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 №5**

**Дисциплина: КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ**

**Тема: «NP-Полные задачи теории чисел»**

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:** NP-полные задачи теории чисел.

**Цель:** Изучение алгоритмов факторизации и тестов на простоту.

**Ход работы:**

1. Привести пример пяти простых чисел Мерсена

2^2-1 = 3; 2^3 – 1 = 7; 2^5 – 1 =31; 2^7-1 = 127; 2^11-1 = 2047; 2^13 – 1 = 8191

1. Привести пример четырех простых чисел Ферма.

F0 = 2^(2^0)+1 = 3; F1 = 2^(2^1)+1 = 5; F2 = 2^(2^2)+1 = 17; F3 = 2^(2^3)+1 = 257

1. Провести тест Ферма для чисел 23, 41, 15, 35, 561 Каково название последнего числа?
2. 23. Пусть a = 2 => 2^^22 = 1 mod 23 => 32^4 \* 4 = 1 mod 23 => 9^4 \* 4 = 1 mod 23 => 12^2 \* 4 = 1 mod 23 => 12\*2 = 1 mod 23 => простое
3. 41. Пусть a = 2 => 2^40 = 1 mod 41 => 32^8 = 1 mod 41 => 40^4 = 1 mod 41 => (-1)^4 = 1 mod 41 => простое
4. 15. Пусть a = 2 => 2^14 = 1 mod 15 => 16^3 \* 2 = 1 mod 15 => 2<>1 => составное
5. 35. Пусть a = 2 => 2^34 = 1 mod 35 => 32^6 \* 16 = 1 mod 35 => (-3)^6 \* 16 = 1 mod 35 => 27^2 \* 4^2 = 1 mod 35 => 9<>1 => составное
6. 561. Пусть a = 2 => 2^560 = 1 mod 561 => 1024^56 = 1 mod 561 => 463^56 = 1 mod 561 => 98^56 = 1 mod 561 => 67^28 = 1 mod 561 => 1 = 1 => число Карлмайка, тк прошло тест, но является составным
7. Провести испытание квадратным корнем для чисел 23, 41, 15, 35, 561.
8. 23. 1 ^ 1 = mod 23; 2^2 = 4mod 23; 3^2 = 9; 4^2 = 16; 5^2 = 2; 6^2 = 13; 7^2 = 3; 8^2 = 18; 9^2 = 12; 10^2 =8; 11^2 = 6 =>простое
9. 41. Аналогично пункту а) является простым
10. 15. Является составным, так как 4^2 =1;
11. 35. Является составным, так как 6^2 = 1;
12. 561. Является составным, так как 67^2 = 1;
13. Провести тест Миллера-Рабина для чисел 23, 41, 15, 35, 561.
14. 23. 23 = 2 \* 11 + 1

Пусть a = 2 => 2^11 mod 23 = 1 => простое

1. 41. 41 = 2^3 \* 5 +1

Пусть a = 2 => 2^5 mod 41 = 32 =>идем дальше

2^(5\*2) mod 41 = 1024 mod 41 = 40 = -1 => простое

1. 15. 15 = 7\*2 + 1

Пусть a = 2 => 2^7 mod 15 = 128 mod 15 = 8 =>идем дальше

2^(7\*2) mod 15 = 64 mod 15 = 4 => составное

1. 35. 35 = 17\*2 + 1

Пусть a = 2 => 2^17 mod 35 = 32 => составное

1. 561. 561 = 2:4\*35 + 1

Пусть a = 2 => 2^35 mod 561 = 263 =>идем дальше

263^2 mod 561 = 166

166^2 mod 561 = 67

67^2 = 1 mod 561 => составное

1. Применить метод Ферма разложения на множители для чисел 483, 1207, 561, 1219
2. 483. Sqrt(483) = 21;

