Математические основы защиты информации и информационной безопасности. Лабораторная работа №4

Вычисление наибольшего общего делителя

Студент: Лесков Данила Валерьевич НФИмд-02-21  
Преподаватель: Кулябов Дмитрий Сергеевич

Содержание

[1 Цель работы 1](#_Toc89518920)

[2 Задание 1](#_Toc89518921)

[3 Теоретическое введение 1](#_Toc89518922)

[3.1 Алгоритм Евклида 2](#_Toc89518923)

[3.2 Бинарный алгоритм Евклида 2](#_Toc89518924)

[3.3 Расширенный алгоритм Евклида 2](#_Toc89518925)

[3.4 Расширенный бинарный алгоритм Евклида 2](#_Toc89518926)

[4 Выполнение лабораторной работы 2](#_Toc89518927)

[4.1 Описание реализации алгоритмов 2](#_Toc89518928)

[4.2 Листинг 2](#_Toc89518929)

[4.3 Полученные результаты 6](#_Toc89518930)

[5 Выводы 8](#_Toc89518931)

[Список литературы 8](#_Toc89518932)

# 1 Цель работы

Ознакомиться с алгоритмами вычисления наибольшего общего делителя.

# 2 Задание

Реализовать четыре алгоритма вычисления НОД: 1. Алгоритм Евклида; 2. Бинарный алгоритм Евклида; 3. Расширенный алгоритм Евклида; 4. Расширенный бинарный алгоритм Евклида.

# 3 Теоретическое введение

Наибольшим общим делителем (НОД) для двух целых чисел a и b называется наибольший из их общих делителей. Наибольший общий делитель существует и однозначно определён, если хотя бы одно из чисел a или b не равно нулю.

## 3.1 Алгоритм Евклида

Для вычисления наибольшего общего делителя двух целых чисел применяется способ повторного деления с остатком, называемый алгоритмом Евклида.[1]

## 3.2 Бинарный алгоритм Евклида

Бинарный алгоритм Евклида является более быстрым при реализации на компьютере, поскольку использует двоичное представление чисел a и b.[2]

## 3.3 Расширенный алгоритм Евклида

Расширенный алгоритм Евклида находит наибольший общий делитель d чисел а и b и его линейное представление, т. е. целые числа x и у, для которых .[3]

## 3.4 Расширенный бинарный алгоритм Евклида

Расширенный бинарный алгоритм Евклида так же, как и предыдущий алгоритм, позволяет найти наибольший общий делитель d чисел а и b и его линейное представление, но при том используется двоичное представление чисел a и b.[4]

# 4 Выполнение лабораторной работы

В рамках данной лабораторной работы были программно описаны 4 алгоритма нахождения наибольшего общего делителя.

## 4.1 Описание реализации алгоритмов

В данной работе были описаны 4 метода для нахождения наибольшего общего делителя. Каждый из методов принимает на вход два целых положительных числа a и b, причем a не должно быть меньше b. В результате отработки каждый из методов возвращает наибольший общий делитель этих двух целых чисел, а расширенные версии этих методов дополнительно возвращают x и y коэффициенты такие, что выполняется следующее равенство:

где d - наибольший общий делитель чисел a и b.

## 4.2 Листинг

Код приведенной ниже программы реализован на языке python.

def euclid(a, b):  
 r = []  
 r.append(a)  
 r.append(b)  
 i = 1  
 while True:  
 r.append(r[i - 1] % r[i])  
 if r[i + 1] == 0:  
 d = r[i]  
 return d  
 else:  
 i = i + 1  
  
  
def binary\_euclid(a, b):  
 g = 1  
 while a % 2 == 0 and b % 2 == 0:  
 a = a / 2  
 b = b / 2  
 g = 2 \* g  
 u = a  
 v = b  
 while u != 0:  
 while u % 2 == 0:  
 u = u / 2  
 while v % 2 == 0:  
 v = v / 2  
 if u >= v:  
 u = u - v  
 else:  
 v = v - u  
 d = g \* v  
 return d  
  
