Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Основы теории чисел и их использование в криптографии

Студент: Валдайцев А. Д.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М. Г.

Минск 2023

# Вычисление наибольшего общего делителя

Наибольшее целое число, которое делит без остатка числа *a* и *b*, называется наибольшим общим делителем этих чисел – НОД (*a, b*).

Наибольший общий делитель можно найти с помощью алгоритма Евклида, который заключается в том, чтобы от наибольшего числа из двух отнимать наименьшее, пока одно из них не станет равно нулю, а после вернуть наибольшее из двух число.

Для вычисления наибольшего общего делителя двух чисел реализована следующая функция, представленная на рисунке 1.1:

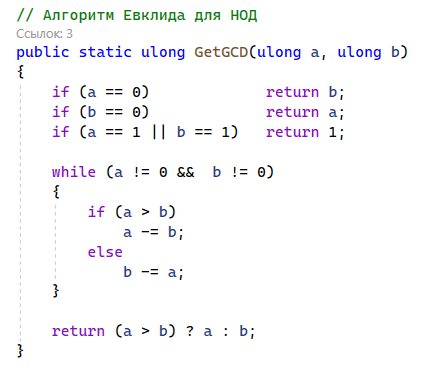


Рисунок 1.1 – Функция нахождения НОД двух чисел

Чтобы найти НОД нескольких чисел (например, *a, b, c*), достаточно найти НОД двух чисел (например, НОД (*a, b*) = *d*), потом НОД полученного (НОД (*a, b*)) и следующего числа (НОД (*c, d*)), и так далее.

Следовательно, для нахождения НОД трёх чисел достаточно в качестве одного из параметров указать вызов функции нахождения НОД. Таким образом получен НОД трёх чисел. Вывод данной функции, а также функции нахождения НОД (*m, n*), где *m* = 421, *n* = 457, представлен на рисунке 1.2.

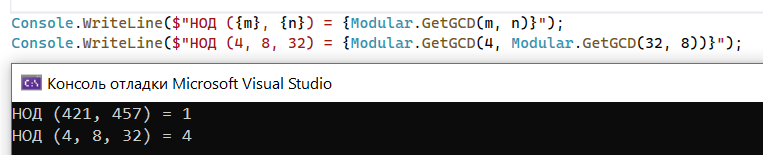


Рисунок 1.2 – Вывод НОД двух и трех чисел

# Поиск простых чисел

Для нахождения всех простых чисел не больше заданного числа *n* используется алгоритм, называемый «решетом Эратосфена». В соответствии с этим алгоритмом нужно выполнить следующие шаги:

1. выписать подряд все целые числа от двух (либо от *m*) до *n* (2, …, *n*). Пусть некоторая переменная (например, *s*) изначально равна 2, то есть первому простому числу;
2. удалить из списка числа от 2*s* до *n*, считая шагами по *s* (это будут числа, кратные *s*: 2*s*, 3*s*, 4*s*, …);
3. найти первое из оставшихся чисел в списке, большее чем *s*, и присвоить значению переменной *s* это число;
4. повторять шаги 2 и 3, пока возможно.

Для реализации данного алгоритма созданы следующие функции, представленные на рисунке 2.1:

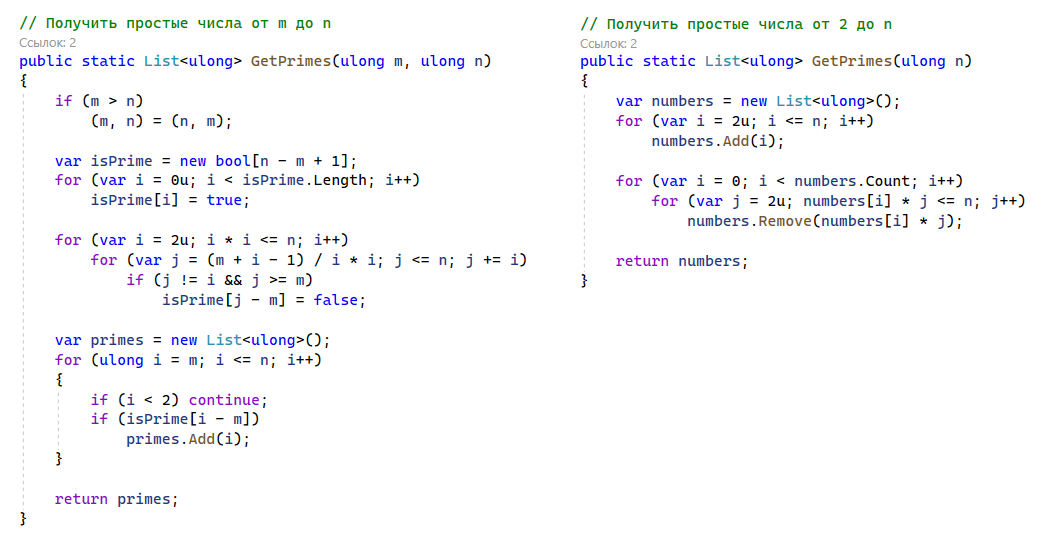


Рисунок 2.1 – Функция поиска простых чисел

Вывод данных функций для поиска простых чисел в интервалах [2, *n*] и [*m*, *n*], а также количество простых чисел в этих диапазонах и значение *n* / ln(*n*) представлены на рисунке 2.2.

