Отчет по лабораторной работе №4

Вычисление наибольшего общего делителя

Бурдина Ксения Павловна

23 октября 2023

Содержание

# 1 Цель работы

Целью данной работы является освоение алгоритмов вычисления наибольшего общего делителя.

# 2 Задание

1. Изучить методы вычисления наибольшего общего делителя.
2. Реализовать алгоритмы вычисления НОД.

# 3 Теоретическое введение

Пусть числа и целые и . Разделить на с остатком - значит представить в виде , где и . Число называется неполным частным, число - неполным остатком от деления на .

Целое число называется *наибольшим общим делителем* целых чисел (обозначается ), если выполняются следующие условия:

1. Каждое из чисел делится на ;
2. Если - другой общий делитель чисел , то делится на .

Например, , , .

Ненулевые целые числа и называются *ассоциированными* (обозначается ), если делится на и делится на [[1]](https://esystem.rudn.ru/pluginfile.php/2089869/mod_folder/content/0/%D0%A2%D1%80%D0%B0%D0%B4%D0%B8%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5%20%D1%88%D0%B8%D1%84%D1%80%D1%8B%20%D1%81%20%D1%81%D0%B8%D0%BC%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D1%8B%D0%BC%20%D0%BA%D0%BB%D1%8E%D1%87%D0%BE%D0%BC.pdf?forcedownload=1).

Для любых целых чисел существует наибольший общий делитель и его можно представить в виде *линейной комбинации* этих чисел:

Например, НОД чисел 91, 105, 154 равен 7. В качестве линейного представления можно взять:

либо

Целые числа называются *взаимно простыми в совокупности*, если . Целые числа и называются *взаимно простыми*, если .

Целые числа называются *попарно взаимно простыми*, если для всех .

# 4 Ход выполнения лабораторной работы

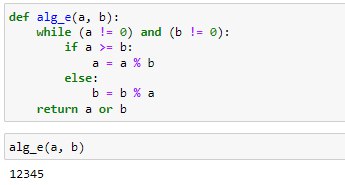
Для реализации алгоритмов вычисления наибольшего общего делителя будем использовать среду JupyterLab. Выполним необходимую задачу.

1. Зададим данные, с которыми будем работать:

Задание данных

Задание данных

1. Реализуем алгоритм Евклида с помощью следующей функции:



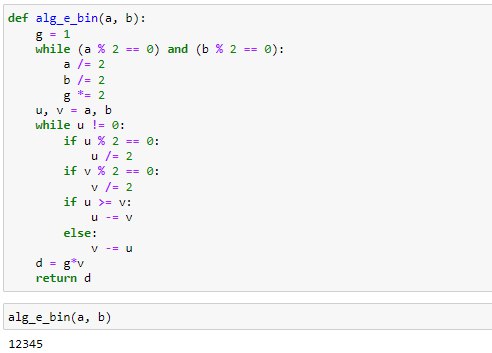
Алгоритм Евклида

Здесь на вход поступают целые числа ; . Необходимо выполнить следующее [[2]](https://esystem.rudn.ru/pluginfile.php/2089883/mod_folder/content/0/mathsec_lection07-data-encryption-standard.pdf?forcedownload=1):

* положить
* найти остаток от деления на
* если , то положить . В противном случае положить и вернуться на шаг 2
* результат:

По итогу при вызове функции мы получим результат .

1. Реализуем бинарный алгоритм Евклида с помощью следующей функции:



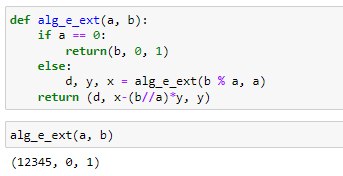
Бинарный алгоритм Евклида

Здесь на вход поступают целые числа ; . Необходимо выполнить следующее:

* положить
* пока оба числа и четные, выполнять до получения хотя бы одного нечетного значения или
* положить
* пока выполнять следующие действия:
  + пока четное, полагать
  + пока четное, полагать
  + при положить . В противном случае положить
* положить
* результат:

По итогу при вызове функции мы получим результат .