  
def extended\_euclid(a, b):  
 r = []  
 x = []  
 y = []  
  
 r.append(a)  
 r.append(b)  
  
 x.append(1)  
 x.append(0)  
  
 y.append(0)  
 y.append(1)  
  
 while r[1] != 0:  
 q = r[0] // r[1]  
 r[0], r[1] = r[1], r[0] - (r[1] \* q)  
 x[0], x[1] = x[1], x[0] - (x[1] \* q)  
 y[0], y[1] = y[1], y[0] - (y[1] \* q)  
  
 d, x, y = r[0], x[0], y[0]  
  
 return d, x, y  
  
  
def binary\_extended\_euclid(a, b):  
 g = 1  
  
 while a % 2 == 0 and b % 2 == 0:  
 a = a / 2  
 b = b / 2  
 g = 2 \* g  
  
 u = a  
 v = b  
  
 A = 1  
 B = 0  
 C = 0  
 D = 1  
  
 while u != 0:  
  
 while u % 2 == 0:  
  
 u = u / 2  
  
 if A % 2 == 0 and B % 2 == 0:  
 A = A / 2  
 B = B / 2  
 else:  
 A = (A + B) / 2  
 B = (B - A) / 2  
  
 while v % 2 == 0:  
 v = v / 2  
  
 if C % 2 == 0 and D % 2 == 0:  
 C = C / 2  
 D = D / 2  
 else:  
 C = (C + B) / 2  
 D = (D - A) / 2  
  
 if u >= v:  
 u = u - v  
 A = A - C  
 B = B - D  
 else:  
 v = v - u  
 C = C - A  
 D = D - B  
  
 d = g \* v  
 x = C  
 y = D  
  
 return d, x, y  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 while True:  
 try:  
 result\_code = int(input(  
 """  
Выберите алгоритм нахождения НОД:  
 1 - Алгоритм Евклида;  
 2 - Бинарный алгоритм Евклида;  
 3 - Расширенный алгоритм Евклида;  
 4 - Расширенный бинарный алгоритм Евклида;  
 -------------------------  
 0 - Выход из программы  
Введите номер операции: """  
 ))  
 if result\_code > 4:  
 print("Ошибка ввода!")  
 continue  
 if result\_code == 0:  
 break  
 except:  
 print("Ошибка ввода!")  
 continue  
  
 first = int(input("Введите первое число: "))  
 second = int(input("Введите второе число: "))  
 if first < second:  
 first, second = second, first  
 print(  
 """  
Ваши числа:  
 a = {}  
 b = {}  
""".format(first, second))  
  
 if result\_code == 1:  
 gcd = euclid(first, second)  
 print("НОД для {} и {} = {}".format(first, second, gcd))  
  
 if result\_code == 2:  
 gcd = binary\_euclid(first, second)  
 print("НОД для {} и {} = {}".format(first, second, gcd))  
  
 if result\_code == 3:  
 gcd, x, y = extended\_euclid(first, second)  
 print("НОД для {} и {} = {}\nx = {}\ny = {}\n\n{}\*{} + {}\*{} = {}"  
 .format(first, second, gcd, x, y, first, x, second, y, gcd))  
  
 if result\_code == 4:  
 gcd, x, y = binwary\_extended\_euclid(first, second)  
 print("НОД для {} и {} = {}\nx = {}\ny = {}\n\n{}\*{} + {}\*{} = {}"  
 .format(first, second, gcd, x, y, first, x, second, y, gcd))

## 4.3 Полученные результаты

При запуске программы пользователю предлагается взаимодействие через диалог с консольным меню. Пользователю в бесконечном цикле предлагается выбрать один из четырех реализованных методов нахождения НОД двух чисел, при этом на вход разрешается вводить только целые неотрицательные числа. На рис. 1 представлен алгоритм работы алгоритма Евклида:

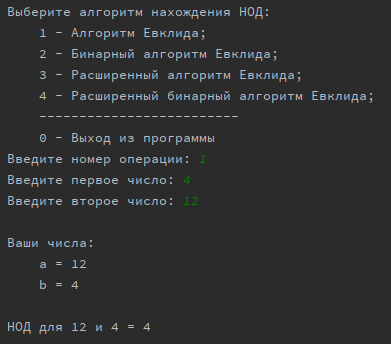


Figure 1: Алгоритм Евклида

Далее на рис. 2 представлена работа бинарного алгоритма Евклида:

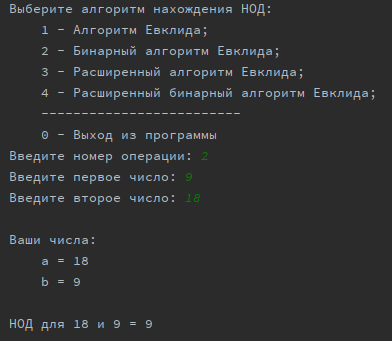


Figure 2: Бинарный алгоритм Евклида

На рис. 3 демонстрируется работа расширенного алгоритма Евклида:

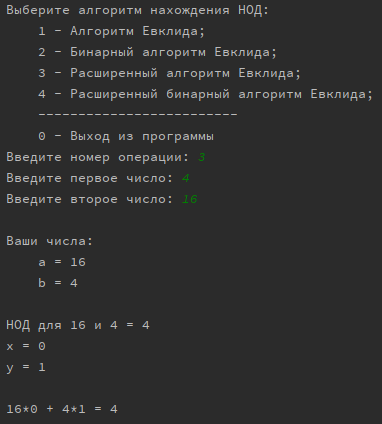


Figure 3: Расширенный алгоритм Евклида

На последнем рис. 4 отображено взаимодействие пользователя с расширенным бинарным алгоритмом Евклида:

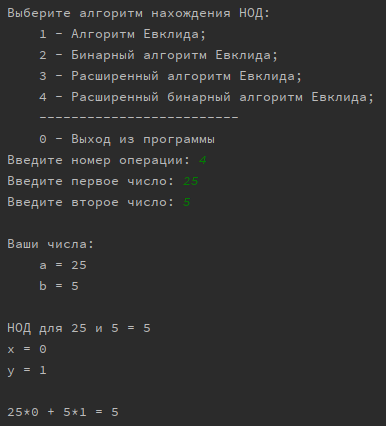


Figure 4: Расширенный бинарный алгоритм Евклида

# 5 Выводы

В ходе выполнения данной лабораторной работы было выполнено ознакомление с различными методами нахождения наибольшего общего делителя.  
В результате проделанной работы были программно реализованы следующие методы нахождение НОД: алгоритм Евклида, бинарный алгоритм Евклида, расширенный алгоритм Евклида и расширенный бинарный алгоритм Евклида.  
В итоге поставленные цели и задачи были успешно достигнуты.

# Список литературы

1. Алгоритм Евклида [Электронный ресурс]. Википедия, 2021. URL: <https://ru.wikipedia.org/wiki/Алгоритм_Евклида>.

2. Бинарный алгоритм вычисления НОД [Электронный ресурс]. Википедия, 2021. URL: <https://ru.wikipedia.org/Бинарный_алгоритм_вычисления_НОД>.

3. Расширенный алгоритм Евклида [Электронный ресурс]. e-maxx, 2012. URL: <http://e-maxx.ru/algo/export_extended_euclid_algorithm>.

4. Вычисление наибольшего общего делителя. Алгоритм Евклида [Электронный ресурс]. DOCPLAYER, 2017. URL: <https://docplayer.com/47145540-Lekciya-2-vychislenie-naibolshego-obshchego-delitelya-algoritm-evklida.html>.