РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ
Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

Отчёт по лабораторной работе №4

Дисциплина: *Математические основы защиты информации и информационной безопасности*

Студент: Майорова О.А., НФИмд-02-21
Преподаватель: д.ф.-м.н. Кулябов Д.С.

Москва 2021

Содержание

List of Figures

List of Tables

1 Цель работы

Цель: Ознакомиться с методами вычисления наибольшего общего делителя.

2 Задание

Программно реализовать алгоритмы вычисления наибольшего общего делителя для двух чисел: алгоритм Евклида, бинарный алгоритм Евклида, расширенный алгоритм Евклида, расширенный бинарный алгоритм Евклида.

3 Теоретическое введение

Делитель натурального числа — это такое натуральное число, которое делит данное число без остатка. Если у натурального числа больше двух делителей, его называют составным. Общий делитель нескольких целых чисел — это такое число, которое может быть делителем каждого числа из указанного множества. Любое число можно разделить на 1, -1 и на само себя. Значит у любого набора целых чисел будет как минимум три общих делителя. Если общий делитель больше 0 — противоположное ему значение со знаком минус также является общим делителем. Любое число, не равное 0, имеет конечное число делителей. Наибольший общий делитель существует и однозначно определён, если хотя бы одно из чисел a или b не равно нулю. Если b — делитель целого числа a, которое не равно нулю, то модуль числа b не может быть больше модуля числа a. Наибольшим общим делителем двух чисел a и b называется наибольшее число, на которое a и b делятся без остатка. Для записи может использоваться аббревиатура НОД. Для двух чисел можно записать вот так: НОД(a, b). Пример: для чисел 54 и 24 наибольший общий делитель равен 6, у чисел 12 и 8 общим делителем будет 4. Понятие наибольшего общего делителя естественным образом обобщается на наборы из более чем двух целых чисел [1,2].

Для вычисления наибольшего общего делителя двух целых чисел применяется способ повторного деления с остатком, называемый алгоритмом Евклида. Сложность алгоритма Евклида равна $O(log^2 a)$. Корректность алгоритма исходит из следующих утверждений:

- Если числа a и b целые, и a делится на b, то $b =$ НОД(a, b);

- Для любых целых чисел a, b, c выполняется равенство НОД$(a + cb, b) =$ НОД(a, b).

Таким образом, для любых $a, b > 0$ алгоритм Евклида останавливается и выдаваемое им число d является наибольшим общим делителем чисел a и b [3].

Бинарный вариант алгоритма Евклида оказывается более быстрым при реализации на компьютере, поскольку использует двоичное представление чисел a и b. Бинарный алгоритм Евклида основан на следующих свойствах наибольшего общего делителя (считаем, что $0 < b \leq a$):

- Если оба числа a и b четные, то НОД$(a, b) = 2$НОД$(\frac{a}{2}, \frac{b}{2})$;
- Если число a нечетное, число b четное, то НОД$(a, b) =$ НОД$(a, \frac{b}{2})$;
- Если оба числа a и b нечетные, $a > b$, то НОД$(a, b) =$ НОД$(a - b, b)$;
- Если $a = b$, то НОД$(a, b) = a$.

Сложность этого алгоритма также равна $O(log^2 a)$.

Для расширенного алгоритма Евклида пусть x, y - такие целые числа, что $ax + by = d$, тогда:

1. Положить $r_0 \leftarrow a, r_1 \leftarrow b, x_0 \leftarrow 1, x_1 \leftarrow 0, y_0 \leftarrow 0, y_1 \leftarrow 1, i \leftarrow 1$.
2. Разделить с остатком r_{i-1} на r_i: $r_{i-1} = q_i r_i + r_{i+1}$.
3. Если $r_{i+1} = 0$, то положить $d \leftarrow r_i, x \leftarrow x_i, y \leftarrow y_i$. В противном случае положить $x_{i+1} \leftarrow x_{i-1} - q_i x_i, y_{i+1} \leftarrow y_{i-1} - q_i y_i, i \leftarrow i + 1$ и вернуться на шаг 2.

Результат: d, x, y.

Аналогично для расширенного бинарного алгоритма Евклида (рис. 3.1).

1. Положить $g \leftarrow 1$.

2. Пока оба числа a и b четные, выполнять $a \leftarrow \frac{a}{2}$, $b \leftarrow \frac{b}{2}$, $g \leftarrow 2g$ до получения хотя бы одного нечетного значения a или b.

3. Положить $u \leftarrow a, v \leftarrow b, A \leftarrow 1, B \leftarrow 0, C \leftarrow 0, D \leftarrow 1$.

4. Пока $u \neq 0$, выполнять следующие действия.

