Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Основы теории чисел и их использование в криптографии

Студент: Яшный Н. С.

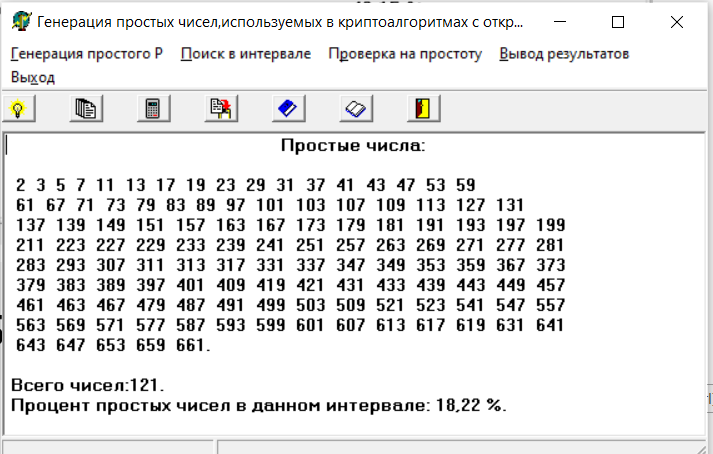
ФИТ 3 курс 4 группа

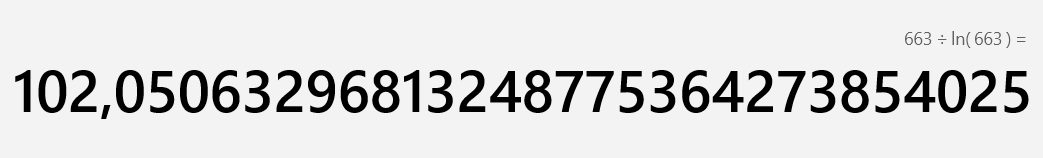
Минск 2024

**Вариант 14**

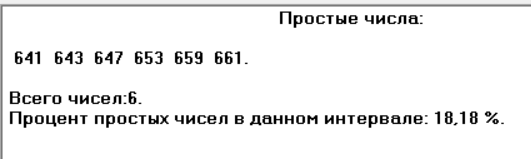
**Первая часть**

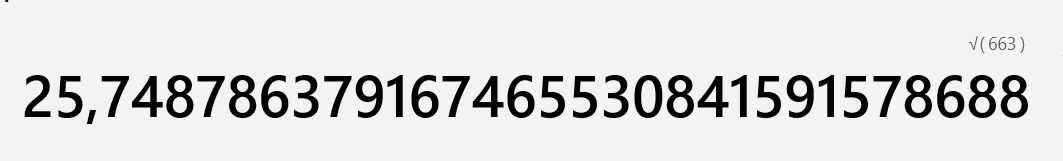
**Задание 1**





**Задание 2**



Проверим вручную через решето эратосфена  


Берём число 26. Простые числа из диапазона 2 – 26: 2, 3, 5, 7, 11, 13, 17, 19, 23.

Все четные делятся на 2, они не подходят.

