Отчёт по лабораторной работе №4

Алгоритм Евклида

Мохаммад хоссейн фарзанфар

Содержание

# Цель работы

Изучение алгоритма Евклида нахождения НОД и его вариаций.

# Теоретические сведения

## Наибольший общий делитель

Наибольший общий делитель (НОД) – это число, которое делит без остатка два числа и делится само без остатка на любой другой делитель данных двух чисел. Проще говоря, это самое большое число, на которое можно без остатка разделить два числа, для которых ищется НОД.

## Алгоритм Евклида

При работе с большими составными числами их разложение на простые множители, как правило, неизвестно. Но для многих прикладных задач теории чисел поиск разложения числа на множители является важной, часто встречающейся практической задачей. В теории чисел существует сравнительно быстрый способ вычисления НОД двух чисел, который называется алгоритмом Евклида.

Алгоритм Евклида

* Вход. Целые числа .
* Выход. НОД.

1. Положить , , .
2. Найти остаток от деления на .
3. Если , то положить . В противном случае положить и вернуться на шаг 2.
4. Результат: .

Пример: Найти НОД для 30 и 18.

30 / 18 = 1 (остаток 12)

18 / 12 = 1 (остаток 6)

12 / 6 = 2 (остаток 0)

Конец: НОД – это делитель 6.

## Бинарный алгоритм Евклида

Бинарный алгоритм Евклида вычисления НОД оказывается более быстрым при реализации этого алгоритма на компьютере, поскольку использует двоичное представление чисел а и b. Бинарный алгоритм Евклида основан на следующих свойствах наибольшего общего делителя (считаем, что 0 < b ≤ а):

* Вход. Целые числа .
* Выход. HOД.

1. Положить .
2. Пока оба числа и четные, выполнять до получения хотя бы одного нечетного значения или .
3. Положить .
4. Пока , выполнять следующие действия.
   * Пока четное, полагать .
   * Пока четное, полагать .
   * При положить . В противном случае положить .
5. Положить .
6. Результат:

## 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 ах + by = d, и не требует «возврата», как в рассмотренном примере. Пусть d – НОД для a и b, т. е. d = (a, b), где a > b. Тогда существуют такие целые числа x и y, что d = ax +by. Иными словам, НОД двух чисел можно представить в виде линейной комбинации этих чисел с целыми коэффициентами

* Вход. Целые числа .
* Выход: НОД; такие целые числа , что .

1. Положить
2. Разделить с остатком на :
3. Если , то положить , , . В противном случае положить , , и вернуться на шаг 2.
4. Результат: .

# Выполнение работы

## Реализация алгоритмов на языке Python

``` # функция уменьшает число до тех пор пока одно из них не станет нулем # практически для этого используется цикл def evklidsimply(a, b): while a != 0 and b != 0: if a >= b: a %= b else: b %= a return a or b

# функция расширенного евклида

# ax + by = gcd(a,b)

# алгоритм находит нод и его линейное представление

def evklid\_extended(a, b): if a == 0: return (b, 0, 1) else: div, x, y = evklid\_extended(b % a, a) return (div, y - (b // a) \* x, x)

# функция бинарного евклида

def binary\_evklid(a, b): g = 1 # переменная для подсчета # согласно условиям и пунктам задачи мы все делаем # по пунктам while (a % 2 == 0 and b % 2 == 0): a //= 2 b //= 2 g *= 2 u, v = a, b while u != 0: if u % 2 == 0: u //= 2 elif v % 2 == 0: v //= 2 elif u >= v: u -= v else: v -= u d = g*  v return d

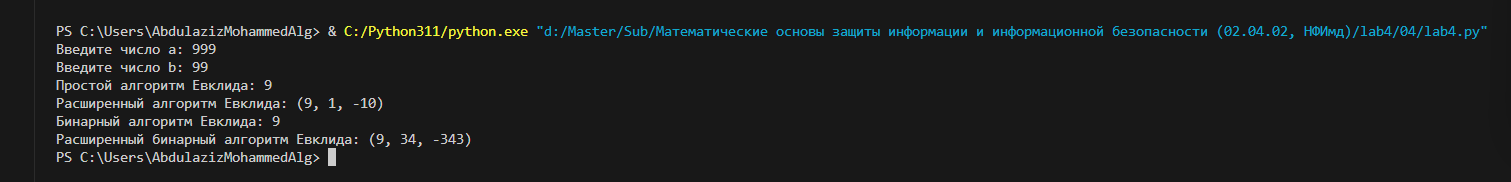
# функция расширенного бинарного евклида

def evklid\_binary\_extended(a, b): g = 1 # переменная для подсчетов # выполняем все согласно алгоритму # объяснять даже не надо все по пунктам расписано в условии задачи while (a % 2 == 0 and b % 2 == 0): a //= 2 b //= 2 g *= 2 u, v = a, b A, B, C, D = 1, 0, 0, 1 while u != 0: if 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 elif 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 elif u >= v: u -= v A -= C B -= D else: v -= u C -= A D -= B d = g*  v x = C y = D return (d, x, y)

def main(): # положим числа в переменные a = int(input(“Введите число a:”)) b = int(input(“Введите число b:”)) if a >= 0 and 0 <= b <= a: # проверяем условия что все в порядке (согласно условиям задачи) print(“Вызываем функцию Евклида”) print(evklidsimply(a, b)) # вызываем функцию простого евклида print(“А теперь можно вызвать функцию расширенного”) print(evklid\_extended(a, b)) # вызываем функцию расширенного евклида print(“А теперь функция бинарного Евклида”) print(binary\_evklid(a, b)) # вызываем функцию бинарного евклида print(“А теперь функция расширенного бинарного Евклида”) print(evklid\_binary\_extended(a, b)) # вызываем функцию расширенного бинарного евклида

if **name** == “**main**”: main()

## Контрольный пример



Работа алгоритмов

# Выводы

Изучили алгоритм Евклида нахождения НОД.

# Список литературы

1. [ВЫЧИСЛЕНИЕ НАИБОЛЬШЕГО ОБЩЕГО ДЕЛИТЕЛЯ](http://ikit.edu.sfu-kras.ru/files/15/l2.pdf)
2. [В очередной раз о НОД](https://habr.com/ru/post/464949/)