Министерство образования и науки Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
—
Институт компьютерных наук и технологий
Кафедра «Информационная безопасность компьютерных систем»

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

по дисциплине «Теоретико-числовые методы в криптографии»
Вариант 36

Выполнил
студент гр. 33508/3 Проценко Е.Г.
____

Руководитель Павленко Е.Ю.

____

Санкт-Петербург
2016

СОДЕРЖАНИЕ

1 ЦЕЛЬ РАБОТЫ

Для каждой пары чисел найти наибольший общий делитель и его линейное представление, используя следующие алгоритмы:

1. Расширенный алгоритм Евклида.
2. Расширенный бинарный алгоритм Евклида.
3. Расширенный алгоритм Евклида с «усечёнными» остатками.

В отчете привести все итерации алгоритмов (последовательность остатков и две последовательности коэффициентов линейного представления). При числе итераций больше 20 привести первые 5 и последние 5 элементов каждой последовательности. Итерации должны быть пронумерованы.

При получении различных линейных представлений одного и того же наибольшего общего делителя обосновать корректность полученных результатов.

В отчете привести сравнение быстродействия реализованных алгоритмов.

Пары чисел приведены в Приложении А.

2 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Алгоритм Евклида – эффективный алгоритм для нахождения наибольшего общего делителя двух целых чисел.

Грубая оценка сложности алгоритма: $O((\log N)^2)$. Т.к. самая сложная операция – остаток от деления, а она приравнивается по сложности к операции умножения.

Расширенный алгоритм так же находит коэффициенты линейного представления:

$$\text{НОД}(a, b) = d = ax + by$$

Здесь x, y – целые коэффициенты линейного представления.

Бинарный алгоритм Евклида – по идее должен быть быстрее для реализации на компьютере, поскольку этот алгоритм использует такие особенности как побитовый сдвиг и основывается на следующих двух свойствах:

1. $\text{НОД}(2a, 2b) = 2 * \text{НОД}(a, b)$
2. $\text{НОД}(2a, b) = \text{НОД}(a, b)$

Алгоритм Евклида с усеченными остатками – позволяет избежать некоторого количества итераций, хотя сложность вычислений остается прежней.

3 РЕЗУЛЬТАТЫ РАБОТЫ

3.1 Первая пара чисел

3.1.1 Расширенный алгоритм Евклида

```
1:
r[i] = 1673588164769116617
x[i] = 0
y[i] = 1
q[i] = 1
2:
r[i] = 745400862137017682
x[i] = 1
y[i] = -1
q[i] = 22
3:
r[i] = 337069197754727613
x[i] = -22
y[i] = 23
q[i] = 2
4:
r[i] = 71262466627562456
x[i] = 45
y[i] = -47
q[i] = 4
5:
r[i] = 52019331244477789
x[i] = -202
y[i] = 211
q[i] = 1
```

```
19:
r[i] = 781038220139
x[i] = -6415828
y[i] = 6701583
q[i] = 2
20:
r[i] = 382090640068
x[i] = 18289067
y[i] = -19103645
q[i] = 2
21:
r[i] = 16856940003
x[i] = -42993962
y[i] = 44908873
q[i] = 22
22:
r[i] = 11237960002
x[i] = 964156231
y[i] = -1007098851
q[i] = 1
23:
r[i] = 5618980001
x[i] = -1007150193
y[i] = 1052007724
```

Рисунок 1 – Первые 5 и крайние 5 итераций расширенного алгоритма Евклида для первой пары чисел

