Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**Отчёт**

**по лабораторной работе №1**

**Дисциплина: КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ**

**Тема: «ВВЕДЕНИЕ В ТЕОРИЮ ЧИСЕЛ. ДИОФАНТОВЫ УРАВНЕНИЯ»**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ К. А. Корнилов

Направление подготовки 02.03.02 Фундаментальная информатика и

информационные технологии

Направленность (профиль) Математическое и программное обеспечение

компьютерных технологий

Преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А. А. Крамаренко

**Тема:** Введение в теорию чисел. Диофантовы уравнения

**Цель:** Изучение расширенного алгоритма Евклида, методов решения Диофантовых уравнений.

**Ход работы:**

1. Для данных пар чисел найти НОД:(256,384), (714,218), (516, 438), (735, 525). Используем алгоритм Евклида и предположение R1 = Q \* R2 +R
2. Нод(256,384)=128 по алгоритму Евклида

|  |  |  |  |
| --- | --- | --- | --- |
| Q | R1 | R2 | R |
| 1 | 384 | 256 | 128 |
| 2 | 256 | 128 | 0 |
| 0 | 128 | 0 | 0 |

1. Нод(714,218) = 2 по алгоритму Евклида

|  |  |  |  |
| --- | --- | --- | --- |
| Q | R1 | R2 | R |
| 3 | 714 | 218 | 60 |
| 3 | 218 | 60 | 38 |
| 1 | 60 | 38 | 22 |
| 1 | 38 | 22 | 16 |
| 1 | 22 | 16 | 6 |
| 2 | 16 | 6 | 4 |
| 1 | 6 | 4 | 2 |
| 2 | 4 | 2 | 0 |
| 0 | 2 | 0 | 0 |

1. Нод(516,438)=6 по алгоритму Евклида

|  |  |  |  |
| --- | --- | --- | --- |
| Q | R1 | R2 | R |
| 1 | 516 | 483 | 78 |
| 4 | 438 | 78 | 48 |
| 1 | 78 | 48 | 30 |
| 1 | 38 | 30 | 18 |
| 1 | 30 | 18 | 12 |
| 1 | 18 | 12 | 6 |
| 2 | 12 | 6 | 0 |
| 0 | 6 | 0 | 0 |

1. Нод(735,525) = 105 по алгоритму Евклида

|  |  |  |  |
| --- | --- | --- | --- |
| q | r1 | r2 | r |
| 1 | 735 | 525 | 210 |
| 2 | 525 | 210 | 105 |
| 2 | 210 | 105 | 0 |
| 0 | 105 | 0 | 0 |

1. Для данных чисел найти значение функции Эйлера:
2. Ф(63) = 63 \* (1-1/3) \* (1-1/7) = 36
3. Ф(100) = 100 \* (1-1/2) \* (1-1/5) = 40
4. Ф(525) = 525 \* (1-1/5) \* (1-1/3) \* (1-1/7) = 240
5. Ф(31) = 31 - 1 = 30
6. Ф(274) = 274 \* (1 – ½) \* (1 – 1/137) = 136
7. Для заданных значений модуля найти все первообразные корни в данном поле.
   1. 7 – простое => существует образующий корень (т.е. такой элемент, порядок которого ф(7))

Вычислим функцию Эйлера от 7: ф(7) = 6 = 2 \* 3

Проверим все числа от 2 до 6, которые взаимно простые с 7, по следующим критериям: для всех p - простых делителей ф(7) должно выполняться a^p <> 1 mod 7 и a^ф(7) = 1 mod 7. Данное предположение основано на свойстве порядков подгрупп: порядок подгруппы делитель порядка всей группы. А в данном случае в поле мы рассматриваем группы относительно умножения.

* 1. Нод(2,7) = 1; 2^(6/2) = 1 mod 7 => не корень
  2. Нод(3,7) = 1; 3^(6/2) <> 1 mod 7 and 3^(6/3) <> 1 mod => корень

Проверим, что можно получить все элементы мультпликативный группы по модулю 7: 3, 3^2 = 2, 3^3 = 6, 3^4 = 4, 3^5 = 5, 3^6 = 1  
Таким образом, 3 -первообразный корень в поле по модулю 7.

