Лабораторная работа №4 Вычисление наибольшего общего делителя Отчет о выполнении

 Содержание Введение

Теоретическая часть

Реализованные алгоритмы

Структура проекта

Результаты тестирования

Примеры использования

Заключение

 Введение Цель работы: Изучение и программная реализация различных алгоритмов вычисления наибольшего общего делителя (НОД) целых чисел.

Задачи:

Реализовать классический алгоритм Евклида

Реализовать бинарный алгоритм Евклида

Реализовать расширенный алгоритм Евклида

Реализовать расширенный бинарный алгоритм Евклида

Провести тестирование на различных наборах данных

 Теоретическая часть Определение НОД Наибольший общий делитель целых чисел a 1 , a 2 , . . . , a k a 1 ,a 2 ,...,a k — это число d ≠ 0 d =0, удовлетворяющее условиям:

Каждое из чисел a 1 , a 2 , . . . , a k a 1 ,a 2 ,...,a k делится на d d

Если d 1 ≠ 0 d 1

=0 — другой общий делитель, то d d делится на d 1 d 1

Основные свойства Линейное представление: d

c 1 a 1 + c 2 a 2 + ⋯ + c k a k , c i ∈ Z d=c 1 a 1 +c 2 a 2 +⋯+c k a k ,c i ∈Z

Числа называются взаимно простыми, если НОД = 1

Попарно взаимно простые числа: НОД(a i , a j a i ,a j) = 1 для всех i ≠ j i □

⚙ Реализованные алгоритмы

1. Классический алгоритм Евклида Принцип работы: Повторное деление с остатком

python def euclidean_algorithm(a, b): r0, r1 = a, b while True: ri_plus_1 = r0 % r1 if ri_plus_1 == 0: return r1 r0, r1 = r1, ri_plus_1 Сложность: O (log (min (a , b))) O(log(min(a,b)))

2. Бинарный алгоритм Евклида Принцип работы: Использование двоичных операций и свойств:

Если оба числа четные: НОД(a,b) = 2·НОД(a/2, b/2)

Если a нечетное, b четное: НОД(a,b) = НОД(a, b/2)

Если оба нечетные: НОД(a,b) = НОД(a-b, b)

python def binary_euclidean_algorithm(a, b): g = 1 while a % 2 == 0 and b % 2 == 0: a, b, g = a//2, b//2, 2*g u, v = a, b while u != 0: # ... операции деления на 2 и вычитания return g * v Преимущества: Быстрее на компьютерах из-за использования битовых операций

3. Расширенный алгоритм Евклида Принцип работы: Находит НОД и коэффициенты линейного представления

python def extended_euclidean_algorithm(a, b): r0, r1 = a, b x0, x1 = 1, 0 y0, y1 = 0, 1

```
while r1 != 0:
    qi = r0 // r1
    r0, r1 = r1, r0 % r1
    x0, x1 = x1, x0 - qi * x1
    y0, y1 = y1, y0 - qi * y1

return r0, x0, y0
```

Выходные данные: d , x , y d,x,y такие, что a x + b y

d ax+by=d

4. Расширенный бинарный алгоритм Евклида Принцип работы: Комбинация бинарного алгоритма с вычислением коэффициентов

python def extended_binary_euclidean_algorithm(a, b): # Вынесение общих степеней двойки # Параллельное обновление коэффициентов # Бинарные операции и вычитания