3.1.2 Расширенный бинарный алгоритм Евклида

```
1:
u = 372700431068508841
v = 16363187733700607776
A = 8367944082384558309
B = -8740644513453067150
C = -8367944082384558309
D = 8740644513453067151
2:
u = 372700431068508841
v = 138649185609635152
A = 8367944082384558309
B = -8740644513453067150
C = 6537456314362936178
D = -6828628526135208710
3:
u = 364034856967906644
v = 8665574100602197
A = -6684549081436102242
B = 6982272667973250906
C = 15052493163820660551
D = -15722917181426318056
4:
u = 82343140141374464
v = 8665574100602197
A = -8355686351795127803
B = 8727840834966563633
C = 15052493163820660551
D = -15722917181426318056
5:
u = 80413222794311
v = 8585160877807886
A = 15207769721726809026
B = -15885109612514301239
C = -155276557906148475
D = 162192431087983183
6:
u = 80413222794311
v = 4212167216109632
A = 15207769721726809026
B = -15885109612514301239
C = -6917463918295324955
D = 7225561314605225681
7:
u = 14598110042598
v = 65815112751713
A = 14531360337726621137
B = -15178573584918782845
C = 676409384000187889
D = -706536027595518394
8:
u = 7299055021299
v = 58516057730414
A = 15633624251247868877
B = -16329931305912458572
C = -14957214867247680988
D = 15623395278316940178
9:
u = 7299055021299
v = 21958973843908
A = 15633624251247868877
B = -16329931305912458572
C = -23112231684871709371
D = 24141628945070928661
10:
u = 1809311560322
v = 5489743460977
A = 8859766048888958757
B = -9254371772000590013
C = 6773858202358910120
D = -7075559533911868559
11:
u = 904655780161
v = 4585087680816
A = 12797827106829037687
B = -13367830399453362156
C = -6023968904470127567
D = 6292270865541493597
12:
u = 618087800110
v = 286567980051
A = 5852374091271932140
B = -6113033379278271750
C = 6945453015557105547
D = -7254797020175090406
13:
u = 22475920004
v = 286567980051
A = -4019265969921139477
B = 4198280330535954531
C = 6945453015557105547
D = -7254797020175090406
14:
u = 5618980001
v = 280949000050
A = 3179155548711994285
B = -3320752174092544942
C = 3766297466845111262
D = -3934044846082545464
15:
u = 5618980001
v = 134855520024
A = 3179155548711994285
B = -3320752174092544942
C = -1296006815289438654
D = 1353729751051272210
16:
u = 5618980001
v = 11237960002
A = 3179155548711994285
B = -3320752174092544942
C = 9210759722953663346
D = -9620998377205646756
17:
u = 0
v = 5618980001
A = -1426224312764837388
B = 1489747014510278436
C = 4605379861476831673
D = -4810499188602823378
```

Рисунок 2 – Все итерации расширенного бинарного алгоритма Евклида для первой пары чисел

3.1.3 Расширенный алгоритм Евклида с «усеченными» остатками

```
1:
r = 1673588164769116617
x = 0
y = 1
q = 1
2:
r = 745400862137017682
x = 1
y = -1
q = 22
3:
r = 337069197754727613
x = -22
y = 23
q = 2
4:
r = 52019331244477789
x = -202
y = 211
q = 4
5:
r = 13533060478308455
x = -696
y = 727
q = 2
6:
r = 5710074904776212
x = 943
y = -985
q = 2
7:
r = 1484253567264150
x = 6107
y = -6379
q = 2
8:
r = 628657101491881
x = -8689
y = 9076
q = 2
9:
r = 174778372931105
x = -55659
y = 58138
q = 2
10:
r = 52160991349283
x = 79144
y = -82669
q = 3
11:
r = 15570193582771
x = 665326
y = -694959
q = 2
12:
r = 1944167080346
x = 5457411
y = -5700479
q = 5
13:
r = 781038220139
x = -6415828
y = 6701583
q = 2
14:
r = 382090640068
x = 18289067
y = -19103645
q = 2
15:
r = 11237960002
x = 964156231
y = -1007098851
q = 22
```

Рисунок 3 – Все итерации расширенного алгоритма Евклида с «усеченными» остатками для первой пары чисел