22 => 484 – 483 = 1 => a = 23 b =21 => 483 = 23 \* 21 =3 \* 7 \* 23

1. 1207. Sqrt(1207) = 34; 1207 = 71\*17

35 => 1225 – 1207 = 18

36 => 1296 – 1207 = 89

37 => 1369 – 1207 = 162

38 => 237; …; 44=>729 => 27 => a = 71 b = 17

1. 561. Sqrt(561) = 23; 561 = 33 \* 17 = 3 \* 11 \* 17

24 => 15; 25=>64 => a = 33, b = 17

1. 1219. Sqrt(1219) = 35; 1219 = 53 \* 23

35 => 6; 36 => 77; 37 => 150; 38=> 225 => a = 53 b = 23

1. Применить p-1 метод Поллара разложения на множители для чисел 483, 1207, 561, 1219
2. 483. 483 = 3 \* 7 \* 23

a = 2 B = 4

Нод(1,483) = 1; Нод(3,483) = 1; Нод(63,483) = 1; Нод(2^24 - 1,483) = 21;

1. 1207. 1207 = 71\*17

a = 2 B = 4

Нод(1,1207) = 1; Нод(2,1207) = 1; Нод(63,1207) = 1; Нод(2^24 - 1,1207) = 17;

1. 561. 561 = 3 \* 11 \* 17

A = 2 B = 8

Нод(1, 561) = 1; Нод(3,561) = 3; … Нод(2^8 – 1, 561) = 51;

1. 1219. 1219 = 23 \* 53

Нод(1, 1219) = 1; Нод(3, 1219) = 3; Нод(63, 1219) = 1; Нод(2^24 - 1, 1219) = 1; Нод(2^120 - 1, 1219) = 3; Нод(2^720 - 1, 1219) = 1; Нод(2^3760 - 1, 1219) = 1; Нод(2^362880 - 1, 1219) = 3; Нод(2^11! - 1, 1219) = 23;

1. Применить PO(Rho) метод Поллара разложения на множители для чисел 483, 1207, 561, 1219
2. 483. 483 = 3 \* 7 \* 23

X = 2 y = 2 Нод(2-2, 483) = 1

X = 5 Y = 26 НОД(21, 483) = 21 => 483 = 3 \* 7 \* 23

1. 1207. 1207 = 71\*17

X = 2 y = 2 Нод(2-2, 1207) = 1

X = 5 Y = 26 НОД(21, 1207) = 1

X = 26 Y = 458330 НОД(458304, 1207) = 1

X =1022 Y =87 => НОД(x – y, 1207) = 17

1. 561. 561 = 3 \* 11 \* 17

X = 2 Y = 2 НОД(0, 561) = 1

X = 5 Y = 26 НОД(21, 561) = 3

1. 1219. 1219 = 23 \* 53

X = 2 Y = 2 НОД(0, 1219) = 1

X = 5 Y = 26 НОД(21, 1219) = 1

…

X = 1205 Y = 1021 НОД(x-y, 1219) = 23

1. Применить квадратичное решето для разложения на множители для чисел 483, 1207, 561, 1219
2. 483. 483 = 3 \* 7 \* 23

S = 4, 5, 7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X | 0 | 1 | 2 | 3 |
| X^2 MOD 4 | 0 | 1 | 0 | 1 |
| Z = X^2 – 483 MOD 4 | 1 | 2 | 1 | 2 |
| S4 | 1 | 0 | 1 | 0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X | 0 | 1 | 2 | 3 | 4 |
| X^2 MOD 5 | 0 | 1 | 4 | 4 | 1 |
| Z = X^2 – 483 MOD 5 | 2 | 3 | 1 | 1 | 3 |
| S5 | 0 | 0 | 1 | 1 | 0 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| X | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| X^2 MOD 7 | 0 | 1 | 4 | 2 | 2 | 4 | 1 |
| Z = X^2 – 483 MOD 7 | 0 | 1 | 4 | 2 | 2 | 4 | 1 |
| S7 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| X | 22 | 23 |
| S4 | 1 | 1 |
| S5 | 1 | 1 |
| S7 | 1 | 1 |

22^2 – 483 = 1 => a = 23 b = 21 => 483 = 23 \* 3 \* 7

1. 1207.

