Отчёт по лабораторной работе №4  
Вычисление наибольшего общего делителя

Студент: Леонова Алина Дмитриевна, 1032212306

Группа: НФИмд-01-21

Преподаватель: Кулябов Дмитрий Сергеевич,

д-р.ф.-м.н., проф.

Москва 2021

Содержание

# 1 Цель работы

Целью данной работы является ознакомление с четырьмя алгоритмами вычисления наибольшего общего делителя и их реализация на выбранном языке программирования.

# 2 Задание

Реализовать программно алгоритмы вычисления наибольшего общего делителя:

* алгоритм Евклида
* бинарный алгоритм Евклида
* расширенный алгоритм Евклида
* расширенный бинарный алгоритм Евклида

# 3 Теоретическое введение

Для вычислния наибольшего общего делителя двух целых чисел применяется способ повторного деления с остатком, называемый алгоритмом Евклида.

## 3.1 Алгоритм Евклида

В самом простом случае алгоритм Евклида применяется к паре положительных целых чисел и формирует новую пару, которая состоит из меньшего числа и разницы между большим и меньшим числом. Процесс повторяется, пока числа не станут равными. Найденное число и есть наибольший общий делитель исходной пары [1].

Алгоритм Евклида для нахождения НОД(A, B) выглядит следующим образом:

* Если A = 0, тогда НОД(A, B) = B, поскольку НОД(0, B) = B, и алгоритм останавливается.
* Если B = 0, тогда НОД(A, B) = A, поскольку GCD(A, 0) = A, и алгоритм останавливается.
* Делим A на B с остатком (A = B\*Q + R)
* Находим НОД(B, R) при помощи алгоритма Евклида, поскольку НОД(A ,B) = НОД(B, R) [2].

## 3.2 Бинарный алгоритм Евклида

Данный алгоритм “быстрее” обычного алгоритма Евклида, т.к. вместо медленных операций деления и умножения используются сдвиги [3].

Он основан на использовании следующих свойств НОД:

* НОД(2m, 2n) = 2 НОД(m, n),
* НОД(2m, 2n+1) = НОД(m, 2n+1),
* НОД(-m, n) = НОД(m, n)

Алгоритм:

* НОД(0, n) = n; НОД(m, 0) = m;НОД(m, m) = m;
* НОД(1, n) = 1; НОД(m, 1) = 1;
* Если m, n чётные, то НОД(m, n) = 2\*НОД(m/2, n/2);
* Если m чётное, n нечётное, то НОД(m, n) = НОД(m/2, n);
* Если n чётное, m нечётное, то НОД(m, n) = НОД(m, n/2);
* Если m, n нечётные и n > m, то НОД(m, n) = НОД((n-m)/2, m);
* Если m, n нечётные и n < m, то НОД(m, n) = НОД((m-n)/2, n);

## 3.3 Расширенный алгоритм Евклида

Алгоритм Евклида можно расширить для нахождения по заданным a и b таких целых x и y, что ax + by = d, где d – наибольший общий делитель a и b [4].

Лемма. Пусть для положительных целых чисел a и b (a > b) известны d = НОД(a, b) = НОД(b, a mod b), а также числа x’ и y’, для которых

Тогда значения и , являющиеся решениями уравнения , находятся из соотношений

Через здесь обозначена целая часть числа a/b.

# 4 Выполнение лабораторной работы

## 4.1 Функция для проверки

Функция для проверки функций вычисления НОД(a,b) на 6 парах целых чисел:

# Функция для проверки разных реализаций вычисления НОД(a,b)  
def check(nod\_func):  
 print(nod\_func(0, 105))  
 print(nod\_func(1, 105))  
 print(nod\_func(91, 105))  
 print(nod\_func(100000, 100))  
 print(nod\_func(12345, 678))  
 print(nod\_func(12345, 24690))

## 4.2 Алгоритм Евклида

Первым делом проверяем входные числа на раверство 0 и 1, а также проверяем, что $ a > b $ (эти действия будут повторяться во всех четырёх алгоритмах).