* 1. Аналогичным образом строится проверка для остальных числе до 6. Получаются следующие результат: 4 – не корень, 5 – корень, 6 – не корень
  2. 11 – простое => существует образующий корень (т.е. такой элемент, порядок которого ф(11))

Вычислим функцию Эйлера от 7: ф(11) = 10 = 2 \* 5

Проверим все числа от 2 до 10, которые взаимно простые с 11, по следующим критериям: для всех p - простых делителей ф(7) должно выполняться a^p <> 1 mod 11 и a^ф(11) = 1 mod 11.

Аналогичным первому примеру образом получаем следующие результаты: 2 – корень, 3 – не корень, 4 – не корень, 5 – не корень, 6 - корень , 7 - корень, 8 - корень, 9 – не корень, 10 – не корень.

* 1. 13 – простое => существует образующий корень (т.е. такой элемент, порядок которого ф(13))

Вычислим функцию Эйлера от 7: ф(13) = 12 = 3 \* 4

Проверим все числа от 2 до 12, которые взаимно простые с 13, по следующим критериям: для всех p - простых делителей ф(7) должно выполняться a^p <> 1 mod 13 и a^ф(13) = 1 mod 13.

Аналогичным первому примеру образом получаем следующие результаты: 2 – корень, 3 – не корень, 4 – не корень, 5 – не корень, 6 - корень , 7 - корень, 8 – не корень, 9 – не корень, 10 – не корень, 11 – корень, 12 – не корень.

* 1. 13 – простое => существует образующий корень (т.е. такой элемент, порядок которого ф(13))

Вычислим функцию Эйлера от 7: ф(13) = 12 = 3 \* 4

Проверим все числа от 2 до 12, которые взаимно простые с 13, по следующим критериям: для всех p - простых делителей ф(13) должно выполняться a^p <> 1 mod 13 и a^ф(13) = 1 mod 13.

Аналогичным первому примеру образом получаем следующие результаты: 2 – корень, 3 – не корень, 4 – не корень, 5 – не корень, 6 - корень , 7 - корень, 8 – не корень, 9 – не корень, 10 – не корень, 11 – корень, 12 – не корень.

* 1. 19 – простое => существует образующий корень (т.е. такой элемент, порядок которого ф(19))

Вычислим функцию Эйлера от 19: ф(19) = 18 = 3 \* 2 \* 3

Проверим все числа от 2 до 18, которые взаимно простые с 19, по следующим критериям: для всех p - простых делителей ф(19) должно выполняться a^p <> 1 mod 19 и a^ф(19) = 1 mod 19.

Аналогичным первому примеру образом получаем следующие результаты: 2 – корень, 3 – корень, 4 – не корень, 5 – не корень, 6 – не корень , 7 - не корень, 8 – не корень, 9 – не корень, 10 – корень, 11 – не корень, 12 – не корень, 13 – корень, 14 – корень, 15 – корень, 16 – не корень, 17 – не корень, 18 – не корень,.

* 1. 23 – простое => существует образующий корень (т.е. такой элемент, порядок которого ф(23))

Вычислим функцию Эйлера от 23: ф(23) = 22 = 2 \* 11

Проверим все числа от 2 до 22, которые взаимно простые с 23, по следующим критериям: для всех p - простых делителей ф(23) должно выполняться a^p <> 1 mod 23 и a^ф(23) = 1 mod 23.

Аналогичным первому примеру образом получаем следующие результаты: 2 – не корень, 3 – не корень, 4 – не корень, 5 – корень, 6 – не корень , 7 - корень, 8 – не корень, 9 – не корень, 10 – корень, 11 – корень, 12 – не корень, 13 – не корень, 14 – корень, 15 – корень, 16 – не корень, 17 – корень, 18 – не корень, 19 – корень, 20 – корень, 21 – корень, 22 – не корень,.

1. Решить Диофантовы уравнения
   1. 5x + 7y = 14

Решим уравнение через расширенный алгоритм Эвклида.

Для начала найдем НОД(5,7) = 1. Проверим 14:1 = 14.

Тогда используем алгоритм Евклида и выразим 1:

7 = 5 \* 1 + 2

5 = 2 \* 2 + 1

2 = 2 \* 1 + 0

Выразим с помощью данных выражений 1 через 5 и 7:

1 = 5 – 2\*2 = 5 – 2\* ( 7 – 5 ) = 3\* 5 – 2 \* 7 = 1

Тогда 14 \* 3 \*5 – 14 \* 2 \* 7 = 1 => x = 42 + (-7)\*k и y = -28 +5\*k

* 1. 27x + 36y = 32

Решим уравнение через расширенный алгоритм Эвклида.

