1 Задачи 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.

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. Привести формулу для вычисления линейных представлений.

Привести сравнение быстродействия реализованных алгоритмов и объяснить полученные результаты.

2 Теоретические сведения

Определение НОД:

Целое число d≠0 называется *наибольшим общим делителем* целых чисел a_1, a_2,.., a_k (обозначается *d=НОД(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 .

Теорема (о существовании и линейном представлении наибольшего общего делителя):

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:

$$d = c_1a_1 + c_2a_2 + .. + c_ka_k, \text{где } c_i \in \text{Z}$$

Линейное представление $ax + by = d$ не единственно. Выведем общий вид коэффициентов x и y.

Пусть есть другое представление $ax' + by' = \text{НОД}(a, b)$. Тогда $a(x' - x) + b(y' - y) = 0$. Найдем такие целые числа s, t, для которых $as = bt$. Запишем $a = a_1 \cdot \text{НОД}(a, b)$, где $\text{НОД}(a_1, b_1) = 1$. Отсюда $a_1 s = b_1 t$, и это равенство выполняется для $\text{s} = b_1 k$, $\text{t} = a_1 k$, где k – произвольно целое число. Получаем

$$x' = x + \frac{bk}{\text{НОД}(a,b)}, y' = y - \frac{ak}{\text{НОД}(a,b)}$$

для произвольного целого числа k.

2.1 Расширенный алгоритм Евклида

Расширенный алгоритм Евклида находит НОД(a,b) и его линейное представление, то есть целые числа x и y, для которых ax+by=d. Для доказательства корректности алгоритма Евклида понадобятся две леммы.

Лемма 1:

Если числа *a* и *b* целые и *a* делится на *b*, то $b = \text{НОД}(\text{a}, \text{b})$.

Лемма 2:

Для любых целых чисел *a*, *b*, *c* выполняется равенство $\text{НОД}(\text{a} + \text{cb}, \text{b}) = \text{НОД}(\text{a}, \text{b})$.

Теорема 1:

Для любых *a*, $b > 0$ алгоритм Евклида останавливается и выдаваемое им число *d* является наибольшим общим делителем чисел *a* и *b*.

Расширенный алгоритм Евклида находит наибольший общий делитель *d* чисел *a* и *b* и его линейное представление, то есть целые числа *x* и *y*, для которых $ax + by = d$, и не требует «возврата».

Сложность алгоритма Евклида равна $O(log^2 a)$.

Лемма 3:

На каждой итерации расширенного алгоритма Евклида выполняется равенство $ax_i + by_i = r_i$ при $i \geq 0$, где r_i – остатки от деления на каждом шаге алгоритма.

2.2 Расширенный бинарный алгоритм Евклида

Этот вариант алгоритма Евклида оказывается более быстрым при реализации на компьютере, поскольку использует двоичное представление чисел a и b. Бинарный алгоритм Евклида основан на следующих свойствах НОД(a,b) (0<b≤a):

1. Если оба числа a и b четные, то $\text{НОД}(a,b) = 2 \cdot \text{НОД}(\frac{a}{2}, \frac{b}{2})$;
2. Если число a нечетное, a b – четное, то $\text{НОД}(a,b) = (\text{a}, \frac{b}{2})$;
3. Если оба числа a и b нечетные, $a > b$, то $\text{НОД}(\text{a}, b) = \text{НОД}(a-b, b)$;
4. Если $a = b$, то $\text{НОД}(a,b) = a$.

Сложность этого алгоритма – $O(log^2 a)$. Здесь по аналогии с леммой 3 предыдущего пункта на каждом шаге выполняются соотношения $u = aA + bB$ и $v = aC + bD$. Если все три числа u, A и B четные, то обе части равенства $u = aA + bB$ можно разделить на 2. Если же при четном u хотя бы одно из чисел A, B нечетное, то соотношение $u = aA + bB$ преобразуется в $u = aA + bB + ab - ab = a(A + b) + b(B - a)$. То же справедливо и для чисел v, C, D.