3.2 Вторая пара чисел

3.2.1 Расширенный алгоритм Евклида

```
1:
r[i] = 1580968849184611610824356717764899371689
x[i] = 0
y[i] = 1
q[i] = 1
2:
r[i] = 238985313365346201387898005916173640640
x[i] = 1
y[i] = -1
q[i] = 6
3:
r[i] = 147056968992534402496968682267857527849
x[i] = -6
y[i] = 7
q[i] = 1
4:
r[i] = 91928344372811798890929323648316112791
x[i] = 7
y[i] = -8
q[i] = 1
5:
r[i] = 55128624619722603606039358619541415058
x[i] = -13
y[i] = 15
q[i] = 1
```

```
42:
r[i] = 1149370115850244181062
x[i] = 828480428807516191
y[i] = -953716707180576393
q[i] = 1
43:
r[i] = 309445800421219587209
x[i] = -1152456498992132835
y[i] = 1326666242399546116
q[i] = 3
44:
r[i] = 221032714586585419435
x[i] = 4285849925783914696
y[i] = -4933715434379214741
q[i] = 1
45:
r[i] = 88413085834634167774
x[i] = -5438306424776047531
y[i] = 6260381676778760857
q[i] = 2
46:
r[i] = 44206542917317083887
x[i] = 15162462775336009758
y[i] = -17454478787936736455
```

Рисунок 4 – Первые 5 и крайние 5 итераций расширенного алгоритма Евклида для второй пары чисел

3.2.2 Расширенный бинарный алгоритм Евклида

```
1:
u = 2389853133653462013878980059161736406640
v = 1580968849184611610824356717764899371689
A = 1
B = -1
C = 0
D = 1
2:
u = 37341455213335343966859063424402131355
v = 1577234703663278076427670811422459158554
A = 963402892471872700346092374887985554623
B = -1109034567803880541816842722243153866888
C = -963402892471872700346092374887985554623
D = 1109034567803880541816842722243153866889
3:
u = 37341455213335343966859063424402131355
v = 784883206310305503817149499368789366142
A = 963402892471872700346092374887985554623
B = -1109034567803880541816842722243153866888
C = -654619914115503245106960203449528646090
D = 753574770430841906619136721524194294168
4:
u = 37341455213335343966859063424402131355
v = 388707457633819217511888843341954469936
A = 963402892471872700346092374887985554623
B = -1109034567803880541816842722243153866888
C = -1290712849529624322899572476612749877668
D = 1485821953019301495126411083005251013972
5:
u = 37341455213335343966859063424402131355
v = 205600705807801666978071463664319412366
A = 963402892471872700346092374887985554623
B = -1109034567803880541816842722243153866888
C = -648830233271321317821226475235057579055
D = 746909899230097432209179734010713802179
38:
u = 751511229594390426079
v = 4968815423906440228898
A = 1441673184555673556503337276651428182173
B = -1659602031135510270942474746892098245142
C = -1163582519585635472866160628668771506110
D = 1339474114927967576319376682015143942202
39:
u = 751511229594390426079
v = 11670527330171710146168
A = 1441673184555673556503337276651428182173
B = -1659602031135510270942474746892098245142
C = -942084389859776619307699074936171372856
D = 1084493478592523258916191555555347724528
40:
u = 751511229594390426079
v = 707304686677073342192
A = 1441673184555673556503337276651428182173
B = -1659602031135510270942474746892098245142
C = -1559433733288145633916799661018449603780
D = 1795163715959575678306998691336516710708
41:
u = 707304686677073342192
v = 442065429173170838887
A = 1143895580590029755917048076023856439487
B = -1316811222745494297783598484180362286479
C = 2977776039656438005862892006275717426866
D = -3427908083900159731588762627117359586663
42:
u = 0
v = 442065429173170838887
A = 663010847487577090247226957866675181357
B = -763234109465928689821991924620140253677
C = 2977776039656438005862892006275717426866
D = -3427908083900159731588762627117359586663
```

Рисунок 5 – Первые 5 и крайние 5 итераций расширенного бинарного алгоритма Евклида для второй пары чисел

3.2.3 Расширенный алгоритм Евклида с «усеченными» остатками

```
1:
r = 1580968849184611610824356717764899371689
x = 0
y = 1
q = 1
2:
r = 147056968992534402496968682267857527849
x = -6
y = 7
q = 6
3:
r = 55128624619722603606039358619541415058
x = -13
y = 15
q = 1
4:
r = 36799719753089195284889965028774697733
x = 20
y = -23
q = 1
5:
r = 18328904866633408321149393590766717325
x = -33
y = 38
q = 2
```

```
28:
r = 5525817864664635485875
x = -143447781746333741
y = 165131898476332776
q = 3
29:
r = 4067001948393171717604
x = 180528288438282903
y = -207817636742636947
q = 1
30:
r = 1149370115850244181062
x = 828480428807516191
y = -953716707180576393
q = 2
31:
r = 221032714586585419435
x = 4285849925783914696
y = -4933715434379214741
q = 3
32:
r = 44206542917317083887
x = 15162462775336009758
y = -17454478787936736455
q = 2
```