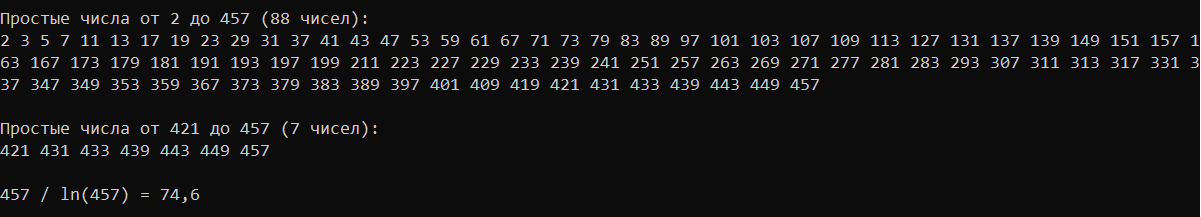


Рисунок 2.2 – Вывод функции поиска простых чисел

Для проверки полученных значений можно воспользоваться ручным вычислением решета Эратосфена. На рисунке 2.3 изображено вручную высчитанное решето Эратосфена, где красным выделены вычеркнутые в ходе алгоритма числа, а зелёным – оставшиеся простые числа в диапазоне [*m*, *n*].

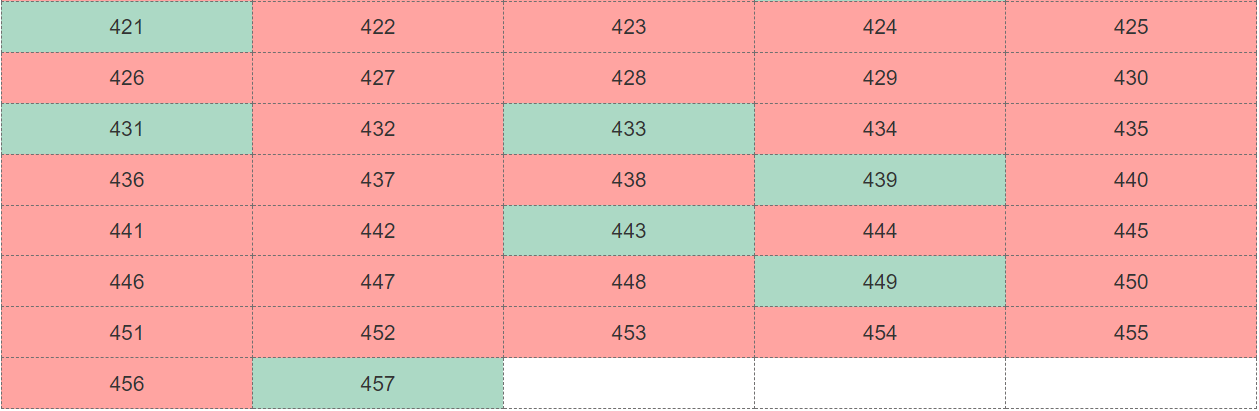


Рисунок 2.3 – Решето Эратосфена в диапазоне [*m*, *n*]

Аналогично высчитывается решето Эратосфена для чисел в диапазоне [2, *n*], равное 88, что отражено на рисунке 2.4.