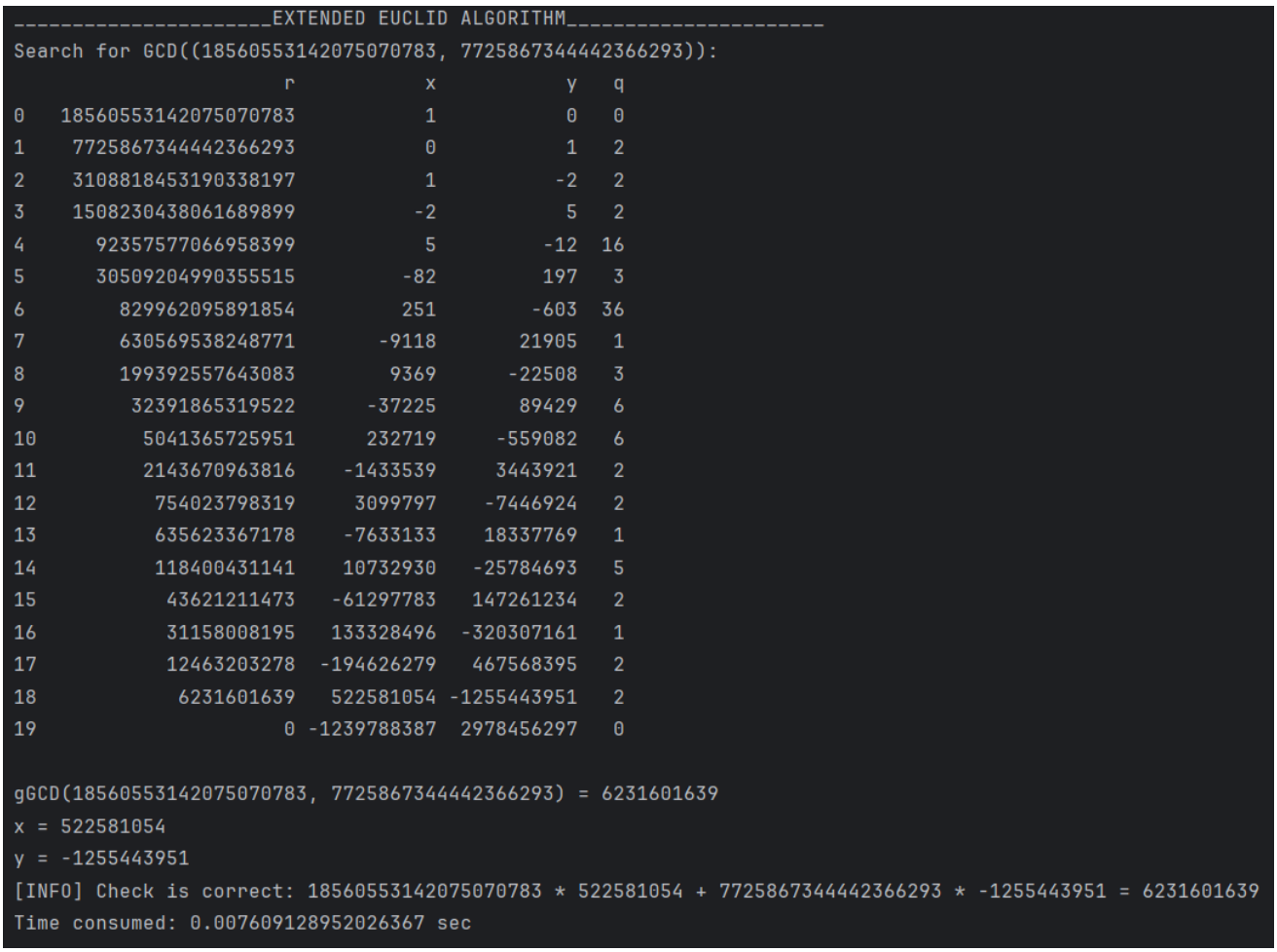
2.3 Расширенный алгоритм Евклида с «усечёнными» остатками

Данная модификация алгоритма Евклида основывается на следующем утверждении: если $r_{i+1} > \frac{r_i}{2}$, то $r_{i+1} = r_i - r_{i+1}$. Сложность данного алгоритма не отличается от сложности расширенного алгоритма Евклида.

