

Вычисление наибольшего общего делителя

Лабораторная работа №3

Данилова А.С.

Изучить различные вариации алгоритма Евклида и реализовать их программно на языке Julia.

Наибольший общий делитель

НОД – это наибольшее натуральное целое число, на которое эти числа делятся без остатка.

Алгоритм Евклида Нужно заменить большее из чисел на остаток от деления его на меньшее и для полученной пары повторять эту процедуру, пока одно из чисел не станет равно нулю. Тогда второе число будет равно наибольшему общему делителю исходных чисел.

Бинарный Алгоритм Евклида Данный алгоритм «быстрее» обычного алгоритма Евклида, так как вместо медленных операций деления и умножения используются сдвиги.

Расширенный Алгоритм Евклида Расширенный алгоритм возвращает не только НОД(a, b), но и коэффициенты x и y , такие что $\text{НОД} = ax + by$.

Расширенный бинарный Алгоритм Евклида Алгоритм находит наибольший делитель и его линейное представление.

```
1  function NOD(a, b)
2      while b != 0
3          n = b
4          b = a % b
5          a = n
6      end
7      return a
8  end
9
10 println(NOD(128, 80))
```

Рис. 1: Код для Алгоритма Евклида

```
1  function binar(a, b)
2      if a == b
3          return a
4      end
5      if (a & 1) == 0 && (b & 1) == 0
6          return 2 * binar(a >> 1, b >> 1)
7      elseif (a & 1) == 0
8          return binar(a >> 1, b)
9      elseif (b & 1) == 0
10         return binar(a, b >> 1)
11     else
12         if a > b
13             return binar((a - b) >> 1, b)
14         else
15             return binar(a, (b - a) >> 1)
16         end
17     end
18 end
19
20 println(binar(24, 36))
21 println(binar(128, 80))
```

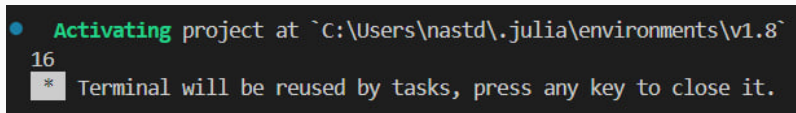
Рис. 2: Код для бинарного Алгоритма Евклида

```
1  function nod(a, b)
2      if b == 0
3          return a, 1, 0
4      end
5
6      NOD, x1, y1 = nod(b, a % b)
7      x = y1
8      y = x1 - div(a, b) * y1
9
10     return NOD, x, y
11 end
12
13 NOD(a, b) = nod(a, b)[1]
14
15 println(nod(24, 36))
16 println(nod(128, 80))
```

Рис. 3: Код для расширенного Алгоритма Евклида


```
1 function binar(a, b)
2     if a == b
3         return a, 1, -1
4     end
5
6     if (a & 1) == 0
7         if (b & 1) == 0
8             d, x, y = binar(a >> 1, b >> 1)
9             return d, x, y
10        elseif (b & 1) != 0
11            d, x, y = binar(a >> 1, b)
12            return d, 2 * x, y
13        end
14    elseif (a & 1) != 0
15        if (b & 1) == 0
16            d, x, y = binar(a, b >> 1)
17            return d, x, 2 * y
18        elseif (b & 1) != 0
19            if a > b
20                d, x, y = binar((a - b) >> 1, b)
21                return d, y, x - y
22            else
23                d, x, y = binar((b - a) >> 1, a)
24                return d, x - y, y
25            end
26        end
27    end
28 end
29
30 println(binar(12, 16))
31 println(binar(24, 36))
32
```

Рис. 4: Код для расширенного бинарного Алгоритма Евклида



A terminal window with a dark background. The first line shows a blue dot followed by the text "Activating project at `C:\Users\nastd\.julia\environments\v1.8`" in green. The second line shows the number "16" in white. The third line shows a grey square with a white asterisk followed by the text "Terminal will be reused by tasks, press any key to close it." in white.

```
● Activating project at `C:\Users\nastd\.julia\environments\v1.8`  
16  
* Terminal will be reused by tasks, press any key to close it.
```

Рис. 5: НОД

```
• Activating project at `C:\Users\nastd\.julia\environments\v1.8`  
  (12, -1, 1)  
  (16, 2, -3)  
  * Terminal will be reused by tasks, press any key to close it.
```

Рис. 6: НОД и коэффициенты

Мы изучили различные вариации алгоритма Евклида и реализовать их программно на языке Julia.