1. Реализуем расширенный алгоритм Евклида с помощью следующей функции:



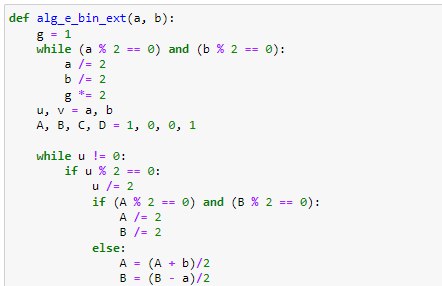
Расширенный алгоритм Евклида

Здесь на вход поступают целые числа ; . Необходимо выполнить следующее:

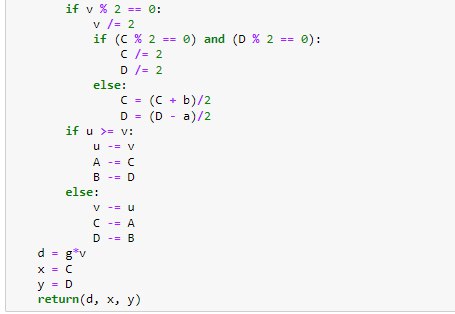
* положить
* разделить с остатком на :
* если , то положить . В противном случае положить и вернуться на шаг 2
* результат:

По итогу при вызове функции мы получим результат ; такие целые числа , что .

1. Реализуем расширенный бинарный алгоритм Евклида с помощью следующей функции:



Расширенный бинарный алгоритм Евклида 1



Расширенный бинарный алгоритм Евклида 2

Здесь на вход поступают целые числа ; . Необходимо выполнить следующее:

* положить
* пока оба числа и четные, выполнять до получения хотя бы одного нечетного значения или
* положить
* пока выполнять следующие действия:
  + пока четное:
    - положить
    - если оба числа и четные, то положить . В противном случае положить
  + пока четное
    - положить
    - если оба числа и четные, то положить . В противном случае положить
  + при положить . В противном случае положить
* положить
* результат:

По итогу при вызове функции мы получим результат :

Расширенный бинарный алгоритм Евклида

Расширенный бинарный алгоритм Евклида

# 5 Листинг программы

a = 86415  
b = 12345  
  
def alg\_e(a, b):  
 while (a != 0) and (b != 0):  
 if a >= b:  
 a = a % b  
 else:  
 b = b % a  
 return a or b  
  
alg\_e(a, b)  
  
def alg\_e\_bin(a, b):  
 g = 1  
 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  
 if v % 2 == 0:  
 v /= 2  
 if u >= v:  
 u -= v  
 else:  
 v -= u  
 d = g\*v  
 return d  
  
alg\_e\_bin(a, b)  
  
def alg\_e\_ext(a, b):  
 if a == 0:  
 return(b, 0, 1)  
 else:  
 d, y, x = alg\_e\_ext(b % a, a)  
 return (d, x-(b//a)\*y, y)  
  
alg\_e\_ext(a, b)  
  
def alg\_e\_bin\_ext(a, b):  
 g = 1  
 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  
 if 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 -= v  
 A -= C  
 B -= D  
 else:  
 v -= u  
 C -= A  
 D -= B  
 d = g\*v  
 x = C  
 y = D  
 return(d, x, y)  
  
alg\_e\_bin\_ext(a, b)

# 6 Выводы

В ходе работы мы изучили и реализовали алгоритмы вычисления наибольшего общего делителя.

# 7 Список литературы

1. Традиционные шифры с симметричным ключом [[1]](https://esystem.rudn.ru/pluginfile.php/2089869/mod_folder/content/0/%D0%A2%D1%80%D0%B0%D0%B4%D0%B8%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5%20%D1%88%D0%B8%D1%84%D1%80%D1%8B%20%D1%81%20%D1%81%D0%B8%D0%BC%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D1%8B%D0%BC%20%D0%BA%D0%BB%D1%8E%D1%87%D0%BE%D0%BC.pdf?forcedownload=1)
2. Методические материалы курса [[2]](https://esystem.rudn.ru/pluginfile.php/2089883/mod_folder/content/0/mathsec_lection07-data-encryption-standard.pdf?forcedownload=1)