Рисунок 6 – Первые 5 и крайние 5 итераций расширенного алгоритма Евклида с «усеченными» остатками для второй пары чисел

3.3 Третья пара чисел

3.3.1 Расширенный алгоритм Евклида

```
1:
r[i] = 475286006558019976423825958437805064648174968365771563081433638986150965682466691
x[i] = 0
y[i] = 1
q[i] = 1
2:
r[i] = 115554317171216812001688092976701748224544209843277700457715789431977208440715562
x[i] = 1
y[i] = -1
q[i] = 4
3:
r[i] = 13068737873152728417073586530998071749998128992660761250570481258242131919604443
x[i] = -4
y[i] = 5
q[i] = 8
4:
r[i] = 11004414185994984665099400728717174224559177901991610453151939366040153083880188
x[i] = 33
y[i] = -41
q[i] = 1
5:
r[i] = 2064323687157743751974185802280897525438951090669150797418541892201978835724255
x[i] = -37
y[i] = 46
q[i] = 5

66:
r[i] = 128880838467941574867179092413108238493780
x[i] = 44552867752642416494058361724060741229
y[i] = -55384821860568902739924381314311583527
q[i] = 7
67:
r[i] = 7088446115736786617694850082720953117157 9
x[i] = -34427534115691987946464311316818792324 5
y[i] = 42797757816233272668943300773336975380 4
q[i] = 1
68:
r[i] = 5799637731057370869023059158589870732220 1
x[i] = 38882820890956229595870147489224866447 4
y[i] = -48336240002290162942935738904768133733 1
q[i] = 1
69:
r[i] = 1288808384679415748671790924131082384937 8
x[i] = -73310355006648217542334458806043658771 9
y[i] = 91133997818523435611879039678105109113 5
q[i] = 4
70:
r[i] = 6444041923397078743358954620655411924689
x[i] = 33212424091754909976520798271339950153 50
y[i] = -41287223127638390539045189761718857018 71
```

Рисунок 7 – Первые 5 и крайние 5 итераций расширенного алгоритма Евклида для третьей пары чисел

3.3.2 Расширенный бинарный алгоритм Евклида

```
1:
u = 115554317171216812001688092976701748224544209843277700457715789431977208440715 62
v = 475286006558019976423825958437805064648174968365771563081433638986150965682466 91
A = 1
B = -1
C = 0
D = 1
2:
u = 5777715858560840600084404648835087411227210492163885022885789471598860422035781
v = 41750884797241157042298191194945419053590286344413271285257574427016236146210910
A = 23764300327900998821191297921890253232408748418288578154071681949307548284123346
B = -29542016186461839421275702570725340643635958910452463176957471420906408706159127
C = -23764300327900998821191297921890253232408748418288578154071681949307548284123346
D = 29542016186461839421275702570725340643635958910452463176957471420906408706159128
3:
u = 5777715858560840600084404648835087411227210492163885022885789471598860422035781
v = 15097726540059737921064690948637622115567932680042750619742997741909257651069674
A = 23764300327900998821191297921890253232408748418288578154071681949307548284123346
B = -29542016186461839421275702570725340643635958910452463176957471420906408706159127
C = -35646450491851498231786946882835379848613122627432867231107522923961322426185019
D = 44313024279692759131913553856088010965453938365678694765436207131359613059238691
4:
u = 5777715858560840600084404648835087411227210492163885022885789471598860422035781
v = 1771147411469028360447940825483723646556755847857490286985709399355768403499056
A = 23764300327900998821191297921890253232408748418288578154071681949307548284123346
B = -29542016186461839421275702570725340643635958910452463176957471420906408706159127
C = -17823225245925749115893473441417689924306561313716433615553761461980661213092510
D = 22156512139846379565956776928044005482726969182839347382718103565679806529619346
5:
u = 5667019145344026327556408347242354683317413251672791879949182634139124896817090
v = 1106967132168142725279963015927327279097972404910931429366068374597355252186 91
A = -48271235041048903855544823903839576878330270224648674375458103959530957452125 54
B = 6000722037875061132446627084678584818238554153685656582819486382371614268438572
C = 28591423832005889206745780312274210920241775440753445591617492345260644029335900
D = -35542738224336900553722329655403925461874513064138119759776957803278022974597699
```