3 Ход работы

3.1 Результаты работы программы

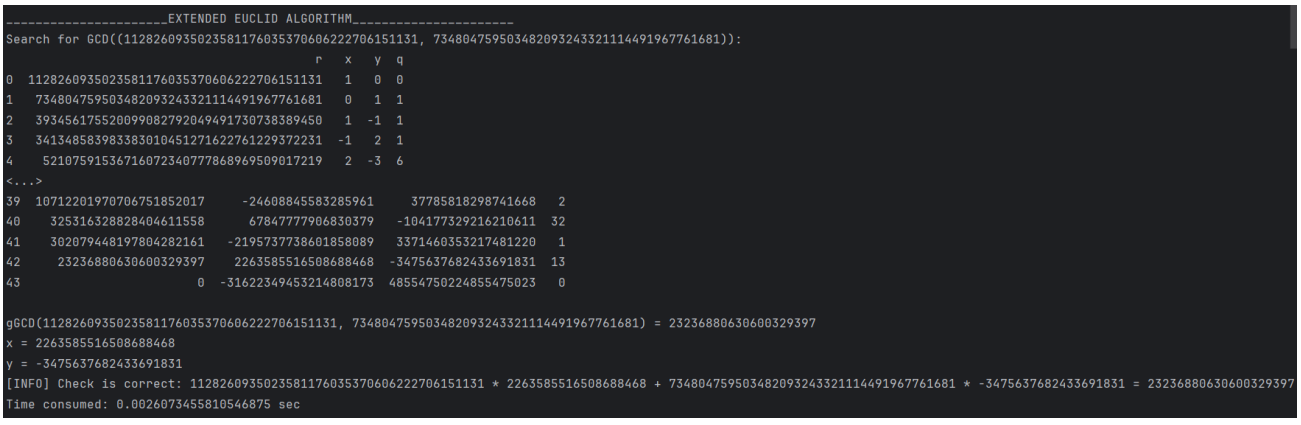
Разработанная программа была запущена для всех наборов чисел. Результаты работы программы представлены на Рисунок 1–9.



```
----------------------EXTENDED EUCLID ALGORITHM----------------------
Search for GCD((18560553142075070783, 7725867344442366293)):
                        r           x           y   q
0   18560553142075070783           1           0   0
1    7725867344442366293           0           1   2
2    3108818453190338197           1          -2   2
3    1508230438061689899          -2           5   2
4      92357577066958399           5         -12  16
5      30509204990355515         -82         197   3
6        829962095891854         251        -603  36
7        630569538248771       -9118       21905   1
8        199392557643083        9369      -22508   3
9         32391865319522      -37225       89429   6
10         5041365725951      232719     -559082   6
11         2143670963816    -1433539     3443921   2
12          754023798319     3099797    -7446924   2
13          635623367178    -7633133    18337769   1
14          118400431141    10732930   -25784693   5
15           43621211473   -61297783   147261234   2
16           31158008195   133328496  -320307161   1
17           12463203278  -194626279   467568395   2
18            6231601639   522581054 -1255443951   2
19                     0 -1239788387  2978456297   0

gGCD(18560553142075070783, 7725867344442366293) = 6231601639
x = 522581054
y = -1255443951
[INFO] Check is correct: 18560553142075070783 * 522581054 + 7725867344442366293 * -1255443951 = 6231601639
Time consumed: 0.007609128952026367 sec
```

Рисунок 1 – Результат работы программы для расширенного алгоритма Евклида, набор чисел №1



```
----------------------EXTENDED EUCLID ALGORITHM----------------------
Search for GCD((11282609350235811760353706062227061511 31, 7348047595034820932433211144919677616 81)):
                                          r   x   y  q
0  11282609350235811760353706062227061511 31   1   0  0
1   7348047595034820932433211144919677616 81   0   1  1
2   3934561755200990827920494917307383894 50   1  -1  1
3   3413485839833830104512716227612293722 31  -1   2  1
4    5210759153671607234077786896950901721 9   2  -3  6
<...>
39  1071220197070675185 2017     -24608845583285961     37785818298741668   2
40    3253163282840461155 8       6784777790683037 9   -104177329216210611  32
41    3020794481978042821 61   -2195737738601858089   3371460353217481220   1
42     2323688063060032939 7    2263585516508688468  -3475637682433691831  13
43                       0  -31622349453214808173  48554750224855475023   0

gGCD(11282609350235811760353706062227061511 31, 7348047595034820932433211144919677616 81) = 2323688063060032939 7
x = 2263585516508688468
y = -3475637682433691831
[INFO] Check is correct: 11282609350235811760353706062227061511 31 * 2263585516508688468 + 7348047595034820932433211144919677616 81 * -3475637682433691831 = 2323688063060032939 7
Time consumed: 0.0026073455810546875 sec
```

Рисунок 2 – Результат работы программы для расширенного алгоритма Евклида, набор чисел №2