Для начала найдем НОД(27,36) = 9. Проверим 32:9 – не делится => нет решения.

* 1. 10x + 12y = 13

Решим уравнение через расширенный алгоритм Эвклида.

Для начала найдем НОД(10,12) = 2. Проверим 13:2 не делится => нет решения

* 1. 54x + 48y = 128

Решим уравнение через расширенный алгоритм Эвклида.

Для начала найдем НОД(54,48) = 6. Проверим 128:6 не делится => нет решения.

* 1. 14x + 27y = 49

Решим уравнение через расширенный алгоритм Эвклида.

Для начала найдем НОД(14,27) = 1. Проверим 49:1 = 14.

Тогда используем алгоритм Евклида и выразим 1:

27 = 14 \* 1 + 13

14 = 13 \* 1 + 1

13 = 1 \* 13 + 0

Выразим с помощью данных выражений 1 через 13 и 14:

1 = 14 – 13 \* 1 = 14 – (27 – 14 \* 1) = 2 \* 14 – 27 = 1

Тогда 98\*14 – 49\*27 = 49 => x = 98 + (-27)\*k и y = -49 +14\*k

* 1. 150x + 75y = 216

Решим уравнение через расширенный алгоритм Эвклида.

Для начала найдем НОД(150,75) = 75. Проверим 216:75 не делится => нет решения.

* 1. 18x + 35y = 36

Решим уравнение через расширенный алгоритм Эвклида.

Для начала найдем НОД(18,35) = 1. Проверим 36:1 = 36.

Тогда используем алгоритм Евклида и выразим 1:

35 = 18 \* 1 + 17

18 = 17 \* 1 + 1

17 = 1 \* 17 + 0

Выразим с помощью данных выражений 1 через 18 и 35:

1 = 18 – 17 \* 1 = 18 – ( 35 – 18) = 2 \* 18 – 35 = 1

Тогда 72 \* 18 – 36 \* 35 = 36 => x = 72 + (-35)\*k и y = -26 +18\*k

* 1. 50x + 44y = 121

Решим уравнение через расширенный алгоритм Эвклида.

Для начала найдем НОД(50,44) = 2. Проверим 121:2 не делится => нет решения.

1. Решить сравнения первой степени двумя способами.
   1. 3x = 19 mod 34

Алгоритм решения следующий:

1. Находим НОД(3,34).
2. Если он равен 1,то у нас будет 1 серия решений
3. Если он не равен 1, то проверяем 19 mod НОД(3,34) = 0.   
   Если данное равенство не выполнено, то решений нет.  
   Иначе делим все коэффициенты уравнения на найденный нод. В таком случае у нас будет НОД(3,34) серий решений, которые имеют вид x = x0+ 19/нод(3,34) \* k, где x0 – решений сокращенного сравнения.
4. X0 = 19/нод(3,34) \* (3/нод(3,34)) ^ - 1 mod 34/нод(3,34)
5. Или второй вариант через теорему Эйлера: a ^ ф(m) = 1 mod m => 3^ф(m) \* b = b mod m => a \* (a^(ф(34)-1)) \* b ) = b mod m =>

x = (a^(ф(m)-1)) \* b )

* + 1. НОД(3,34) = 1 => 1 серия
    2. По первому варианту решения: x = 19 \* 3 ^ -1 mod 34 = 19 \* 23 mod 34 = 29
    3. По второму варианту решения: x = 3 ^ 15 \* 19 mod 34 = 29
    4. Итоговый ответ: x = 29 + 34 \* k
  1. 7x = 11 mod 39
     1. Согласно раннее описанному алгоритму итоговый ответ: x = (7^23 \* 11) mod 39 + 39 \* k = (11\*7^-1) mod 39 + 39\*k = 35 + 39k
  2. 5x = 8 mod 21
     1. Согласно раннее описанному алгоритму итоговый ответ: x = (5^11 \* 8) mod 21 + 21 \* k = (8\*5^-1) mod 21 + 21\*k = 10 + 39k
  3. 15x = 35 mod 100
     1. Согласно раннее описанному алгоритму итоговый ответ 5 серий:   
        xобр = (3^7 \* 7) mod 20 = (7\*3^-1) mod 20 = 9  
        x0 = 9 + 100k