Рисунок 8 – Первые 5 итераций расширенного бинарного алгоритма Евклида для третьей пары чисел

```
97:
u = 12888083846794157486717909241310823849378
v = 148212964238132811097255956275074474267847
A = 28857925552789346676652966794028267913468853947466675143244565558091139594822297
B = -35874033404101203050668396219412999325320989080578367455040236887464604693850593
C = 1881634684449366597713137463469942687389168605079331519515775999029533892108767
D = -2339108727714054924066900430619384788259085817614321526504652726483882600039844
98:
u = 6444041923397078743358954620655411924689
v = 141768922314735732353897001654419062343158
A = 38193263104295672159517781318904387189143175392021915725693964728353118081534494
B = -47479032888512440946609900680431840306296453450741646904477589864638711053084423
C = -36311628419846305561804643855434444501754006786942584206178188729323584189425727
D = 45139924160798386022543000249812455518037367633127325377972937138154828453044579
99:
u = 6444041923397078743358954620655411924689
v = 64440419233970787433589546206554119246890
A = 38193263104295672159517781318904387189143175392021915725693964728353118081534494
B = -47479032888512440946609900680431840306296453450741646904477589864638711053084423
C = -32584776986317826119228805324731356207611430367204629674711377143707361892124012
D = 40506978782449794536605698234612727421679178356852846416506587012809716573447586
100:
u = 6444041923397078743358954620655411924689
v = 25776167693588314973435818482621647698756
A = 38193263104295672159517781318904387189143175392021915725693964728353118081534494
B = -47479032888512440946609900680431840306296453450741646904477589864638711053084423
C = -54485651597454585219132183981270065292948890575624230563049653300206799027596500
D = 67732522279737338214912749797738204017136042629168070112730883371043569339808216
101:
u = 0
v = 6444041923397078743358954620655411924689
A = 51814676003659318464300827314221903512380398035927973366456378053404817838433619
B = -64412163458446775500338088129866391310580464108033664432660310707399603388036477
C = -13621412899363646304783045995317516323237222643906057640762413325051699756899125
D = 16933130569934334553728187449434551004284010657292017528182720842760892334952054
```

Рисунок 9 – Крайние 5 итераций расширенного бинарного алгоритма Евклида для третьей пары чисел

3.3.3 Расширенный алгоритм Евклида с «усеченными» остатками

```
1:
r = 47528600655801997642382595843780506464817496836577156308143363898615096568246691
x = 0
y = 1
q = 1
2:
r = 11555431717121681200168809297670174822454420984327770045771578943197720844071562
x = 1
y = -1
q = 4
3:
r = 1100441418599498466509940072871717422455917790199161045315193936604015308388018
x = 33
y = -41
q = 8
4:
r = 206432368715774375197418580228089752543895109066915079741854189220197883572425
x = -37
y = 46
q = 5
5:
r = 68279575020626590522847171731268659736442244864585646605922990503025890525893
x = 218
y = -271
q = 3
50:
r = 3176912668234759820475964627983118078871677
x = -8110065159984059030587019851166721468
y = 10081831693913417050038252970942840291
q = 2
51:
r = 973050330432958890247202147718967200628039
x = -32405266888422964006234581099762734642
y = 40283825138350407509962338533188669115
q = 2
52:
r = 70884461157367866176948500827209531171579
x = -344275341156919879464643113168187923245
y = 427977578162332726689433007733369753804
q = 7
53:
r = 57996377310573708690230591585898707322201
x = 388828208909562295958701474892248664474
y = -483362400022901629429357389047681337331
q = 1
54:
r = 6444041923397078743358954620655411924689
x = 3321242409175490997652079827133995015350
y = -4128722312763839053904518976171885701871
q = 4
```

