

Отчёт по лабораторной работе 4

Вычисление наибольшего общего делителя

Каримов Зуфар Исматович НФИ-01-22

Содержание

1	Цель работы	5
2	Теоретические сведения	6
3	Выполнение лабораторной работы	8
4	Выводы	13
5	Список литературы	14

List of Tables

List of Figures

3.1	Функция для вычисления алгоритма Евклида	8
3.2	Функция для вычисления бинарного алгоритма Евклида	9
3.3	Функция для вычисления вычисления расширенного алгоритма Евклида.	10
3.4	Функция для вычисления расширенного бинарного алгоритма Евклида. Первая часть	11
3.5	Функция для вычисления расширенного бинарного алгоритма Евклида. Вторая часть	12
3.6	Результат алгоритмов	12

1 Цель работы

Реализация алгоритмов вычисления наибольшего общего делителя (Евклида).

2 Теоретические сведения

Алгоритм Евклида — это способ нахождения наибольшего общего делителя (НОД) двух целых чисел. Оригинальная версия алгоритма, когда НОД находится вычитанием, была открыта Евклидом (III в. до н. э). В настоящее время чаще при вычислении НОД алгоритмом Евклида используют деление, так как данный метод эффективнее.

Вычисление НОД делением

Наибольший общий делитель пары чисел – это самое большое число, которое нацело делит оба числа пары. Пусть требуется вычислить НОД для чисел 108 и 72. Алгоритм вычисления делением будет таковым:

1. Разделим большее число (делимое) на меньшее (делитель): $108 / 72 = 1$, остаток 36.
2. Поскольку остаток не был равен нулю, то сделаем делитель делимым, а остаток – делителем: $72 / 36 = 2$, остаток 0.
3. Когда остаток равен нулю, то делитель является искомым НОД для пары заданных чисел. То есть $\text{НОД}(108, 72) = 36$. Действительно, $108 / 36 = 3$ и $72 / 36 = 2$. [1]

Расширенный алгоритм Евклида

Расширенным алгоритм называется не из-за более высокой скорости работы или более сложной реализации, а потому что он позволяет извлекать из входных данных дополнительную информацию.

Расширенный алгоритм также находит наибольший общий делитель, а ещё он определяет два коэффициента x и y , такие что:

$ax + by = \gcd(a, b)$, где \gcd – это функция по нахождения НОД.

Иными словами, алгоритм находит наибольший делитель и его линейное представление.

\gcd – это аббревиатура, которую часто используют для обозначения функции по назначению НОД:

g – Greatest (наибольший);

c – Common (общий);

d – Divisor (делитель).

Бинарный алгоритм Евклида

Суть бинарного алгоритма точно такая же — найти наибольший делитель. От классического он отличается только способом реализации.

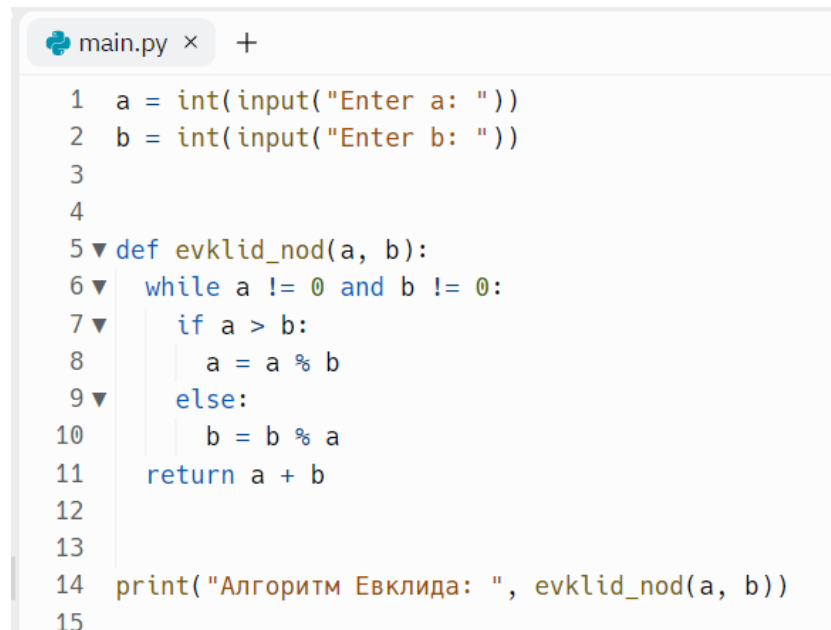
Вместо классических арифметических операций, в бинарном алгоритме Евклида используются только битовые сдвиги влево и вправо, которые соответствуют умножению и делению на 2.[2]

3 Выполнение лабораторной работы

1. Написал функцию `evklid_nod` для вычисления алгоритма Евклида. (рис. 3.1)

Алгоритм нахождения НОД делением:

1. Большее число делим на меньшее.
2. Если делится без остатка, то меньшее число и есть НОД (следует выйти из цикла).
3. Если есть остаток, то большее число заменяем на остаток от деления.
4. Переходим к пункту 1.



```
main.py × +
1 a = int(input("Enter a: "))
2 b = int(input("Enter b: "))
3
4
5 def evklid_nod(a, b):
6     while a != 0 and b != 0:
7         if a > b:
8             a = a % b
9         else:
10            b = b % a
11    return a + b
12
13
14 print("Алгоритм Евклида: ", evklid_nod(a, b))
15
```

Figure 3.1: Функция для вычисления алгоритма Евклида

2. Написал функцию `evklid_binary` для вычисления бинарного алгоритма Евклида. (рис. 3.2)

