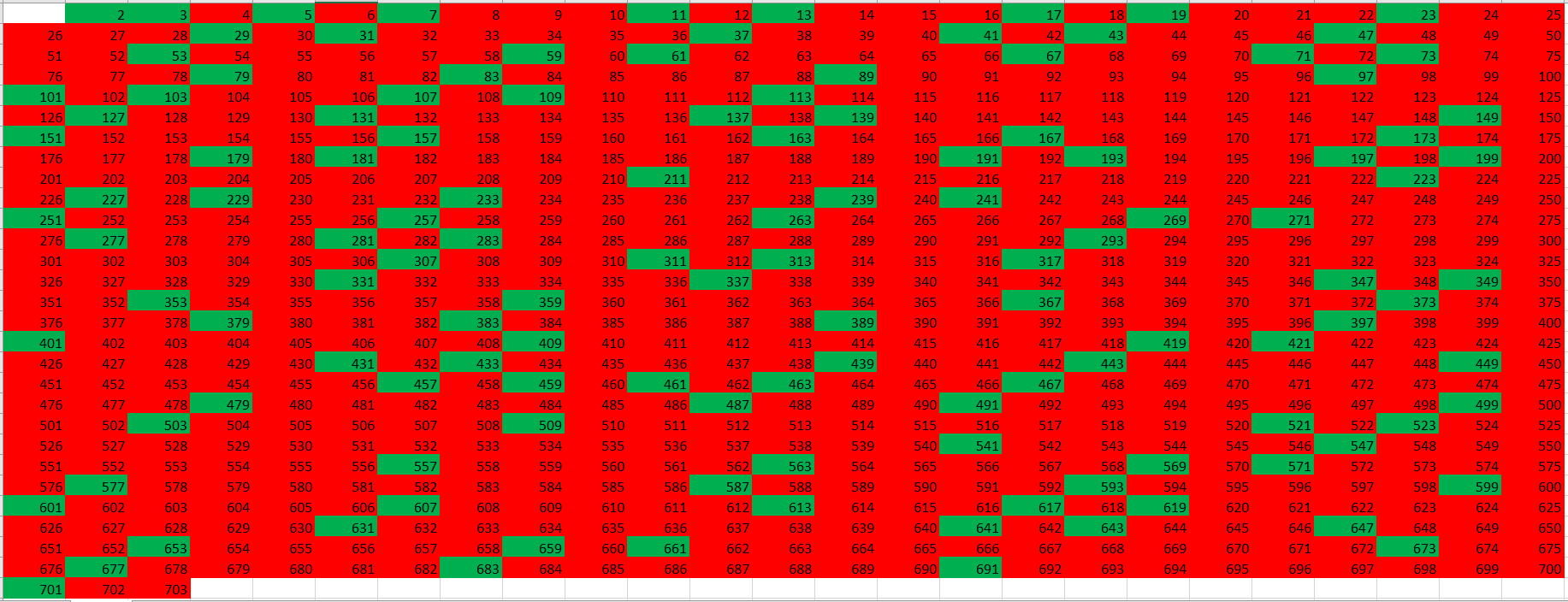


Рисунок 3.5 – Решето Эратосфена в диапазоне [*2*, *n*]

**Задание 3.** Записать числа *m* и *n* в виде произведения простых множителей (форма записи – каноническая).

Любое число *n* можно представить в следующем виде, называемое канонической формой записи числа:



где *p1*, *p2*,…, *pn* – разные простые множители числа, *a1*, *a2*, …, *an* – степени данных простых множителей.

*m* = 667 = 23 \* 29

*n* = 703 = 19 \* 37

**Задание 4.** Проверить, является ли число, состоящее из конкатенации цифр m ǀǀ n, простым.

667703 = 1 \* 667703 = 59 \* 11317

Конкатенация чисел 667 и 703 не является простым числом.

**Задание 5.** Найти НОД (*m*, *n*).

703 = 667 \* 1 + 36

667 = 36 \* 18 +19

36 = 19 \* 1 + 17

19 = 17 \* 1 + 2

17 = 2 \* 8 + 1

2 = 1 \* 2 + 0

НОД (*m*, *n*) = 1

**Задание 6.** Разработать приложение в соответствии с целью лабораторной работы. Приложение должно реализовывать следующие операции:

* вычислять НОД двух либо трех чисел;
* выполнять поиск простых чисел.

С помощью созданного приложения выполнить задания по условиям п. 1 и 2.

const sieveOfEratosthenes = (m, n) => {

let arr = Array.from({ length: (n - m + 1) }, (value, index) =>

m + index);

let square = Math.round(Math.sqrt(n));

let primes = [];

for (let i = 2; i <= square; i++) {

if (isPrime(i)) {

primes.push(i);

}

}

let result = [];

for (let i = 0; i < arr.length; i++) {

let isPrime = true;

for (let j = 0; j < primes.length; j++) {

if (primes[j] \* primes[j] > arr[i])

break;

if (arr[i] % primes[j] === 0) {

isPrime = false;

break;

}

}

if (isPrime && arr[i] > 1)

result.push(arr[i]);

}

return result;

};

Листинг 3.1 – Функция для нахождения простых чисел в интервале

const primeFactors = (n) => {

    const factors = [];

    let divisor = 2;

    while (n >= 2) {

        if (n % divisor === 0) {

            factors.push(divisor);

            n /= divisor;

        }

        else {

            divisor++;

        }

    }

    return factors.join(' \* ');

};

Листинг 3.2 – Функция для разбиения числа на простые множители

const isPrime = (a) => {

    if (a < 2) return false;

    let square = Math.round(Math.sqrt(a));

    for (let i = 2; i <= square; i++) {

        if (a % i === 0) return false;