633 делится на 3

635 делится на 5

637 делится на 7

639 делится на 3

641 простое

643 простое

645 делится на 5

647 простое

649 делится на 11

651 делится на 7

653 простое

655 делится на 5

657 делится на 3

659 простое

661 простое

663 делится на 3

Таким образом простыми являются 641, 643, 647, 653, 659, 661

**Задание 3**  
Число 632 в канонической форме: 2^3 \* 79

Число 633 в канонической форме: 3 \* 211

Число 634 в канонической форме: 2 \* 317

Число 635 в канонической форме: 5 \* 127

Число 636 в канонической форме: 2^2 \* 3 \* 53

Число 637 в канонической форме: 7^2 \* 13

Число 638 в канонической форме: 2 \* 11 \* 29

Число 639 в канонической форме: 3^2 \* 71

Число 640 в канонической форме: 2^7 \* 5

Число 641 в канонической форме: 641

Число 642 в канонической форме: 2 \* 3 \* 107

Число 643 в канонической форме: 643

Число 644 в канонической форме: 2^2 \* 7 \* 23

Число 645 в канонической форме: 3 \* 5 \* 43

Число 646 в канонической форме: 2 \* 17 \* 19

Число 647 в канонической форме: 647

Число 648 в канонической форме: 2^3 \* 3^4

Число 649 в канонической форме: 11 \* 59

Число 650 в канонической форме: 2 \* 5^2 \* 13

Число 651 в канонической форме: 3 \* 7 \* 31

Число 652 в канонической форме: 2^2 \* 163

Число 653 в канонической форме: 653

Число 654 в канонической форме: 2 \* 3 \* 109

Число 655 в канонической форме: 5 \* 131

Число 656 в канонической форме: 2^4 \* 41

Число 657 в канонической форме: 3^2 \* 73

Число 658 в канонической форме: 2 \* 7 \* 47

Число 659 в канонической форме: 659

Число 660 в канонической форме: 2^2 \* 3 \* 5 \* 11

Число 661 в канонической форме: 661

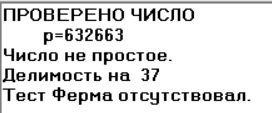
Число 662 в канонической форме: 2 \* 331

Число 663 в канонической форме: 3 \* 13 \* 17

Число 632 в канонической форме: 2^3 \* 79

Число 663 в канонической форме: 3 \* 13 \* 17

**Задание 4**



**Задание 5**

По алгоритму Евклида

a = 663, b = 632

663 = 632 \* 1 + 31

632 = 31 \* 20 + 12

31 = 12 \* 2 + 7

12 = 7 \* 1 + 5

7 = 5 \* 1 + 2

5 = 2 \* 2 + 1

2 = 1 \* 2 + 0  
последний ненулевой = 1. Ответ 1

**Вторая часть**

# 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:

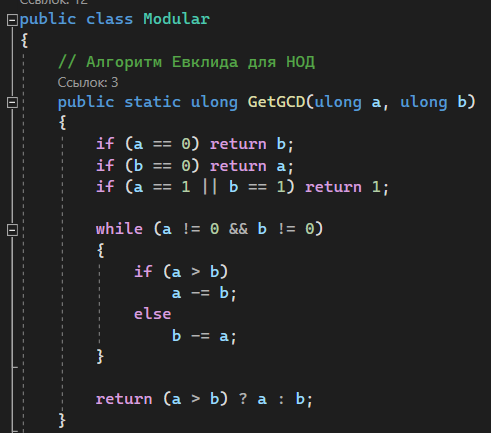


Рисунок 1.1 – Функция нахождения НОД двух чисел

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* = 632, *n* = 663, представлен на рисунке 1.2.

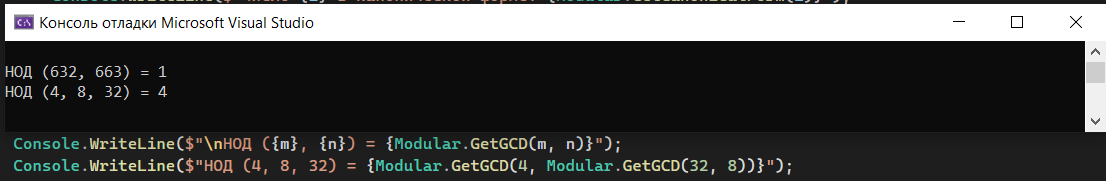


Рисунок 1.2 – Вывод НОД двух и трех чисел

# 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:

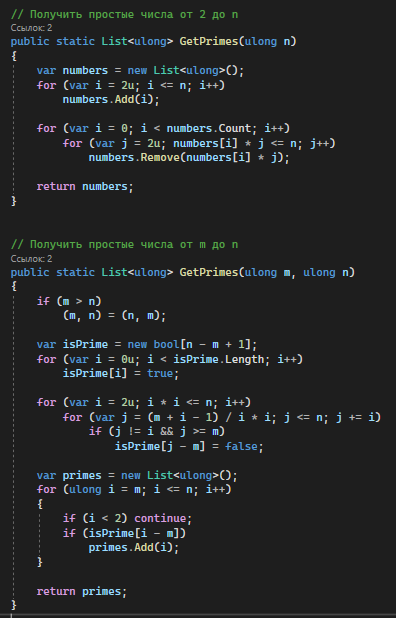


Рисунок 2.1 – Функция поиска простых чисел

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.

