МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3 НА ТЕМУ:**

**Основы теории чисел и их использование в криптографии**

Выполнила студентка 3 курса 4 группы

Сятковская Екатерина Дмитриевна

Минск 2023

**Задание 1:** Разработать авторское приложение в соответствии с целью лабораторной работы. Разработать авторское приложение в соответствии с целью лабораторной работы. Приложение должно реализовывать следующие операции:

• вычислять НОД двух либо трех чисел;

• выполнять поиск простых чисел.

Для выполнения этого задания был использован следующий код на языке *python*:

def get\_gcd2(a, b):

    while a!= 0 and b !=0:

        if a > b:

            a = a % b

        else:

            b = b % a

    return a + b

def get\_gcd3(a, b, c):

    return get\_gcd2(get\_gcd2(a,b),c)

def get\_prime(min, max):

    sieve = []

    a = [0] \* (max + 1)

    for i in range(max + 1):

        a[i] = 1

    for i in range(2, int(sqrt(max)) + 1):

        if a[i] == 1:

            j = 2

            while i \* j <= max:

                a[i \* j] = 0

                j = j + 1

    for i in range(2, max + 1):

        if a[i] == 1:

            # print(i, end = ' ')

            sieve.append(i)

    sieve = list(filter(lambda num: num >= min, sieve))

    return(sieve)

Листинг 1 – Задание 1

**Задание 2:** С помощью приложения выполнить задания по условиям п. 1 и 2.

Для выполнения этого задания был использован следующий код на языке *python*:

print('\nПоиск простых чисел в интервале [2, n]')

print(get\_prime(2,n))

print('Количество простых чисел в указанном интервале:', len(get\_prime(2,n)))

print('n/ln(n)', n / log(n))

print('\nПоиск простых чисел в интервале [m, n]')

print(get\_prime(m,n))

print('Количество простых чисел в указанном интервале:', len(get\_prime(m,n)))

Листинг 2 – Задание 2

Количество подсчитанных простых чисел в интервале примерно совпадает с *n/ln(n)*.

**Задание 3:** Записать числа m и n в виде простых множителей (использовать каноническую форму записи).

Для выполнения этого задания был использован следующий код на языке *python*:

def get\_factors(n):

    result = []

    d = 2

    while d \* d <= n:

        if n % d == 0:

            result.append(d)

            n //= d

        else:

            d += 1

    if n > 1:

        result.append(n)

    canon = str(n) + ' = ' + str(result[0])

    for i in range (1, len(result)):

        canon += ' \* ' + str(result[i])

    return canon

Листинг 3 – Задание 3

**Задание 4:** Проверить, является ли число, состоящее из конкатенации цифр *m* || *n* простым. Для выполнения этого задания код, разработанный в предыдущем задании был дополнен следующим кодом:

def is\_prime(n):

    k = 0

    for i in range(2, n // 2+1):

        if (n % i == 0):

            k = k + 1

    if k <= 0:

        return True

    else:

        return False

Листинг 4 – Задание 4

**Задание 5:** Найдите НОД (*m*, *n*). Для выполнения этого задания код, разработанный в предыдущем задании был дополнен следующим кодом:

print('\nНОД (m, n)', get\_gcd2(m, n))

Листинг 5 – Задание 5

Результат работы итогового кода:

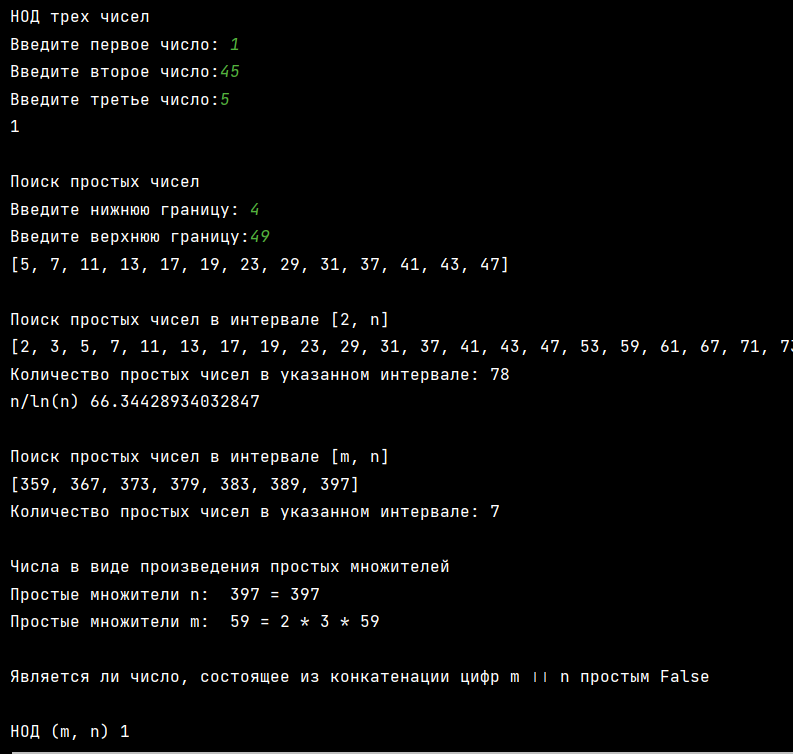


Рисунок 1 – Результат работы программы

**Вывод:** в ходе лабораторной работы были приобретены практические навыков выполнения операций с числами для решения задач в области криптографии и разработка приложений для автоматизации этих операций.