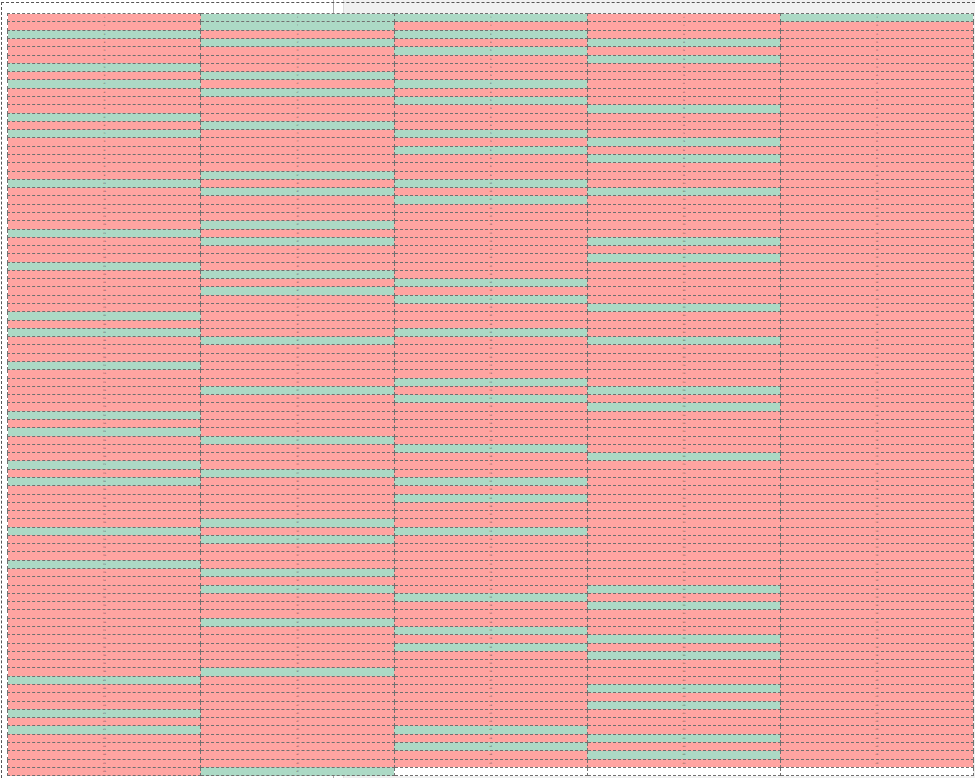


Рисунок 2.4 – Решето Эратосфена в диапазоне [2, *n*]

# Каноническая форма записи числа

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

 (1.1)

где *p1*, *p2*,…, *pn* – разные простые множители числа, *a1*, *a2*, …, *an* – степени данных простых множителей. Также данные множители можно записывать подряд, по возрастанию, без использования степеней.

Для реализации представления числа в канонической форме реализована следующая функция, представленная на рисунке 3.1.

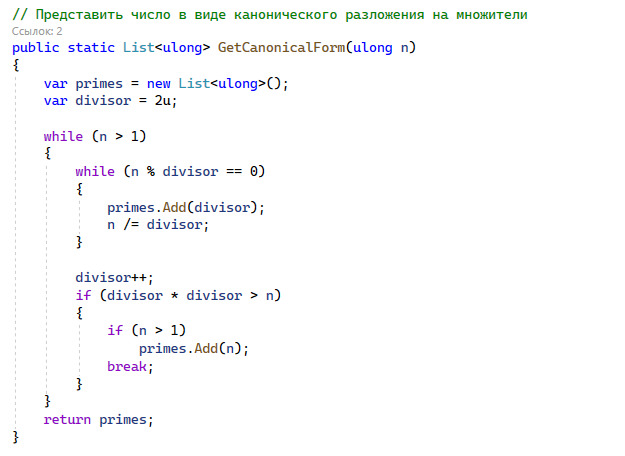


Рисунок 3.1 – Функция представления числа в канонической форме

Данная функция принимает в качестве параметра некоторое число *n* и последовательно делит его на числа от 2 до *√n*. Эти числа записываются в список, который возвращает данная функция. По данному списку в дальнейшем можно циклически пройти и вывести все числа из канонического разложения. Вывод данной функции представлен на рисунке 3.2.

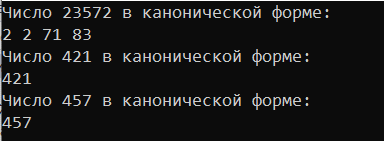


Рисунок 3.2 – Вывод канонической формы числа

# Проверка числа на простоту

Для проверки числа на простоту можно использовать следующий алгоритм: в цикле чисел от 2 до *√n* проверить, делится ли исходное число на одно из них. Если исходное число не делится ни на одно в заданном диапазоне, то оно является простым. Реализация функции проверки числа на простоту представлена на рисунке 4.1.

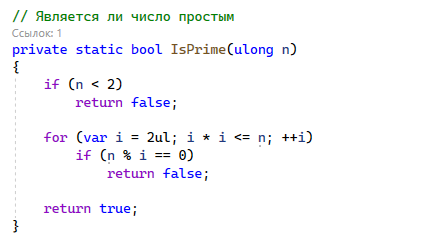


Рисунок 4.1 – Функция проверки числа на простоту

Для выполнения задания проверки на простоту числа, состоящего из конкатенации цифр, из которых состоят числа *m* и *n*, реализована следующая функция, представленная на рисунке 4.2.

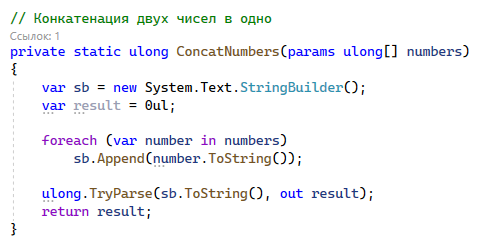


Рисунок 4.2 – Функция конкатенации двух чисел

Далее, необходимо вызвать метод проверки числа на простоту, передав в параметры возвращаемое значение функции конкатенации чисел. Вывод данной функции для чисел *m* = 421 и *n* = 457 представлен на рисунке 4.3.

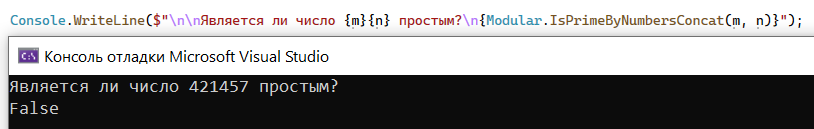


Рисунок 4.3 – Вывод функции проверки на простоту конкатенации чисел

**Вывод:** в данной лабораторной работе были приобретены практические навыки выполнения операций с числами для решения задач в области криптографии, и разработано приложение для автоматизации этих операций.