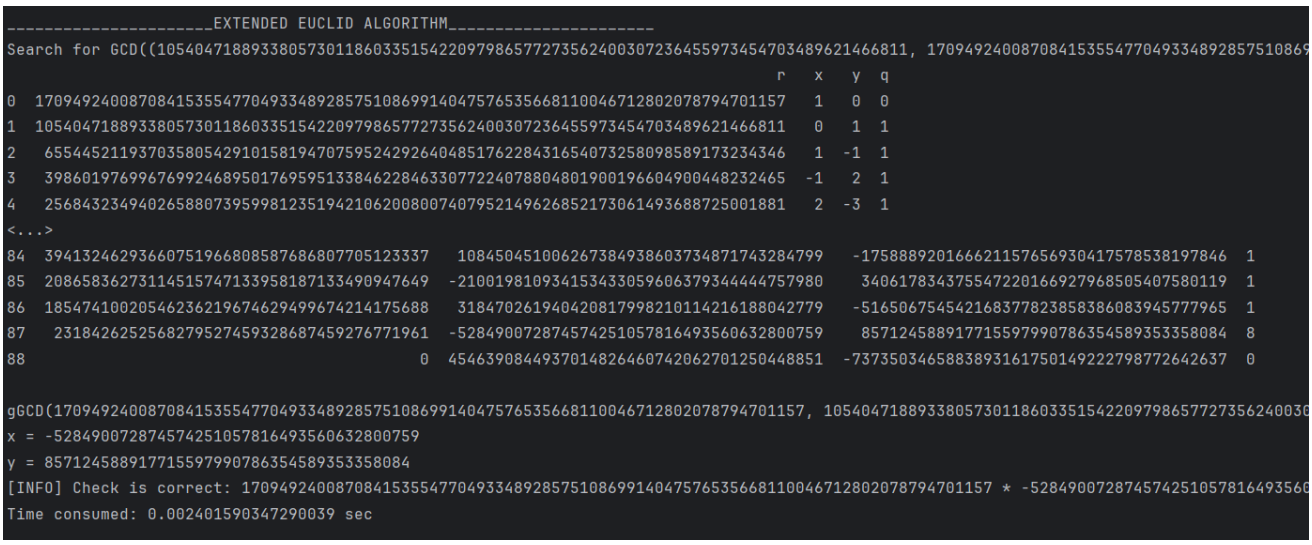


Рисунок 3 – Результат работы программы для расширенного алгоритма Евклида, набор чисел №3

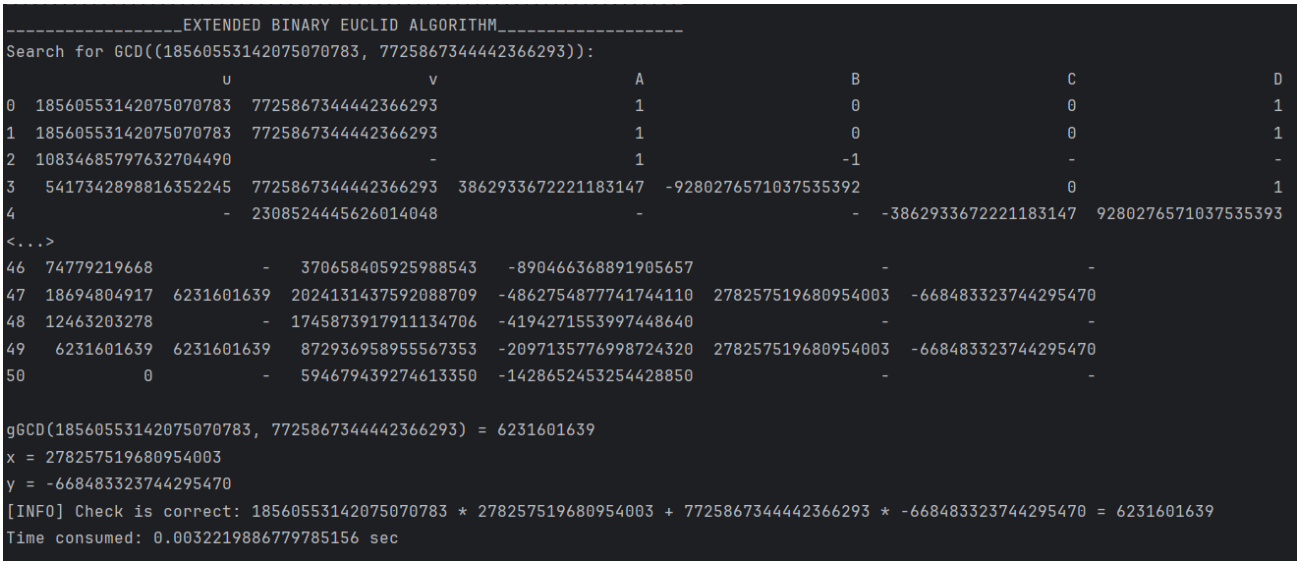


Рисунок 4 – Результат работы программы для расширенного бинарного алгоритма Евклида, набор чисел №1

Рисунок 5 – Результат работы программы для расширенного бинарного алгоритма Евклида, набор чисел №2