S = 4, 5, 7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X | 0 | 1 | 2 | 3 |
| X^2 MOD 4 | 0 | 1 | 0 | 1 |
| Z = X^2 – 1207 MOD 4 | 1 | 2 | 1 | 2 |
| S4 | 1 | 0 | 1 | 0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X | 0 | 1 | 2 | 3 | 4 |
| X^2 MOD 5 | 0 | 1 | 4 | 4 | 1 |
| Z = X^2 – 1207 MOD 5 | 3 | 4 | 2 | 2 | 4 |
| S5 | 0 | 1 | 0 | 0 | 1 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| X | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| X^2 MOD 7 | 0 | 1 | 4 | 2 | 2 | 4 | 1 |
| Z = X^2 – 1207 MOD 7 | 4 | 5 | 1 | 6 | 6 | 1 | 5 |
| S7 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |

|  |  |  |
| --- | --- | --- |
| X | … | 44 |
| S4 | … | 1 |
| S5 | … | 1 |
| S7 | … | 1 |

44^2 – 1207 = 729 => a = 71 b = 17 => 1207 = 17 \* 71

Аналогично предыдущим примерам:

1. 561. 561 = 3 \* 11 \* 17
2. 1219. 1219 = 23 \* 53
3. Применить решето поля чисел для разложения на множители для чисел 483, 1207, 561, 1219
4. 483. 483 = 3 \* 7 \* 23

T =3 S = {2, 3, 5}

A,b: (2,4), (3,9), (5,25), (6,36)

B = 2^4 \* 3^4 \* 5^2

X = 2 \* 3 \*5 = 30

Y = 2^(4/2) \* 3^(4/2) \* 5^(2/2) = 180

НОД(150, 483) = 3 => 483 = 3 \* 7 \* 23

1. 1207 = 17 \* 71
2. 561 = 3 \* 11 \* 17
3. 1219 = 23 \* 53
4. Определить являются ли простыми числа Мерсена M23, M29, M31

M23 = 2^ 23 – 1 =8388607

D = 2\*23+1 = 47 => 8388607%47 = 0 – не простое

M29 = 2^29 – 1

D = 2 \* 4 \* 29 = 233 – делитель => не простое

M31 = 2^31 – 1 – простое

1. Приведите несколько примеров, чтобы показать, что если 2^n – 1 – простое число, то n – простое число.

2^2 – 1 =3; 2^3 -1 =7; 2^5 – 1 = 31; 2^7-1 = 127; 2^4 – 1 = 15 – не простое

Как использовать: если n – составное, то 2^n – 1 составное всегда => не надо проверять на простоту 2^n – 1

1. Определите, сколько из следующих целых чисел пройдут испытание Ферма на простоту чисел: 100, 110, 130, 150, 200, 250, 271, 341, 561. Использовать основание 2.
2. 2^99 mod 100 <> 1
3. 2^109 mod 110 = 72 <> 1
4. 2^129 mod 133 = 122 <> 1
5. 2^149 mod 150 = 62 <> 1
6. 2^199 mod 200 = 197 <> 1
7. 2^249 mod 250 = 62 <> 1
8. 2\*\*270%271 = 1 – прошло
9. 2^340 % 341 = 1 – прошло
10. 2:560 % 561 = 1 - прошло
11. Определите, сколько из следующих целых чисел пройдут испытание Миллера-Рабина на простоту чисел: 100, 109, 201, 271, 341, 349. Используйте основание 2.
12. 100 – нет
13. 109 = 1 + 27\*2^2

2^27 mod 109 = 33

33^2 mod 109 = 108

108^2 mod 109 = 1 – простое

1. 201 = 1 + 5^2 \* 2^3

2^25 mod 201 = 95

95^2 mod 201 = 181

181^2 mod 201 = 199

1. 271 = 1 + 2 \* 135

2^135 mod 271 = 1 - простое

1. 341 = 1 + 2^2 \* 85

2^85 mod 341 = 32

32^2 mod 341 = 1 => простое

1. 349 = 1 + 4\*87

2^87 mod 349 = 213

213^2 mod 349 = 348

348^2 = 1 => простое

1. Используйте рекомендованное испытание, чтобы определить, является ли любое из следующих целых чисел простым числом: 271, 3149, 9673..
2. 271

