

## Лабораторная работа 4

Попов Дмитрий Павлович, НФИмд-01-23

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра математического моделирования и искусственного интеллекта

ПРЕЗЕНТАЦИЯ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

дисциплина: Математические основы защиты информации и информационной безопасности

Преподаватель: Кулябов Дмитрий Сергеевич

Студент: Попов Дмитрий Павлович

Группа: НФИмд-01-23

МОСКВА

2023 г.

# **Прагматика выполнения лабораторной работы**

# Прагматика выполнения лабораторной работы

► Требуется реализовать:

1. Алгоритм Евклида
2. Бинарный алгоритм Евклида
3. Расширенный алгоритм Евклида
4. Расширенный бинарный алгоритм Евклида

## Цель работы

# Цель работы

Освоение на практике алгоритмов вычисления НОД

## **Выполнение лабораторной работы**

# 1. Алгоритм Евклида

1. Берёт два числа  $a$  и  $b$ , где  $a > b$
2. Повторяет деление  $a$  на  $b$ , заменяя  $a$  значением  $b$  и  $b$  остатком от деления, пока  $b$  не станет равным 0.
3. Последнее ненулевое значение  $a$  будет НОД.

```
6 def euclidean_gcd(a, b):  
7     r0, r1, i = a, b, 1  
8     while True:  
9         ri_next = r0 % r1  
10        if ri_next == 0:  
11            d = r1  
12            return d  
13        r0, r1, i = r1, ri_next, i + 1
```

Figure 1: nod1



## 2. Бинарный алгоритм Евклида

1. Если оба числа четные, делит оба числа на 2 и удваивает итоговый НОД
2. Если только одно из чисел четное, делит только его на 2.
3. Из большего числа вычитает меньшее.
4. Повторяет процесс, пока числа не станут равными. Это число становится НОД, умноженным на полученный ранее множитель.

```
16 def binary_gcd(a, b):
17     g = 1
18     while a % 2 == 0 and b % 2 == 0:
19         a, b, g = a // 2, b // 2, 2 * g
20
21     u, v = a, b
22     while u != 0:
23         while u % 2 == 0:
24             u //= 2
25         while v % 2 == 0:
26             v //= 2
```

### 3. Расширенный алгоритм Евклида

1. Кроме нахождения НОД, алгоритм находит такие числа  $x$  и  $y$ , что  $ax+by=\text{НОД}(a,b)$ .
2. Начинается с базовых коэф.:  $x_0 = 1, y_0 = 0$  (для  $a$ ) и  $x_1 = 0, y_1 = 1$  (для  $b$ ).
3. При каждом шаге обновляются значения коэффициентов, используя остаток и частное от деления.

```
34 def extended_gcd(a, b):  
35     r0, r1, x0, x1, y0, y1, i = a, b, 1, 0, 0, 1, 1  
36     while True:  
37         q, ri_next = divmod(r0, r1)  
38         if ri_next == 0:  
39             return r1, x1, y1  
40         x_next = x0 - q * x1  
41         y_next = y0 - q * y1  
42         r0, r1, x0, x1, y0, y1, i = r1, ri_next, x1, x_next, y1, y_next, i + 1
```

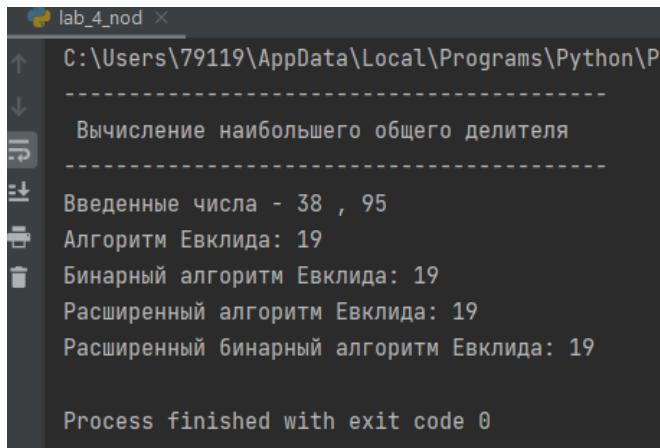
Figure 3: nod3

## 4. Расширенный бинарный алгоритм Евклида

1. Как и обычный бинарный алгоритм, но также отслеживает коэффициенты  $x$  и  $y$ .
2. Когда числа делятся на 2, коэффициенты корректируются соответствующим образом.
3. Когда из одного числа вычитается другое, соответствующие коэффициенты также вычитаются.

```
45 def extended_binary_gcd(a, b):
46     g = 1
47     while a % 2 == 0 and b % 2 == 0:
48         a, b, g = a // 2, b // 2, 2 * g
49
50     u, v, A, B, C, D = a, b, 1, 0, 0, 1
51     while u != 0:
52         while u % 2 == 0:
53             u //= 2
54             if A % 2 == 0 and B % 2 == 0:
55                 A, B = A // 2, B // 2
56             else:
57                 A, B = (A + b) // 2, (B - a) // 2
58
59         while v % 2 == 0:
60             v //= 2
61             if C % 2 == 0 and D % 2 == 0:
```

## 5. Результат работы программы



```
lab_4_nod x
C:\Users\79119\AppData\Local\Programs\Python\P
-----
Вычисление наибольшего общего делителя
-----
Введенные числа - 38 , 95
Алгоритм Евклида: 19
Бинарный алгоритм Евклида: 19
Расширенный алгоритм Евклида: 19
Расширенный бинарный алгоритм Евклида: 19

Process finished with exit code 0
```

Figure 5: out

## Вывод

## Вывод

В результате выполнения работы я освоил на практике применение алгоритмов нахождения НОД.