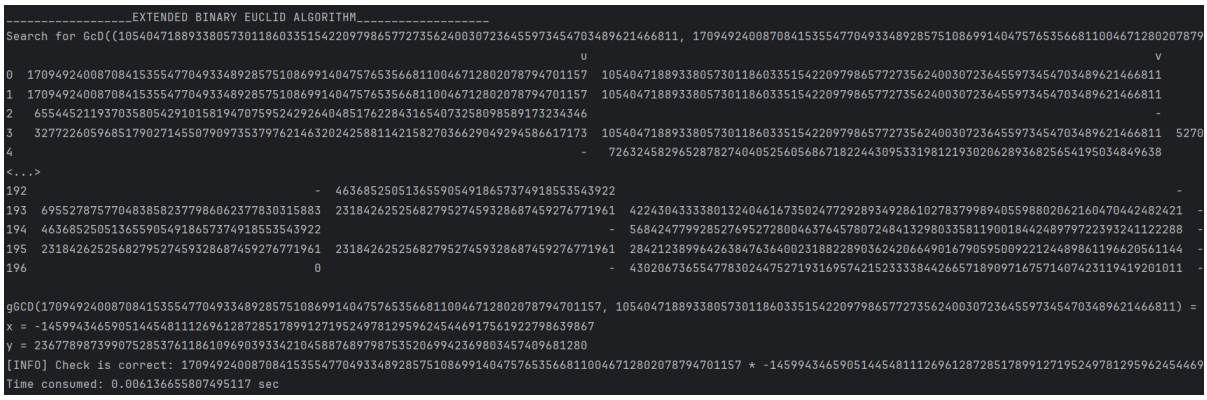


Рисунок 6 – Результат работы программы для расширенного бинарного алгоритма Евклида, набор чисел №3

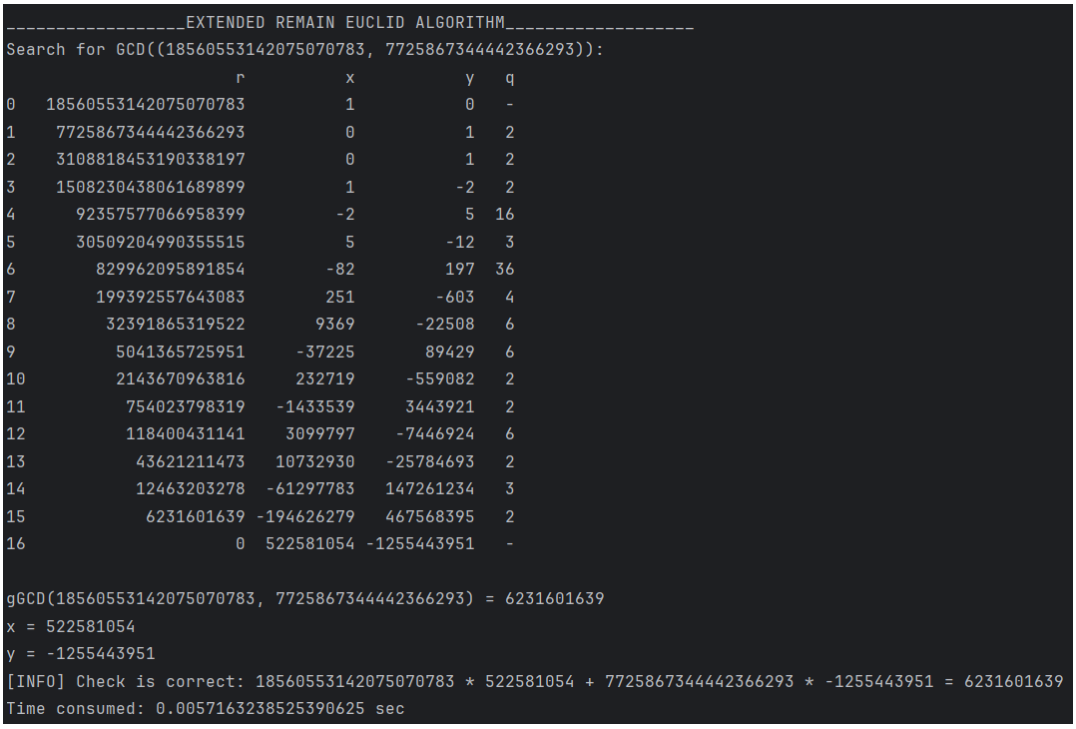


Рисунок 7 – Результат работы программы для расширенного алгоритма Евклида с "усеченными" остатками, набор чисел №1

