Отчет по лабораторной работе №4

Дисциплина: Математические основы защиты информации и информационной безопасности

Живцова Анна

Содержание

# 1 Цель работы

Изучить алгоритмы вычисления наибольшего общего делителя. Реализовать их.

# 2 Задание

Реализовать поиск наибольшего общего делителя с помощью алгоритма Евклида, бинарного алгоритма Евклида, расширенного алгоритма Евклида и бинарного расширенного алгоритма Евклида.

# 3 Теоретическое введение

Наибольший общий делитель чисел, позволяющий, например, определять взаимно простые числа, используется в криптографии. Подробнее в источниках [1,2].

Наибольшим общим делителем (НОД) двух натуральных чисел называют такое наибольшее натуральное число, на которое нацело делятся два данных числа. Если наибольший общий делитель двух натуральных чисел равен , то такие числа называют взаимно простыми.

Интересно, что НОД чисел можно выразить в виде их линейной комбинации с целыми коэффициентами.

# 4 Выполнение лабораторной работы

Далее везде решаем задачу поиска НОД между числами

## 4.1 Алгоритм Евклида

Для реализации алгоритма Евклида на языке Python была написанна следующая функция.

def alg1(a, b):  
 while b != 0:  
 a = a%b  
 a, b = b, a  
 return a

## 4.2 Бинарный алгоритм Евклида

Реализован с помощью функции

def alg2(a, b):  
 a, b = max(a,b), min(a,b)  
 if b == 0:  
 return a  
 if b == 1:  
 return 1  
 if b == a:  
 return b  
 g = 1  
 while (a%2 + b%2) == 0:  
 a /= 2  
 b /= 2  
 g \*= 2  
 while a%2 == 0:  
 a /= 2  
 while b%2 == 0:  
 b /= 2  
 return g\*alg2((a-b)/2, b) if a >= b else g\*alg2(a, (b-a)/2)

## 4.3 Расширенный алгоритм Евклида

Алгоритм называется расширенным так как возвращает дополнительно числа и такие что НОД(a, b) = Алгоритм реализован с помощью функции

def alg3(a, b):  
 x0 = 1; x1 = 0; y0 = 0; y1 = 1;  
 while b != 0:  
 x0 -= (a//b)\*x1  
 y0 -= (a//b)\*y1  
 a = a%b  
 a, b = b, a  
 x0, x1 = x1, x0  
 y0, y1 = y1, y0  
 return a, x0, y0

## 4.4 Бинарный расширенный алгоритм Евклида

В виде, предложенном в методических рекомендациях, реализован с помощью функции

def alg4(a, b):  
 g = 1  
 while a%2 + b%2 == 0:  
 a /= 2  
 b /= 2  
 g \*= 2  
 u = a; v = b; A = 1; B = 0; C = 0; D = 1;  
 while u%2 == 0:  
 u /= 2  
 if A%2 + B%2 == 0:  
 A /=2   
 B /=2  
 else:  
 A = (A + b)/2  
 B = (B - a)/2  
 while v%2 == 0:  
 v /= 2  
 if C%2 + D%2 == 0:  
 C /=2   
 D /=2  
 else:  
 C = (C + b)/2  
 D = (D - a)/2  
 if u <= v:  
 v = v - u   
 C = C - A  
 D = D - B  
 return g\*v, C, D

## 4.5 Проверка написанных алгоритмов

Функциональность данных функций была протестирована в среде jupyter notebook (см. рис. 1).

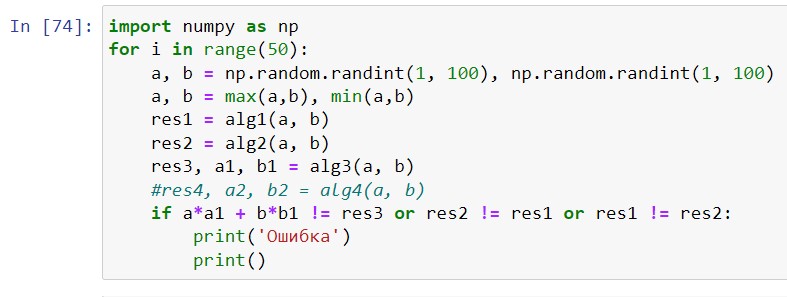


Рис. 1: Тестирование алгоритмов нахождения НОД

# 5 Выводы

В данной работе я изучила алгоритмы вычисления НОД. Реализовала поиск НОД с помощью алгоритма Евклида, бинарного алгоритма Евклида, расширенного алгоритма Евклида и бинарного расширенного алгоритма Евклида. Проверила правильность написанных программ.

# Список литературы

1. Kulyabov D., Korolkova A., Gevorkyan M. Информационная безопасность компьютерных сетей: лабораторные работы. 2015.

2. Самуйлов К.Е. и др. Сети и телекоммуникации : Учебник и практикум. Издательство Юрайт, 2019. С. 1–363.