

# Алгоритмы Евклида для нахождения НОД.

---

Кейела Патачона

02 декабря, 2021, Москва, Россия

Российский Университет Дружбы Народов

## Цели и задачи

---

## Цель лабораторной работы

Изучить алгоритмы для вычисления наибольшего общего делителя.

# **Выполнение лабораторной работы**

---

# Алгоритм Евклида

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

Вход. Целые числа  $a, b$ ;  $0 < b < a$ . Выход.  $d = \text{НОД}(a, b)$ .

1. Положить  $r_0 = a, r_1 = b, i = 1$
2. Найти остаток  $r_{i+1}$  от деления  $r_{i-1}$  на  $r_i$
3. Если  $r_{i+1} = 0$ , то положить  $d = r_i$ . В противном случае положить  $i = i+1$  и вернуться на шаг 2
4. Результат:  $d$

# Бинарный алгоритм Евклида

Бинарный алгоритм Евклида является 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  $0 < b < a$ ):

1. если оба числа  $a$  и  $b$  четные, то  $\text{НОД}(a,b) = 2 \text{НОД}(a/2,b/2)$
2. если число  $a$  - нечетное, число  $b$  — четное, то  $\text{НОД}(a,b) = \text{НОД}(a, b/2)$
3. если оба числа  $a$  и  $b$  нечетные,  $a > b$ , то  $\text{НОД}(a,b) = \text{НОД}(a - b, b)$
4. если  $a = b$ , то  $\text{НОД}(a,b) = a$

# Расширенный алгоритм Евклида и Бинарный алгоритм Евклида

Данные алгоритмы Евклида находят, помимо  $g = \text{НОД}(a, b)$  такие целые коэффициенты  $x$  и  $y$ , что:

$$ax + by = d$$

Заметим, что решений бесконечно много: имея решение  $(x, y)$ , можно  $x$  увеличить на  $b$ , а  $y$  уменьшить на  $a$ , и равенство при этом не изменится.

# Контрольные примеры

```
C:\Users\patat > Desktop > Master Rush > Git_work > 2021-2022 > Cybersecurity > laboratory > lab04 > 1_task.py > ...
1 print('a >= b > 0')
2 a = int(input('Введите a: '))
3 b = int(input('Введите b: '))
4 r = [a, b]
5 while True:
6     if r[-2] % r[-1] == 0:
7         d = r[-1]
8         break
9     else:
10        r.append(r[-2] % r[-1])
11 print(f'НОД(a,b) = {d}')
12
```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ

PS C:\Users\patat> & C:\Users\patat\AppData\Local\Programs\Python\Python38\python.exe "C:\Users\patat\Desktop\Master Rush\Git\_work\2021-2022\Cybersecurity\laboratory\lab04\1\_task.py"

a >= b > 0  
Введите a: 13  
Введите b: 7  
НОД(a,b) = 1  
PS C:\Users\patat> & C:\Users\patat\AppData\Local\Programs\Python\Python38\python.exe "C:\Users\patat\Desktop\Master Rush\Git\_work\2021-2022\Cybersecurity\laboratory\lab04\1\_task.py"

a >= b > 0  
Введите a: 20  
Введите b: 15  
НОД(a,b) = 5  
PS C:\Users\patat>

Figure 1: Алгоритм Евклида

```
C:\Users\patat > Desktop > Master Rush > Git_work > 2021-2022 > Cybersecurity > laboratory > lab04 > 2_task.py > ...
1 print('a >= b > 0')
2 a = int(input('Введите a: '))
3 b = int(input('Введите b: '))
4 w = 1
5 while True:
6     if a % 2 == 0 and b % 2 == 0:
7         a *= 0.5
8         b *= 0.5
9         w *= 2
10    else:
11        break
12 u = a
13 v = b
14 while u != v:
15     while u % 2 == 0:
16         u *= 0.5
17
```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ