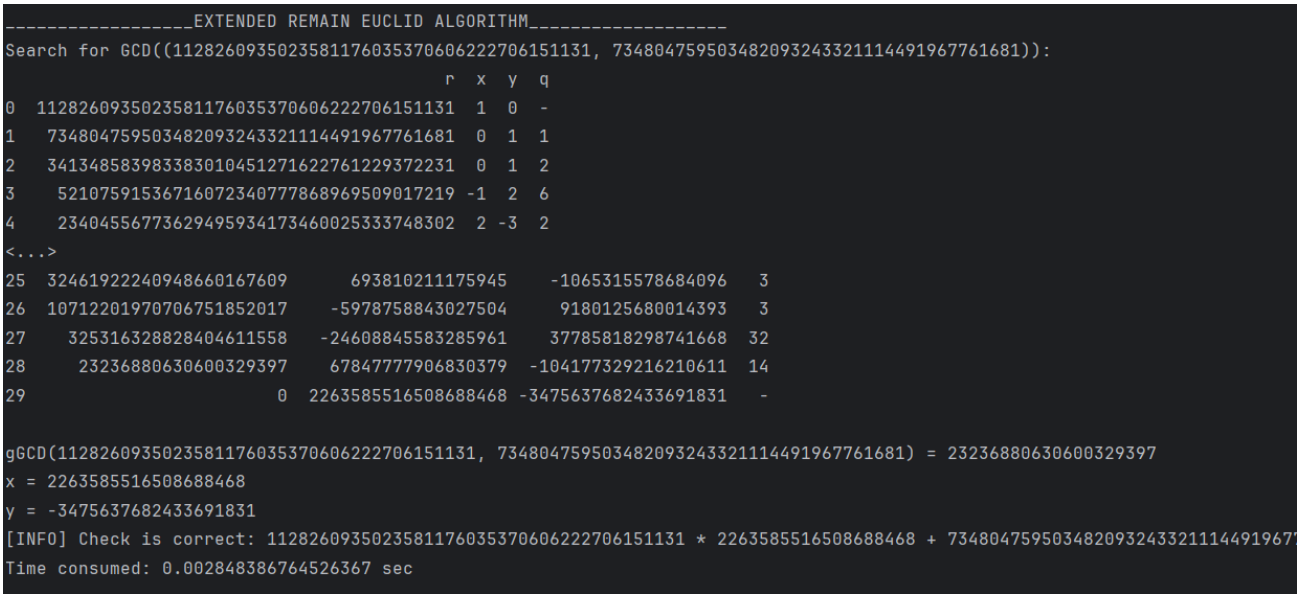


Рисунок 8 – Результат работы программы для расширенного алгоритма Евклида с "усеченными" остатками, набор чисел №2

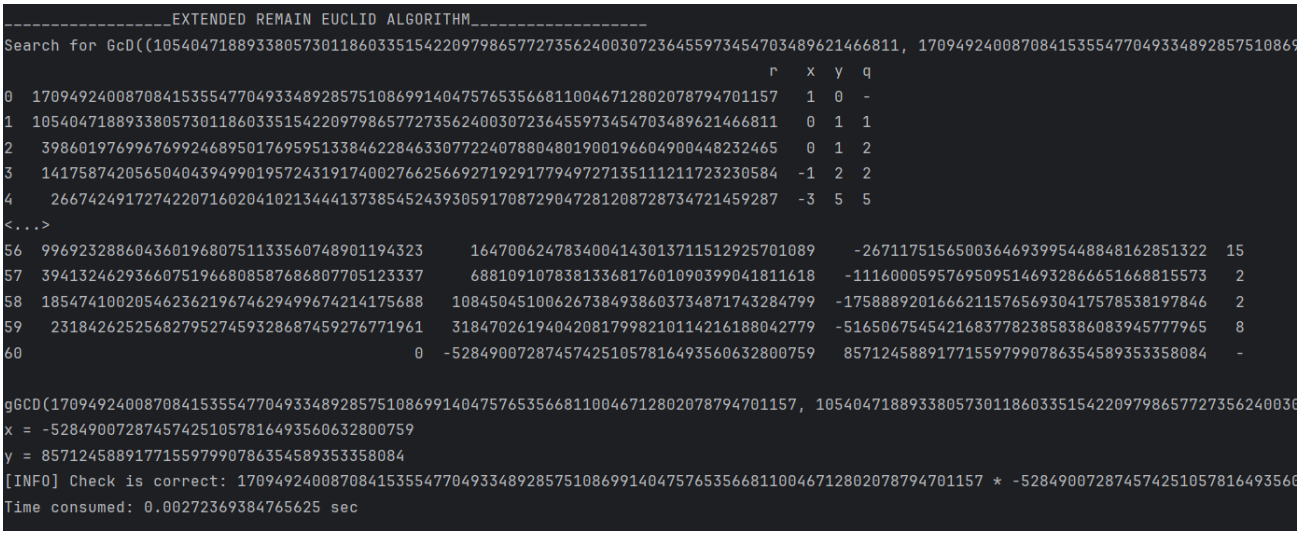


Рисунок 9 – Результат работы программы для расширенного алгоритма Евклида с "усеченными" остатками, набор чисел №3

3.2 Сравнение алгоритмов

Результаты работы программы в соответствие с входным набором чисел также занесены в Таблица 1. Можно заметить, что бинарный алгоритм дал иные коэффициенты линейного представления, которые также оказались верны для равенства $ax + by = d$. Это можно объяснить тем, что во время работы бинарный алгоритм использует другие операции – битовые сдвиги, которые дают иной результат.