4.1. Пока u четное:

1. Положить $u \leftarrow \frac{u}{2}$

2. Если оба числа A и B четные, то положить $A \leftarrow \frac{A}{2}$, $B \leftarrow \frac{B}{2}$

В противном случае положить $A \leftarrow \frac{A+b}{2}$, $B \leftarrow \frac{B-a}{2}$

4.2. Пока v четное:

4.2.1. Положить $v \leftarrow \frac{v}{2}$

4.2.2. Если оба числа C и D четные, то положить $C \leftarrow \frac{C}{2}$, $D \leftarrow \frac{D}{2}$, В противном случае положить $C \leftarrow \frac{C+b}{2}$, $D \leftarrow \frac{D-a}{2}$

4.3. При $u \geq v$ положить $u \leftarrow u-v, A \leftarrow A-C, B \leftarrow B-D$. В противном случае положить $v \leftarrow v-u, C \leftarrow C-A, D \leftarrow D-B$.

5. Положить $d \leftarrow gv, x \leftarrow C, y \leftarrow D$.

6. Результат: d, x, y.

Figure 3.1: Расширенный бинарный алгоритм Евклида

Сложность этих алгоритмов также равна $O(log^2 a)$ [3].

4 Выполнение лабораторной работы

Для выполнения лабораторной работы был выбран язык Python. Далее реализуем представленные алгоритмы в виде функций в соответствии с псевдокодом из задания к лабораторной работе.

Сначала реализуем алгоритм Евклида:

```
def Euclid(a, b):
    rp = a
    rc = b
    rn = rp % rc
    d = rc
    while rn != 0:
        rn = rp % rc
        d = rc
        rp = rc
        rc = rn

    return d
```

Результатом запуска функции будет рис. 4.1.

```
1 Euclid(4269, 228)
```

```
3
```

```
1 Euclid(888888, 666)
```

```
222
```

Figure 4.1: Проверка функции алгоритма Евклида

Реализуем бинарный алгоритм Евклида:

```
def BiEuclid(a, b):
    g = 1
    while a % 2 == 0 and b % 2 == 0:
        a /= 2
        b /= 2
        g *= 2

    u = a
    v = b
    while u != 0:
        while u % 2 == 0:
            u /= 2

        while v % 2 == 0:
            v /= 2

        if u >= v:
            u = u - v
        else:
            v = v - u
```

```
    d = g*v

    return d
```

Результатом запуска функции будет рис. 4.2.

```
1 BiEuclid(4269, 228)
```

```
3.0
```

```
1 BiEuclid(888888, 666)
```

```
222.0
```

Figure 4.2: Проверка функции бинарного алгоритма Евклида

Реализуем расширенный алгоритм Евклида:

```
def ExtEuclid(a, b):
    rp = a
    rc = b
    xp, xc = 1, 0
    yp, yc = 0, 1
    rn = rp % rc
    d = rc
    while rn != 0:
        rn = rp % rc
        q = (rp - rn)/rc
        d, x, y = rc, xc, yc

        rp = rc
        rc = rn

        xc = xp - q*xc
```

```
        xp = x

        yc = yp - q*yc
        yp = y

    return d, x, y
```

Результатом запуска функции будет рис. 4.3.

```
1 ExtEuclid(4269, 228)
```

```
(3, -29.0, 543.0)
```

```
1 ExtEuclid(888888, 666)
```

```
(222, -1.0, 1335.0)
```

Figure 4.3: Проверка функции расширенного алгоритма Евклида

Реализуем расширенный бинарный алгоритм Евклида в виде функции:

```
def ExtBiEuclid(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:
        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 = u - v
            A = A - C
            B = B - D
        else:
            v = v - u
            C = C - A
            D = D - B

    d, x, y = g*v, C, D

    return d, x, y
```

Результатом запуска функции будет рис. 4.4.

```
1 ExtBiEuclid(4269, 228)
```

```
(3.0, 123.0, -2303.0)
```

```
1 ExtBiEuclid(888888, 666)
```

```
(222.0, -10.0, 13347.0)
```

Figure 4.4: Проверка функции расширенного бинарного алгоритма Евклида

Можно видеть, что полученные наибольшие общие делители для двух примеров совпали для всех четырёх функций. Таким образом, можно сказать, что нахождение НОД функциями было произведено корректно.

5 Выводы

Таким образом, была достигнута цель, поставленная в начале лабораторной работы. Было осуществлено знакомство с методами вычисления наибольшего общего делителя чисел: алгоритм Евклида, бинарный алгоритм Евклида, расширенный алгоритм Евклида, расширенный бинарный алгоритм Евклида. Также была получена реализация на языке Python рассмотренных алгоритмов для двух чисел.

Список литературы

1. Наибольший общий делитель (НОД), свойства и формулы [Электронный ресурс]. Skysmart, 2020. URL: https://skysmart.ru/articles/mathematic/naibolshij-obshchij-delitel.
2. Наибольший общий делитель [Электронный ресурс]. Wikipedia, 2021. URL: https://ru.wikipedia.org/w/index.php?title=Наибольший_общий_делитель&oldid=117797985.
3. Вычисление наибольшего общего делителя [Электронный ресурс]. StudFiles, 2015. URL: https://studfile.net/preview/3073287/page:2/.