3, 5, 7, 11, 13 => на все не делится

Пусть a = 2 => 271 = 1+ 2 \* 135

2^135 mod 271 = 1 =>ок

Пусть a=3 => 3^135 mod 271 = 270 = -1 => не простое

1. 3149

3, 5, 7, 11, 13 => на все не делится

Пусть a = 2 => 271 = 1+ 2^2 \* 787

2^787 mod 3149 = 2523 =>не простое

1. 9673

3, 5, 7, 11, 13 => на все не делится

Пусть a = 2 => 9673 = 1+ 2^3 \* 1209

2^1209 mod 9673 = 2620 => не простое

1. Используйте a=2, *x*=3 и несколько простых чисел, чтобы показать, что если *p* — простое число, то выполняется следующее сравнение: (x−a)^p=(x^p−a)(mod p)

P – простое => (3-2)^p = (3^p – 2) mod p

P =2 1 = 1

P = 3 1 = 25 mod 3 = 1

P = 5 1 = (3^5 – 2) mod 5 = 1

P = 7 1 = (3^7-2) mod 7 = 1

1. Говорят, что n-ное простое число может быть приближенно вычислено  
   как p\_n=n\*ln(n). Проверьте это на нескольких простых числах.

P1 = 1 ln 1 = 0 (2)

P2 = 2 ln 2 = 1.38 (3)

P3 = 3 ln 3 = 3.29 (5)

P4 = 4 \* ln(4) = 5.54

P5 = 8 (11)

P10 = 23 (29)

Относительная ошибка между настоящим простым p\_n и формульным начинает уменьшаться

1. Используя свойства дискретных логарифмов, покажите, как решить сравнения:  
   a. X^5=11 mod 17

Пусть g - первообразный корень поля по модулю 17 => x^5 = 3^(5a) = 11 mod 17 => 5a = log3(11) => 5a = 7 mod 16 => a = 7\*5^-1 mod 16 = 11 =>

x = 3^11 mod 17 = 7 mod 17  
b. 2x^11=22 mod 19

x^11 = 3 \* 2^-1 mod 19 => x^11 = 3 \* 10 mod 19 => x^11 = 11 mod 19 => x^11 = 2^(11a) => 11a = log2(11) = 7 mod 18 => a = 7 \* 11^-1 mod 18 = 17 => x = 2^17 mod 19 = 14  
c. 5x^12+6x=8 mod 23

x^12 + 15x = 20 mod 23

Пусть g = 5=> 5^12a = 20 +8\*5^a =>

12a = log5(20+5^a\*8)

Честно сделано перебором: корней нет.

1. Пусть мы имеем компьютер, выполняющий операции со скоростью 1 миллион бит в секунду. Вы хотите затратить только 1 час на испытание простоты чисел. Какое наибольшее число вы можете проверить, используя следующие методы, проверяющие простоту чисел?
2. теория делимости  
   b. AKS-алгоритм  
   c. Ферма  
   d. извлечением квадратного корня  
   e. Миллера-Рабина

a. Теория делимости

Метод теории делимости предполагает проверку делимости числа на все простые числа до его квадратного корня. Это очень медленный метод для больших чисел.

* Оценка: Для числа *n* требуется проверить до *n*​ делителей.
* Время: Ограничено 1 часом (3600 секунд) при скорости 1 миллион операций в секунду.
* Максимальное число: Очень маленькое, порядка 10^12.

b. AKS-алгоритм

AKS-алгоритм является детерминированным полиномиальным алгоритмом проверки простоты.

* Оценка: Временная сложность O(log^6(n)).
* Время: Ограничено 1 часом.
* Максимальное число: Очень большое, порядка 10^18 и более.

c. Тест Ферма

Тест Ферма является вероятностным тестом, основанным на малой теореме Ферма.

* Оценка: Временная сложность O(klog^3(n)), где *k* — количество итераций.
* Время: Ограничено 1 часом.
* Максимальное число: Очень большое, порядка 10^20.

d. Извлечение квадратного корня

Метод извлечения квадратного корня предполагает проверку делимости числа на все простые числа до его квадратного корня.