Таблица 1 – Сравнение результатов работы программы

Набор чисел (№п\п), категория выходных данных		Значение данных для алгоритма…		
		Расширенного алгоритма Евклида	Расширенного бинарного алгоритма Евклида	Расширенного алгоритма Евклида с «усечёнными» остатками
1	НОД(a, b)	6231601639	6231601639	6231601639
	x	522581054	278257519680954003	522581054
	y	-1255443951	-668483323744295470	- 1255443951
	Проверка	Пройдена	Пройдена	Пройдена
	Число итераций	19	50	16
2	НОД(a, b)	23236880630600329397	2323688063060032939397	23236880630600329397
	x	2263585516508688468	6916338275682825102962735252470582416634	2263585516508688468
	y	-3475637682433691831	-1061973835762057075424855979559506407697	-3475637682433691831
	Проверка	Пройдена	Пройдена	Пройдена
	Число итераций	43	104	29
3	НОД(a, b)	2318426252568279527459328687459276771961	23184262525682795274593286874592761961	23184262525682795274593286874592761961
	x	-5284900728745742	-145994346590514454	-5284900728745742

		5105781649356063 2800759	811126961287285178 991271952497812959 624544691756192279 8639867	5105781649356063 2800759
	y	8571245889177155 9799078635458935 3358084	236778987399075285 376118610969039334 210458876897987535 206994236980345740 9681280	8571245889177155 9799078635458935 3358084
	Проверка	Пройдена	Пройдена	Пройдена
	Число итераций	88	196	60

Измерения времени работы алгоритмов отражены в Таблица 2. Результаты при первом запуске программы показали противоречивые данные – простой расширенный алгоритм на двух наборах данных оказался быстрее своих улучшенных версий. Было выдвинуто предположение, что причина тому – фиксация результатов каждой итерации и последующий вывод этих данных.

Таблица 2 – Сравнение времени работы алгоритмов

Набор чисел (№п\п)	Время работы алгоритма…(сек)		
	Расширенного алгоритма Евклида	Расширенного бинарного алгоритма Евклида	Расширенного алгоритма Евклида с «усечёнными» остатками
1	0.007609128952026367	**0.0032219886779785156**	0.005716323852539062 5
2	**0.0026073455810546875**	0.003330707550048828	0.002848386764526367
3	**0.002401590347290039**	0.006136655807495117	0.00272369384765625

Для того, чтобы проверить предположение, был проведён повторный замер времени «чистой» работы алгоритмов, не включающей запись и вывод данных о промежуточных результатах каждой итерации. Вывод программы показан на рисунках, а сравнение времени – в Таблица 3.