x1 = 29 + 100k

x2 = 49 + 100k

x3 = 69 + 100k

x4 = 89 + 100k

* 1. 9x = 21 mod 48
     1. Согласно раннее описанному алгоритму итоговый ответ 3 серии:   
        xобр = (3^7 \* 7) mod 16 = (7\*3^-1) mod 16 = 13  
        x0 = 13 + 48k

x1 = 29 + 48k

x2 = 45 + 100k

* 1. 14x = 8 mod 50
     1. Согласно раннее описанному алгоритму итоговый ответ 2 серии:   
        xобр = (14^19 \* 8) mod 25 = (4\*7^-1) mod 25 = 22  
        x0 = 22 + 50k

x1 = 47 + 50k

* 1. 16x = 19 mod 34
     1. Согласно раннее описанному алгоритму решение отсутствует, так как НОД(16,34) = 2 и 19 : 2 mod <> 0
  2. 22x = 11 mod 38
     1. Согласно раннее описанному алгоритму решение отсутствует, так как НОД(22,38) = 2 и 11 : 2 mod <> 0
  3. 6x = 7 mod 22
     1. Согласно раннее описанному алгоритму решение отсутствует, так как НОД(22,6) = 2 и 7 : 2 mod <> 0
  4. 4x = 3 mod 7
     1. Согласно раннее описанному алгоритму итоговый ответ 1 сериz:   
        x = (3 \* 4^5) mod 7 + 7k = (3\*4^-1) mod 7 + 7k = 6 + 7k
  5. 11x = 5 mod 17
     1. Согласно раннее описанному алгоритму итоговый ответ 1 серия:   
        x = (5 \* 11^15) mod 17 + 17k = (5\*11^-1) mod 17 + 17k = 2 + 17k
  6. 2x = 5 mod 11
     1. Согласно раннее описанному алгоритму итоговый ответ 1 серия:   
        x = (5 \* 2^9) mod 11 + 11k = (5\*2^-1) mod 11 + 11k = 8 + 11k

1. Реализовать программный продукт решения сравнений первой степени

двумя способами с указанием всех промежуточных шагов вычисления

(текущее значение коэффициентов расширенном алгоритме Евклида и

текущее значение степеней в формуле Эйлера), программный продукт так же

должен реализовывать возможность того, что сравнение не имеет решений

или имеет больше одного решения. В первом случае сообщать пользователю

с пояснением, во втором строить все возможные решения.

Для решения данной задачи была написана функция. Она решает уравнения вида ax = b mod m. Сначала в ней происходит сокращение коэффициентов на m. Затем происходит вычисления нод(a,m) и преобразования согласно алгоритму в пункте 5. При этом происходит сохранения количества серий решений

Внутри функции содержится вложенную функцию, вычисляющую расширенный алгоритм Евклида для двух чисел согласно его определению. Данная функция работает рекурсивно и возвращает нод(a,m), а также их коэффициенты для решения уравнения a\*q + m \* s = 1

Затем коэффициент при a используется для нахождение решения преобразованного сравнения в виде: x = b \* q.

Также содержится другая функция, которая вычисляет значение функции Эйлера через перебор всех чисел, нод которых с аргументом равен 1. Затем данное значение используется для нахождения решения преобразованного сравнения в виде: x = b \* a^(ф(m) – 1). При этом для вычисления степени используется еще одна отдельная функция, которая производить умножение аргумента на себя и сразу сокращение по модулю.

Затем найденные значения x используются для нахождения серий решений согласно алгоритму решения в пункте 5.

1. Написать реализацию вычисления функции Эйлера двумя способами.

Были написаны две функции для вычисления значений функции Эйлера. Первая функция осуществляет перебор всех чисел от 1 до аргумента и вычисляет их нод с аргументом. Если нод(I, аргумента) = 1, то счетчик увеличивается на 1. После перебора возвращается значение счетчика

Вторая функция вычисляет все простые делители аргумента. Затем вычисляется значение функции Эйлера как: n \* (1-1/p1) \* (1-1/p2) \* …, где n – значение аргумента, p1,p2,… - простые делители n

**Вывод:** Были изучены способы вычисления линейных сравнений, Диофантовых уравнения и функции Эйлера