PS C:\Users\patat> & C:\Users\patat\AppData\Local\Programs\Python\Python38\python.exe "C:\Users\patat\Desktop\Master Rush\Git\_work\2021-2022\Cybersecurity\laboratory\lab04\2\_task.py"

a >= b > 0  
Введите a: 10  
Введите b: 12  
НОД(a,b) = 6.0  
PS C:\Users\patat> & C:\Users\patat\AppData\Local\Programs\Python\Python38\python.exe "C:\Users\patat\Desktop\Master Rush\Git\_work\2021-2022\Cybersecurity\laboratory\lab04\2\_task.py"

a >= b > 0  
Введите a: 36  
Введите b: 19  
НОД(a,b) = 1.0  
PS C:\Users\patat>

Figure 2: Бинарный алгоритм Евклида



# Контрольные примеры

```
C:\Users\patat\Desktop> Master-Ruh\git_work\2021-2022\Cybersecurity\laboratory\lab04> 3_task.py > ...
4 r = [a, b]
5 x = [1, 0]
6 y = [0, 1]
7 while True:
8     if r[-2] % r[-1] == 0:
9         d = r[-1]
10        a = r[-1]
11        b = y[-1]
12        break
13    else:
14        q, s = r[-2] // r[-1]
15        x.append(x[-2]-q,1*x[-1])
16        y.append(y[-2]-q,1*y[-1])
17        r.append(r[-2] % r[-1])
18    print(d, x, y)
19
```

ПРОБЛЕМЫ Выходные данные ТЕРМИНАЛ КОМАНДЫ ОТКАЗОВ

```
PS C:\Users\patat> & C:\Users\patat\AppData\Local\Programs\Python\Python38\python.exe "c:/Users/patat/Desktop/Master-Ruh/git_work/2021-2022/Cybersecurity/laboratory/lab04/3_task.py"
a >= b > 0
Введите a: 23
Введите b: 11
3 1 0
PS C:\Users\patat>
PS C:\Users\patat> & C:\Users\patat\AppData\Local\Programs\Python\Python38\python.exe "c:/Users/patat/Desktop/Master-Ruh/git_work/2021-2022/Cybersecurity/laboratory/lab04/3_task.py"
a >= b > 0
Введите a: 15
Введите b: 10
5 1 -1
PS C:\Users\patat>
```

Figure 3: Расширенный алгоритм Евклида

```
C:\Users\patat\Desktop> Master-Ruh\git_work\2021-2022\Cybersecurity\laboratory\lab04> 4_task.py > ...
1 20
2 print('a >= b > 0')
3 a = int(input('Введите a: '))
4 b = int(input('Введите b: '))
5 B = 1
6 while True:
7     if a % 2 == 0 and b % 2 == 0:
8         a *= 0.5
9         b *= 0.5
10        B *= 2
11    else:
12        break
13    u = a
14    v = b
15    A = 1
16    B = 0

```

ПРОБЛЕМЫ Выходные данные ТЕРМИНАЛ КОМАНДЫ ОТКАЗОВ

```
PS C:\Users\patat> & C:\Users\patat\AppData\Local\Programs\Python\Python38\python.exe "c:/Users/patat/Desktop/Master-Ruh/git_work/2021-2022/Cybersecurity/laboratory/lab04/4_task.py"
a >= b > 0
Введите a: 20
Введите b: 15
5.0 -2.0 3.0
PS C:\Users\patat> 102
PS C:\Users\patat> & C:\Users\patat\AppData\Local\Programs\Python\Python38\python.exe "c:/Users/patat/Desktop/Master-Ruh/git_work/2021-2022/Cybersecurity/laboratory/lab04/4_task.py"
a >= b > 0
Введите a: 302
Введите b: 9
3.0 -2.0 23.0
PS C:\Users\patat>
```

## Выводы

---

Мной были узчены алгоритмы для вычисления наибольшего общего делителя Евклида.