Рисунок 10 – Первые 5 и крайние 5 итераций расширенного алгоритма Евклида с «усеченными» остатками для второй пары чисел

3.4 Линейное представление

Далее будут приведены результаты работы трех алгоритмов. Первое число – НОД. Второе и третье – коэффициенты линейного представления. Первая строка – расширенный алгоритм Евклида. Вторая строка – расширенный бинарный алгоритм Евклида. Третья – расширенный алгоритм Евклида с «усеченными» остатками.

```
[5618980001, -1007150193, 1052007724]
[5618980001, 4605379861476831673, -4810499188602823378]
[5618980001, -1007150193, 1052007724]
```

Рисунок 11 – результаты работы алгоритмов для первой пары чисел

```
[4420654291731708 3887, 1516246277533600 9758, -1745447878793673 6455]
```

```
[44206542917317083887, 15162462775336009758, -17454478787936736455]
[44206542917317083887, 29777603965643800586289200627571742686, -34279080839001597315887626271173595866 3]
[44206542917317083887, -20600769200112057289, 23714860464715497312]
```

Рисунок 12 - результаты работы алгоритмов для второй пары чисел

```
[6444041923397078743358954620655411924689, 3321242409175490997652079827133995015350, -4128722312763839053904518976171885701871]
[
    6444041923397078743358954620655411924689,
    -13621412899363646304783045995317516323237222643906057640762413325051699756899125,
    1693313056993433455372818744943455100428401065729201752818272084276089233495 2054
]
[6444041923397078743358954620655411924689, -4054345959241973173075424415194431603069, 5040062290949073410023309372952936793006]
```

Рисунок 13 - результаты работы алгоритмов для третьей пары чисел

В разных случаях могут получаться разные коэффициенты x и y.

Рассмотрим 2 уравнения: $ax + by = d$ и $ax' + by' = d$.

Вычитаем из одного другое, получаем: $a(x - x') = b(y - y')$.

Нужно найти такие s, t, чтобы $at = bs$.

Пусть $a = a_1 d$, $b = b_1 d$. $\text{НОД}(a_1, b_1) = 1$. $a_1 t = b_1 s$.

s должно делаться на a_1. $t \vdots b_1$.

Пусть $t = b_1 k$, $s = a_1 k$, $k -$ целое.

$$x - x' = \frac{b}{\text{НОД}(a, b)} * k$$

$$x' = x - \frac{b * k}{\text{НОД}(a, b)}$$

$$y' = y + \frac{a * k}{\text{НОД}(a, b)}$$

Таким образом, пара чисел на ограничена одним линейным представлением

4 ВЫВОД

Это работа познакомила меня с очень интересной особенностью языка Python: в переменную можно записать очень большое число, почти сколь угодно большое.

Познакомился с алгоритмом нахождения НОД, думаю, это очень важный алгоритм, когда речь идет о криптографических алгоритмах.

Были проведены замеры количества времени, которое выполняется алгоритм и количество итераций для каждой пары чисел в каждом алгоритме. Результаты измерений моно увидеть в Приложении Б.

Алгоритм с «усеченными» остатками в теории должен был давать меньшее кол-во итераций и меньшее врем вычислений. Практический результат это подтверждает.

Однако бинарный алгоритм, который должен быстро считаться на компьютерах из-за операции сдвига, показал очень плохой результат. Возможно это связано с одной или всеми следующими утверждения:

- На каждой итерации цикла приходится делать очень много проверок на делимость чисел на 2.
- Замедление алгоритма связано с особенностями языка Python, т.к. каждая переменная – это объект, а не просто число.
- Автор кода плохо написал алгоритм

Возможны и другие причины.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. «Теоретико-числовые методы в криптографии» Е.Б.Маховенко 2006

ПРИЛОЖЕНИЕ А

```
numbers = (
    [
        17481289026906134299,
        16735888164769116617,
    ],
    [
        181995416254995781221225472368107301232 9,
        158096884918461161082435671776489937168 9
    ],
    [
        590840323729236788425514051414506812872719178209049263539149428418128174123182 53,
        475286006558019976423825958437805064648174968365771563081433638986150965682466 91,
    ],
)
```

Рисунок 14 – Пары чисел, необходимые для демонстрации работы алгоритмов.