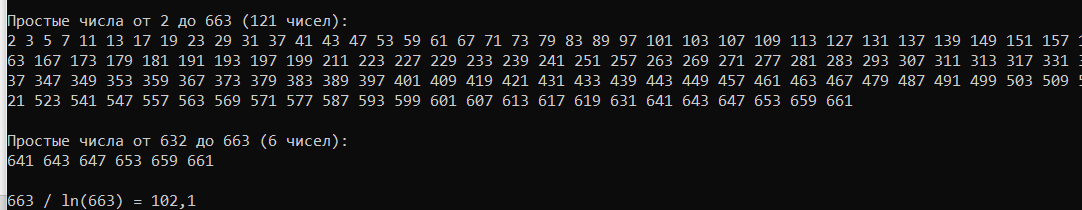


Рисунок 2.2 – Вывод функции поиска простых чисел

# 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.

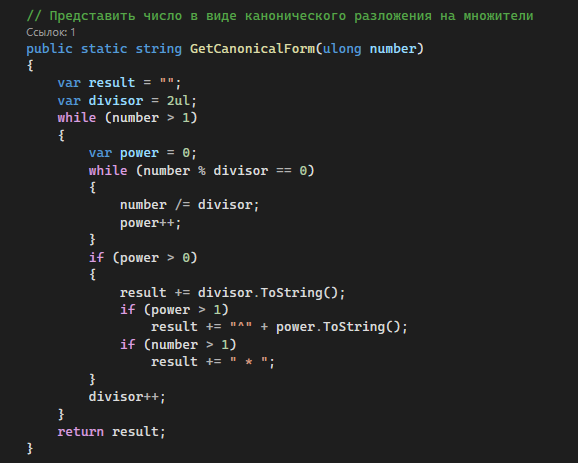


Рисунок 3.1 – Функция представления числа в канонической форме

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.

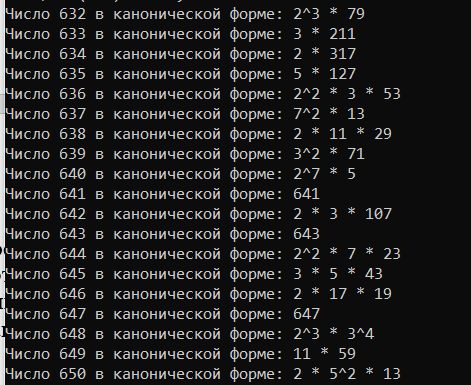


Рисунок 3.2 – Вывод канонической формы числа

# 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.

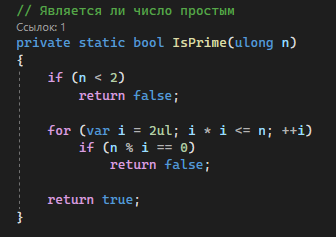


Рисунок 4.1 – Функция проверки числа на простоту

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.

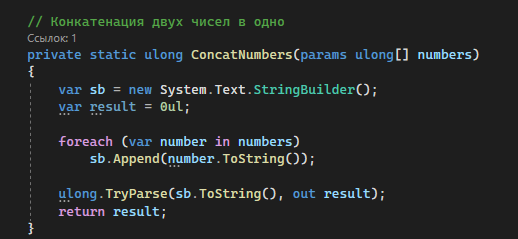


Рисунок 4.2 – Функция конкатенации двух чисел

Далее, необходимо вызвать метод проверки числа на простоту, передав в параметры возвращаемое значение функции конкатенации чисел. Вывод данной функции для чисел *m* = 632 и *n* = 663 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.