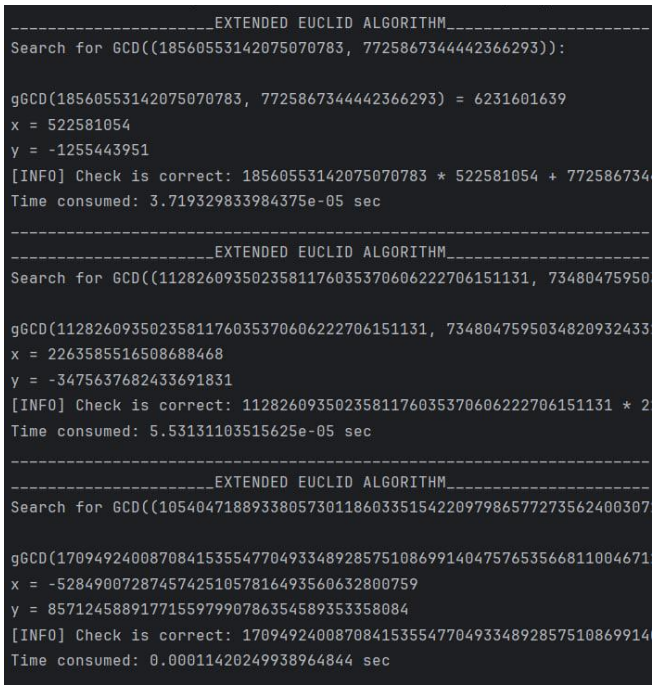


Рисунок 10 – Результаты новых замеров времени, расширенный алгоритм Евклида

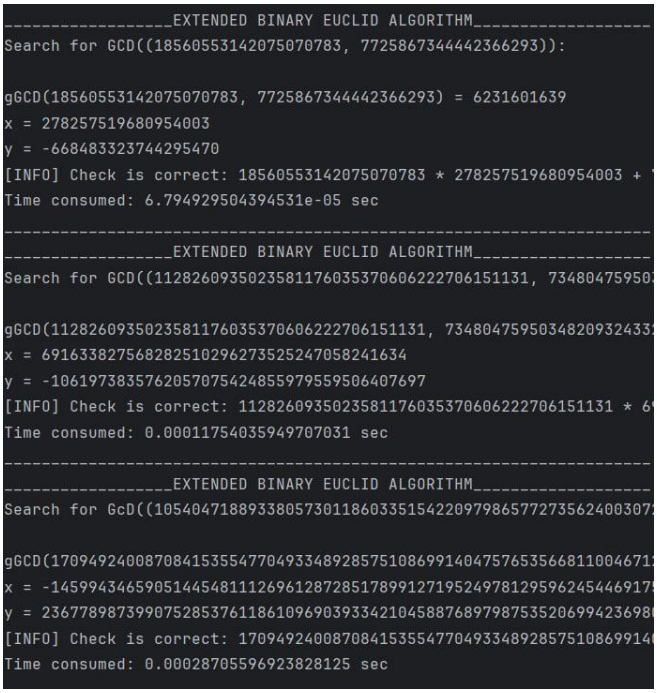


Рисунок 11 – Результаты новых замеров времени, расширенный бинарный алгоритм Евклида

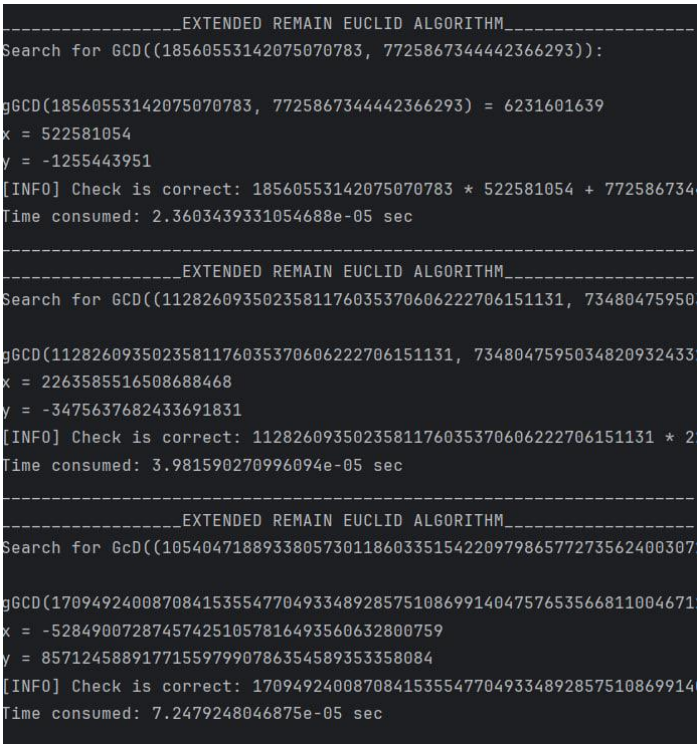


Рисунок 12 – Результаты новых замеров времени, расширенный алгоритм Евклида с "усеченными" остатками

Таблица 3 – Сравнение времени "чистой" работы алгоритмов

Набор чисел (№п\п)	Время работы алгоритма…(сек)		
	Расширенного алгоритма Евклида	Расширенного бинарного алгоритма Евклида	Расширенного алгоритма Евклида с «усечёнными» остатками
1	3.719329833984375e-05	6.794929504394531e-05	**2.3603439331054688e-05**
2	5.53131103515625e-05	0.0001175403594970703 1	**3.981590270996094e-05**
3	0.0001142024993896484 4	0.0002870559692382812 5	**7.2479248046875e-05**

Исходя из полученных данных, можно сделать вывод, что при реализации на языке Python наиболее эффективным показывает себя расширенный алгоритм

Евклида с «усеченными» остатками. Это объясняется тем, что по сравнению с другими алгоритмами он выполняет гораздо меньше итераций, и преимущество бинарного алгоритма в скорости вычисления на каждой итерации не настолько велико, чтобы компенсировать большую разницу в числе итераций.

4 Выводы

В ходе выполнения лабораторной работы были реализованы три различных алгоритма для нахождения наибольшего общего делителя (НОД) двух чисел и их линейного представления. При запуске программы было показано на практике, что линейное представление НОД не единственно и зависит от последовательности действий при нахождении коэффициентов. Результаты измерения времени работы программы при использовании различных алгоритмов показали, что на полученных наборах входных данных наиболее эффективным оказался расширенный алгоритм Евклида с «усеченными» остатками по причине малого числа итераций в сравнении с другими алгоритмами.