    }

    return true;

};

Листинг 3.3 – Функция для определения простоты числа

const gcd = (a, b, c = null) => {

    if (c === null) {

        let q = a % b;

        if (q === 0) return b;

        return gcd(b, q);

    } else {

        let gcdAB = gcd(a, b);

        return gcd(gcdAB, c);

    }

};

Листинг 3.4 – Функция для нахождения НОД чисел

Результат работы приложения:

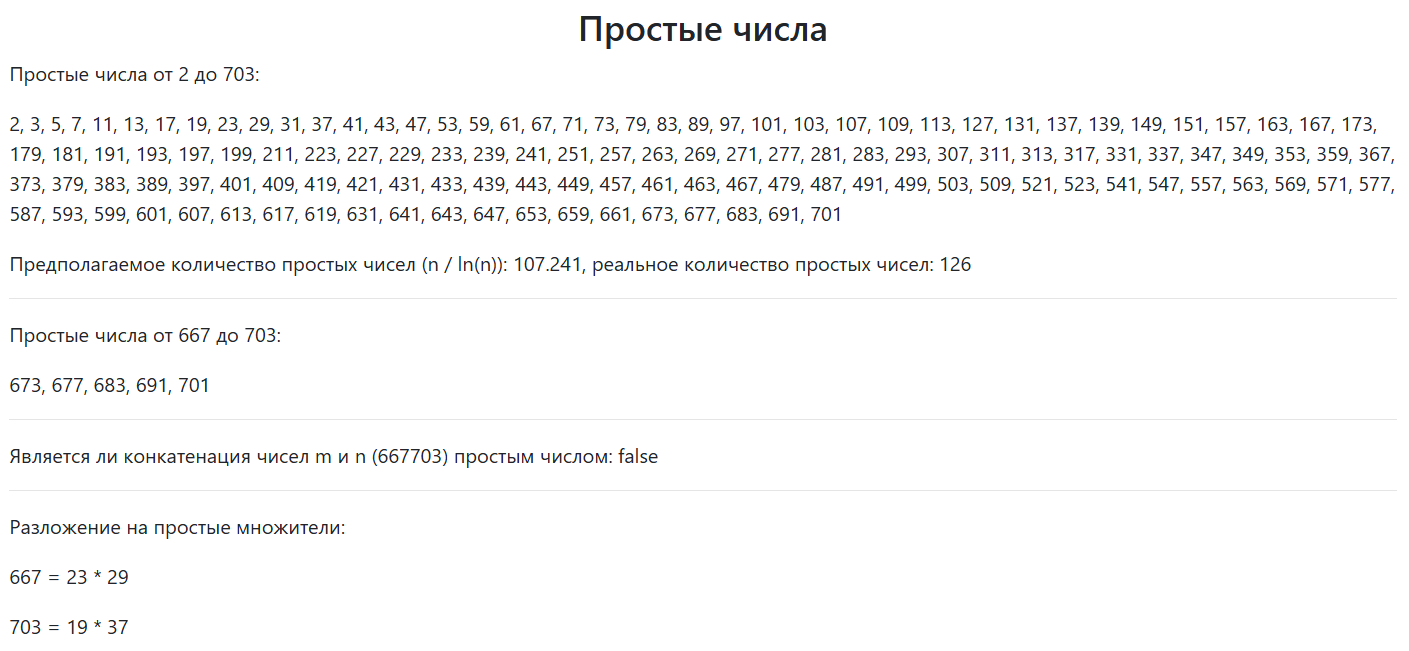


Рисунок 3.6 – Работа с простыми числами

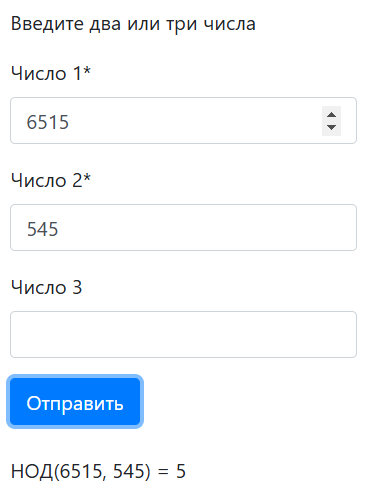


Рисунок 3.7 – Нахождение НОД двух чисел

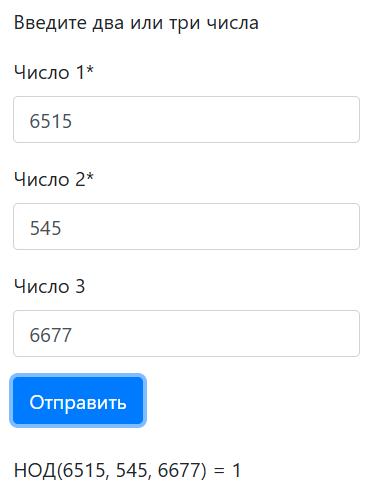


Рисунок 3.8 – Нахождение НОД трех чисел

**Вывод:** в ходе выполнения лабораторной работы были освоены основы теории чисел, необходимые для применения в криптографии. Это включало в себя изучение свойств простых и составных чисел, взаимной простоты чисел, а также критериев делимости. Целью работы было укрепление теоретических знаний в области высшей арифметики и приобретение навыков решения практических задач, включающих простые и взаимно простые числа, а также операции модулярной арифметики и нахождение обратных чисел по модулю. В результате работы было разработано приложение, позволяющее выполнять такие операции с числами как вычисление НОД и нахождение простых чисел.