ПРИЛОЖЕНИЕ Б

Для каждой пары чисел и алгоритма для замера времени выполнения функция вызывалась 1000000 раз.

Таблица 1 – Измерения работы алгоритмов

Пара чисел	Параметр измерения	Расширенный алгоритм Евклида	Расширенный бинарный алгоритм Евклида	Расширенный алгоритм с “усеченными” остатками
1	Итерации алгоритма	23	17	15
	Время	39.943881057349735	56.822424654786715	36.2147915377467
2	Итерации алгоритма	46	42	32
	Время	93.1904101076962	139.64993112626624	89.8772101711601
3	Итерации алгоритма	70	101	54
	Время	165.00916334985078	330.95354647341253	160.6251639519881

ПРИЛОЖЕНИЕ В

Исходный код

```
def setter(a, b):
    r = []
    if a > b:
        r.append(a)
        r.append(b)
    else:
        r.append(b)
        r.append(a)
    return r


def extended_euclid(a, b):
    r = setter(a, b)
    x = [1, 0]
    y = [0, 1]
    q = [0]
    i = 1

    while True:
        # calculating new iteration
        r.append(r[i - 1] % r[i])

        # checking if nod is found
        if r[i + 1] == 0:
            break
        else:
            # calculating new iteration
            q.append(int(r[i - 1] / r[i]))
            x.append(x[i - 1] - x[i] * q[i])
            y.append(y[i - 1] - y[i] * q[i])

            i += 1

    return [r[i], x[i], y[i]]


def extended_cut_euclid(a, b):
    def swapper():
        nonlocal r
        nonlocal x
        nonlocal y

        tmp = x[0]
        x[0] = x[1]
        x[1] = tmp

        tmp = y[0]
        y[0] = y[1]
        y[1] = tmp

        tmp = r[0]
        r[0] = r[1]
        r[1] = tmp
```

```
    r = setter(a, b)
    a = r[0]
    b = r[1]
    x = [1, 0]
    y = [0, 1]

    while True:
        q = int(r[0] / r[1])
        r[0] = r[0] % r[1]

        if r[0] == 0:
            d = r[1]
            return [d, x[1], y[1]]

        x[0] -= q * x[1]
        y[0] -= q * y[1]

        if abs(r[0]) > abs(r[1] / 2):
            r[1] -= r[0]
            x[1] -= x[0]
            y[1] -= y[0]
        else:
            swapper()


def extended_binary_euclid(a, b):
    def is_even(x):
        bit = 0x1
        return x & bit == 0

    r = setter(a, b)
    a = r[0]
    b = r[1]

    # step 1
    g = 1

    # step 2
    while is_even(a) and is_even(b):
        a >>= 1
        b >>= 1
        g <<= 1

    # step 3
    u = a
    v = b
    A = 1
    B = 0
    C = 0
    D = 1

    # step 4
    while u != 0:

        # step 4.1
        while is_even(u):
            # step 4.1.1
            u >>= 1
            # step 4.1.2
```

```
            if is_even(A) and is_even(B):
                A >>= 1
                B >>= 1
            else:
                A = (A + b) >> 1
                B = (B - a) >> 1

        # step 4.2
        while is_even(v):
            # step 4.2.1
            v >>= 1
            # step 4.2.2
            if is_even(C) and is_even(D):
                C >>= 1
                D >>= 1
            else:
                C = (C + b) >> 1
                D = (D - a) >> 1

        # step 4.3
        if u >= v:
            u -= v
            A -= C
            B -= D
        else:
            v -= u
            C -= A
            D -= B

    # step 5
    d = g * v
    z = C
    x = D

    # step 6
    return [d, z, x]


numbers = (
    [
        17481289026906134299,
        16735888164769116617,
    ],
    [
        181995416254995781221225472368107301 2329,
        158096884918461161082435671776489937 1689,
    ],
    [
        59084032372923678842551405141450681287271917820904926353914942841812817412318253,
        47528600655801997642382595843780506464817496836577156308143363898615096568246691,
    ],
)
```