Далее рекурсивно вызываю эту же функцию, уменьшая входные параметры:

# 1. Алгоритм Евклида  
def nod1(a, b):  
 if a == 0 or b == 0:  
 return max(a, b)  
 if a == 1 or b == 1:  
 return 1  
 if a < b:  
 a, b = b, a  
  
 d = nod1(a % b, b)  
 return d

## 4.3 Бинарный алгоритм Евклида

Ускоряю уменьшение входных значений в случае, если они чётные:

# 2. Бинарный алгоритм Евклида  
def nod2(a, b):  
 if a == 0 or b == 0:  
 return max(a, b)  
 if a == 1 or b == 1:  
 return 1  
 if a < b:  
 a, b = b, a  
  
 g = 1  
 if a % 2 == 0 and b % 2 == 0:  
 a /= 2  
 b /= 2  
 g \*= 2  
   
 d = int( g \* nod2(a - b, b) )  
 return d

## 4.4 Расширенный алгоритм Евклида

Использую линейное представление наибольшего общего делителя:

# 3. Расширенный алгоритм Евклида  
# d = НОД(a,b) = ax + by  
def nod3(a, b):  
 if a == 0 or b == 0:  
 return max(a, b)  
 if a == 1 or b == 1:  
 return 1  
 if a < b:  
 a, b = b, a  
  
 x, y = [1,0], [0,1]  
 a\_, b\_ = a, b  
   
 while b\_ != 0:  
 a\_, b\_, p = b\_, a\_ % b\_, a\_ // b\_  
   
 if b\_ != 0:  
 x[0], x[1] = x[1], x[0] - p\*x[1]  
 y[0], y[1] = y[1], y[0] - p\*y[1]  
   
 d = a\_  
 print(a,'\*',x[1],' + ',b,'\*',y[1],' = ',d)  
 return d

## 4.5 Расширенный бинарный алгоритм Евклида

Комбинирую предыдущие подходы:

# 4. Расширенный бинарный алгоритм Евклида  
def nod4(a, b):  
 if a == 0 or b == 0:  
 return max(a, b)  
 if a == 1 or b == 1:  
 return 1  
 if a < b:  
 a, b = b, a  
  
 g = 1  
 while a % 2 == 0 and b % 2 == 0:  
 a /= 2  
 b /= 2  
 g \*= 2  
   
 a, b, g = int(a), int(b), int(g)  
 u, v, A, B, C, D = a, b, 1 , 0, 0, 1  
   
 while u != 0:  
 while u % 2 == 0:  
 u /= 2  
 if A % 2 == 0 and 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 == 0 and D % 2 == 0:  
 C /= 2  
 D /= 2  
 else:  
 C = (C + b) / 2  
 D = (D - a) / 2  
   
 if u >= v:  
 u, A, B = u-v, A-C, B-D  
 else:  
 v, C, D = v-u, C-A, D-B  
   
 A, B, C, D = int(A), int(B), int(C), int(D)   
 d = int( g \* v )  
 print(a,'\*',C,' + ',b,'\*',D,' = ',d)  
 return d

Вызов проверок работы всех реализованных функций на шести разных вариантах входных параметров, задаваемых в функции check:

print('Алгоритм Евклида')  
check(nod1)  
  
print('\nБинарный алгоритм Евклида')  
check(nod2)  
  
print('\nРасширенный алгоритм Евклида')  
check(nod3)  
  
print('\nРасширенный бинарный алгоритм Евклида')  
check(nod4)

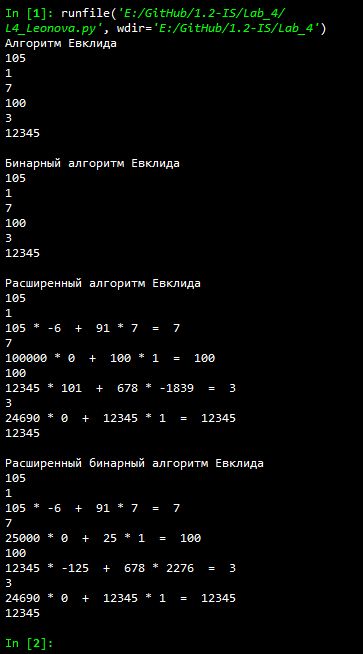


Figure 1: Результат выполнения L3\_Leonova.py

Результат выполнения программы, проверка реализации 4-ех вариаций алгоритма Евклида, находящих наибольший общий делитель шести разных пар целых чисел (см. рис. 1).

# 5 Выводы

Цель лабораторной работы была достигнута, четыре алгоритмама вычисления наибольшего общего делителя были реализованы на языке программирования Python.

# Список литературы

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

2. Алгоритм Евклида [Электронный ресурс]. khanacademy, 2021. URL: <https://ru.khanacademy.org/computing/computer-science/cryptography/modarithmetic/a/the-euclidean-algorithm>.

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

4. Расширенный алгоритм Евклида [Электронный ресурс]. eolymp, 2021. URL: <https://www.eolymp.com/ru/blogs/posts/18>.