* Оценка: Аналогично теории делимости, очень медленный для больших чисел.
* Время: Ограничено 1 часом.
* Максимальное число: Очень маленькое, порядка 10^12.

e. Тест Миллера-Рабина

Тест Миллера-Рабина является вероятностным тестом, который эффективен для больших чисел.

* Оценка: Временная сложность O(klog^3(n)), где *k* — количество итераций.
* Время: Ограничено 1 часом.
* Максимальное число: Очень большое, порядка 10^20.

1. Пусть мы имеем компьютер, выполняющий операции со скоростью 1 миллион бит в секунду. Вы хотите потратить только 1 час на разложение составного целого числа. Какое наибольшее число вы можете разложить на множители, используя следующие методы разложения на множители?
   1. проверка делением
   2. Ферма
   3. Полларда (РО)
   4. квадратичное решето
   5. решето поля чисел

a. Проверка делением

Метод проверки делением предполагает проверку делимости числа на все простые числа до его квадратного корня. Это очень медленный метод для больших чисел.

* Оценка: Для числа *n* требуется проверить до sqrt(*n)*​ делителей.
* Время: Ограничено 1 часом (3600 секунд) при скорости 1 миллион операций в секунду.
* Максимальное число: Очень маленькое, порядка 10^12.

b. Метод Ферма

Метод Ферма основан на представлении числа в виде разности квадратов.

* Оценка: Временная сложность O(n^(1/4)).
* Время: Ограничено 1 часом.
* Максимальное число: Очень маленькое, порядка 10^16.

c. Метод Полларда (РО)

Метод Полларда (РО) является вероятностным методом, эффективным для чисел с небольшими множителями.

* Оценка: Временная сложность O(n^(1/4)).
* Время: Ограничено 1 часом.
* Максимальное число: Очень маленькое, порядка 10^20.

d. Квадратичное решето

Квадратичное решето является одним из самых эффективных методов для разложения чисел средней величины.

* Оценка: Временная сложность O(e^sqrt(log n \* log log n)).
* Время: Ограничено 1 часом.
* Максимальное число: Очень большое, порядка 10^50.

e. Решето поля чисел

Решето поля чисел является самым эффективным методом для разложения очень больших чисел.

* Оценка: Временная сложность O(e^(log n)^1/3\*(log log n)^2/3)).
* Время: Ограничено 1 часом.
* Максимальное число: Очень большое, порядка 10^100.

1. Написать программу для решения квадратичных сравнений по составному модулю.

Программа решает квадратичное сравнение x^2≡a(mod n) для заданных чисел a и n.

Программа начинает с факторизации числа n на простые множители. Для этого используется метод Ферма, который эффективен для чисел, близких к квадрату. Функция fermat\_factorization(n) находит два множителя числа n, если они существуют. Функция factor(n) рекурсивно разлагает число n на простые множители, используя метод Ферма и проверку на простоту.

Затем для каждого простого множителя p числа n программа проверяет, является ли *a* квадратичным вычетом по модулю p. Это делается с помощью символа Лежандра. Функция legendre\_symbol(a, p) вычисляет символ Лежандра, который показывает, является ли a квадратичным вычетом по модулю p. Если a не является квадратичным вычетом по модулю хотя бы одного простого множителя p, то сравнение x^2≡a mod n не имеет решений.

Затем для каждого простого множителя p и его степени k программа решает квадратичное сравнение x^2≡a mod p^k. Если *p*=2, программа использует специальные правила для нахождения решений:

* + Для k=1 и k=2 решения находятся напрямую.
  + Для k≥3 решения существуют только если a≡1 mod8 , и в этом случае находятся 4 корня.

Если p≠2, программа использует метод Гензеля для подъема решений с модуля p до модуля p^k.

После нахождения решений для каждого простого множителя p^k программа объединяет их с помощью китайской теоремы об остатках.

Программа возвращает все возможные решения квадратичного сравнения x^2≡a mod n, отсортированные по возрастанию. Если решений нет, программа возвращает пустой список.