Отчет по лабораторной работе №4

Вычисление наибольшего общего делителя

Бурдина Ксения Павловна

23 октября 2023

Содержание

Список иллюстраций

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 a и b целые и $b \neq 0$. Разделить a на b с остатком - значит представить a в виде $a = qb + r$, где $q, r \in Z$ и $0 \leqslant r \leqslant \mid b \mid$. Число q называется неполным частным, число r - неполным остатком от деления a на b.

Целое число $d \neq 0$ называется *наибольшим общим делителем* целых чисел $a_1, a_2, ..., a_k$ (обозначается $d = (a_1, a_2, ..., a_k)$), если выполняются следующие условия:

1. Каждое из чисел $a_1, a_2, ..., a_k$ делится на d;
2. Если $d_1 \neq 0$ - другой общий делитель чисел $a_1, a_2, ..., a_k$, то d делится на d_1.

Например, $(12345, 24690) = 12345$, $(12345, 54321) = 3$, $(12345, 12541) = 1$.

Ненулевые целые числа a и b называются *ассоциированными* (обозначается $a \sim b$), если a делится на b и b делится на a [1].

Для любых целых чисел $a_1, a_2, ..., a_k$ существует наибольший общий делитель d и его можно представить в виде *линейной комбинации* этих чисел:

$$d = c_1a_1 + c_2a_2 + ... + c_ka_k, c_i \in Z(Z -).$$

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

$$7 = 7 * 91 + (-6) * 105 + 0 * 154,$$

либо

$$7 = 4 * 91 + 1 * 105 - 3 * 154.$$

Целые числа $a_1, a_2, ..., a_k$ называются *взаимно простыми в совокупности*, если $(a_1, a_2, ..., a_k) = 1$. Целые числа a и b называются *взаимно простыми*, если $(a, b) = 1$.

Целые числа $a_1, a_2, ..., a_k$ называются *попарно взаимно простыми*, если $(a_i, a_j) = 1$ для всех $1 \leqslant i \neq j \leqslant k$.

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:

```
a = 86415
b = 12345
```

Рис. 4.1.: Задание данных

2. Реализуем алгоритм Евклида с помощью следующей функции:

```
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)
```

```
12345
```

Рис. 4.2.: Алгоритм Евклида

Здесь на вход поступают целые числа $a, b; 0 < b \leqslant a$. Необходимо выполнить следующее [2]:

- положить $r_0 \leftarrow a, r_1 \leftarrow b, i \leftarrow 1$
- найти остаток r_{i+1} от деления r_{i-1} на r_i
- если $r_{i+1} = 0$, то положить $d \leftarrow r_i$. В противном случае положить $i \leftarrow i+1$ и вернуться на шаг 2
- результат: d

По итогу при вызове функции мы получим результат $d = (a, b)$.

3. Реализуем бинарный алгоритм Евклида с помощью следующей функции:

```
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)
```

```
12345
```

Рис. 4.3.: Бинарный алгоритм Евклида

Здесь на вход поступают целые числа $a, b; 0 < b \leqslant a$. Необходимо выполнить следующее:

- положить $g \leftarrow 1$

- 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
- положить $u \leftarrow a, v \leftarrow b$
- пока $u \neq 0$ выполнять следующие действия:
  - пока u четное, полагать $u \leftarrow \frac{u}{2}$
  - пока v четное, полагать $v \leftarrow \frac{v}{2}$
  - при $u \geqslant v$ положить $u \leftarrow u - v$. В противном случае положить $v \leftarrow v - u$
- положить $d \leftarrow gv$
- результат: d

По итогу при вызове функции мы получим результат $d = (a, b)$.

4. Реализуем расширенный алгоритм Евклида с помощью следующей функции:

```
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)
```

```
(12345, 0, 1)
```

Рис. 4.4.: Расширенный алгоритм Евклида

Здесь на вход поступают целые числа $a, b; 0 < b \leqslant a$. Необходимо выполнить следующее:

- положить $r_0 \leftarrow a, r_1 \leftarrow b, x_0 \leftarrow 1, x_1 \leftarrow 0, y_0 \leftarrow 0, y_1 \leftarrow 1, i \leftarrow 1$

- разделить с остатком r_{i-1} на r_i: $r_{i-1} = q_i r_i + r_{i+1}$
- 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

По итогу при вызове функции мы получим результат $d = (a, b)$; такие целые числа x, y, что $ax + by = d$.

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

```
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
```

Рис. 4.5.: Расширенный бинарный алгоритм Евклида 1

```
        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)
```

Рис. 4.6.: Расширенный бинарный алгоритм Евклида 2

Здесь на вход поступают целые числа $a, b; 0 < b \leqslant a$. Необходимо выполнить следующее:

- положить $g \leftarrow 1$
- 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
- положить $u \leftarrow a, v \leftarrow b, A \leftarrow 1, B \leftarrow 0, C \leftarrow 0, D \leftarrow 1$
- пока $u \neq 0$ выполнять следующие действия:
  - пока u четное:
    - положить $u \leftarrow \frac{u}{2}$
    - если оба числа A и B четные, то положить $A \leftarrow \frac{A}{2}, B \leftarrow \frac{B}{2}$. В противном случае положить $A \leftarrow \frac{A+b}{2}, B \leftarrow \frac{B-a}{2}$
  - пока v четное
    - положить $v \leftarrow \frac{v}{2}$
    - если оба числа C и D четные, то положить $C \leftarrow \frac{C}{2}, D \leftarrow \frac{D}{2}$. В противном случае положить $C \leftarrow \frac{C+b}{2}, D \leftarrow \frac{D-a}{2}$

- при $u \geqslant v$ положить $u \leftarrow u - v, A \leftarrow A - C, B \leftarrow B - D$. В противном случае положить $v \leftarrow v - u, C \leftarrow C - A, D \leftarrow D - B$

- положить $d \leftarrow gv, x \leftarrow C, y \leftarrow D$
- результат: d, x, y

По итогу при вызове функции мы получим результат $d = (a, b)$:

```
alg_e_bin_ext(a, b)
```

```
(12345, 0, 1)
```

Рис. 4.7.: 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]

2. Методические материалы курса [2]