1. Сначала положим $g = 1$
2. Пока оба числа a и b четные, выполнить $a = \frac{a}{2}, b = \frac{b}{2}, g = 2g$ до получения хотя одного нечетного значения a или b .
3. Положим $u = a, v = b$
4. Пока $u \neq 0$:
 1. Пока u четное, полагать $u = \frac{u}{2}$
 2. Пока v четное, полагать $v = \frac{v}{2}$
 3. При $u \geq v$ положим $u = u - v$. В противном случае положим $v = v - u$
5. Положим $d = gv$
6. Получим результат d

```
16
17 def evklid_binary(a, b):
18     g = 1
19     while a % 2 == 0 and b % 2 == 0:
20         a = a / 2
21         b = b / 2
22         g = 2 * g
23     u = a
24     v = b
25     while u != 0:
26         if u % 2 == 0:
27             u = u / 2
28         if v % 2 == 0:
29             v = v / 2
30
31         if u >= v:
32             u = u - v
33         else:
34             v = v - u
35     d = g * v
36     return d
37
38 print("Бинарный алгоритм Евклида: ", evklid_binary(a, b))
39
```

Figure 3.2: Функция для вычисления бинарного алгоритма Евклида

3. Написал функцию `evklid_extend` для вычисления расширенного алгоритма Евклида. (рис. 3.3)

Сначала проверяется, равно ли первое число нулю, если это так, то второе число является делителем, а коэффициенты равны 0 и 1, так как « $\text{num1} * x + \text{num2} * y = y$ » в том случае, если $y = 1$, а левое произведение равно нулю.

Функция возвращает три числа: делитель, коэффициент x и коэффициент y .

```
41 def evklid_extend(a, b):
42     if a == 0:
43         return (b, 0, 1)
44     else:
45         div, x, y = evklid_extend(b % a, a)
46         return (div, y - (b // a) * x, x)
47
48 print("Расширенный алгоритм Евклида: ", evklid_extend(a, b))
49
50
```

Figure 3.3: Функция для вычисления вычисления расширенного алгоритма Евклида.

4. Написал функцию `evklid_binary_extend` для вычисления расширенного бинарного алгоритма Евклида. (рис. 3.4) (рис. 3.5)

1. Сначала положим $g = 1$
2. Пока оба числа a и b четные, выполнить $a = \frac{a}{2}, b = \frac{b}{2}, g = 2g$ до получения хотя одного нечетного значения a или b .
3. Положим $u=a, v=b, A=1, B=0, C=0, D=1$.
4. Пока $u \neq 0$:
 1. Пока u четное, полагать $u = \frac{u}{2}$
 2. Если оба числа A и B четные, полагать $A = \frac{A}{2}, B = \frac{B}{2}$. В противном случае положить $A = \frac{A+b}{2}, B = \frac{B-a}{2}$
 3. Пока v четное, полагать $v = \frac{v}{2}$
 4. Если оба числа C и D четные, полагать $C = \frac{C}{2}, D = \frac{D}{2}$. В противном случае положить $C = \frac{C+b}{2}, D = \frac{D-2}{2}$

5. При $u \geq v$ положим $u = u - v$, $A = A - C$, $B = B - D$. В противном случае положим $v = v - u$, $C = C - A$, $D = D - B$.
5. Положим $d = \gcd$, $x = C$, $y = D$
6. Получим результат d, x, y

```
main.py × +
51 ▼ def evklid_binary_extend(a, b):
52     g = 1
53 ▼ while a % 2 == 0 and b % 2 == 0:
54     a = a / 2
55     b = b / 2
56     g = 2 * g
57     u = a
58     v = b
59     A = 1
60     B = 0
61     C = 0
62     D = 1
63
64 ▼ while u != 0:
65 ▼     if u % 2 == 0:
66         u = u / 2
67 ▼     if A % 2 == 0 and B % 2 == 0:
68         A = A / 2
69         B = B / 2
70 ▼     else:
71         A = (A + b) / 2
72         B = (B - a) / 2
73 ▼     if v % 2 == 0:
74         v = v / 2
75 ▼     if C % 2 == 0 and D % 2 == 0:
76         C = C / 2
77         D = D / 2
```

Figure 3.4: Функция для вычисления расширенного бинарного алгоритма Евклида. Первая часть

```

main.py x +
70 else:
71     A = (A + b) / 2
72     B = (B - a) / 2
73 if v % 2 == 0:
74     v = v / 2
75 if C % 2 == 0 and D % 2 == 0:
76     C = C / 2
77     D = D / 2
78 else:
79     C = (C + b) / 2
80     D = (D - a) / 2
81 if u >= v:
82     u = u - v
83     A = A - C
84     B = B - D
85
86 else:
87     v = v - u
88     C = C - A
89     D = D - B
90 d = g * v
91 x = C
92 y = D
93 return d, x, y
94
95 print("Расширенный бинарный алгоритм Евклида: ", evklid_binary_extend(a, b))
96

```

Figure 3.5: Функция для вычисления расширенного бинарного алгоритма Евклида. Вторая часть

5. Получил результат (рис. 3.6)

```

>_ Console x +
Enter a: 91
Enter b: 105
Алгоритм Евклида: 7
Бинарный алгоритм Евклида: 7.0
Расширенный алгоритм Евклида: (7, 7, -6)
Расширенный бинарный алгоритм Евклида: (7.0, 52.0, -45.0)
>

```

Figure 3.6: Результат алгоритмов

4 Выводы

Реализовал алгоритм вычисления наибольшего общего делителя (Евклида).

5 Список литературы

1. Алгоритм Евклида [Электронный ресурс] - Режим доступа: <https://scienceland.info/algebra8/euclid-algorithm>
2. Бинарный алгоритм вычисления НОД [Электронный ресурс] - Режим доступа: <https://intellect.icu/binarnyj-algoritm-vychisleniya-nod-4394>