Задания СРС:

**Цель: у**глубление теоретических знаний и развитие практических навыков по обеспечению кибербезопасности, формирование компетенций в области защиты информации, освоение принципов функционирования систем защиты, а также анализ современных киберугроз и методов противодействия им.

**Лекция 4: Основные положения теории чисел, используемые в криптографии с открытым ключом**

### ****Цель:****

Освоить ключевые математические основы, лежащие в основе алгоритмов асимметричной криптографии, включая операции с простыми числами, остатками и модульной арифметикой.

### ****Задание:****

#### ****1. Теоретическая часть: изучить и законспектировать****

* Понятие делимости, простых чисел и их свойства
* Евклидов алгоритм и расширенный алгоритм Евклида
* Модульная арифметика: операции по модулю
* Обратные элементы по модулю
* Теорема Эйлера и функция Эйлера φ(n)
* Маленькая теорема Ферма
* Остаточные классы и китайская теорема об остатках (обзорно)

#### ****2. Практическая часть:****

* Вычислить:
  + НОД двух заданных чисел с помощью алгоритма Евклида
  + Обратный элемент по модулю (например, 7^(-1) mod 26)
  + φ(n) для нескольких значений n
* Привести пример использования этих операций в RSA (например, как выбирается e и d)

#### ****3. Подготовить:****

* Презентацию (8–10 слайдов) с краткими объяснениями каждого математического инструмента
* Мини-доклад (1 страница) о роли теории чисел в криптографии и примерах её применения

**Цель лекции:** напомнить читателю основные положения теории чисел, используемые в криптографии с открытым ключом

### Простые и составные числа

Каждое *натуральное число*, большее единицы, делится по крайней мере на два числа: на 1 и на само себя. Если число не имеет делителей, кроме самого себя и единицы, то оно называется **простым**, а если у числа есть еще делители, то **составным**. *Единица* же не считается ни простым числом, ни составным. Например, числа 7, 29 — простые; числа 9, 15 — составные ( 9 делится на 3, 15 делится на 3 и на 5 ).

Интересный факт: если два простых числа отличаются на 2, то их называют числами-"близнецами". Чисел-"близнецов" не очень много. Например, "близнецами" являются 5 и 7, 29 и 31, 149 и 151, а также 242 206 083\*238 880±1 (наибольшая найденная на момент написания учебного пособия пара "близнецов").

Не о всяком числе можно сразу сказать, простое оно или составное. Если число меньше ста, то, скорее всего мы сразу сможем ответить на этот вопрос. Однако с большими числами дело сложнее. Возьмем, например, число 2009. Простое оно или составное? Попробуем найти возможные делители этого числа среди первых простых чисел. 2009 определенно не делится на 2 (так как оно нечетное), на 3 (так как сумма его цифр 2+9=11 не делится на 3 ), на 5. А вот, попробовав разделить 2009 на 7, мы увидим, что в результате получается *целый* результат – 287. Таким образом, получен ответ: число 2009 – составное. В данном случае ответ получен достаточно быстро. Бывает, что проверка на простоту производится гораздо дольше, а для работы с большими целыми числами требуются даже специальные компьютерные программы.

*Поиск* больших простых чисел имеет важное *значение* для математики и не только. Например, в криптографии большие простые числа используются в алгоритмах шифрования с открытым ключом. Для обеспечения надежности шифрования там используются простые числа длиной до 1024 *бит*.

Перемножить два числа сравнительно нетрудно, особенно если у нас есть калькулятор, а числа не слишком велики. Существует и обратная задача – *задача факторизации* – нахождение двух или более чисел, дающих при перемножении заданное число. Эта задача гораздо труднее, чем перемножение чисел, и любому, кто пытался ее решить, об этом известно. Например, если от нас требуется умножить 67 на 113, то результат, 7571, будет получен, наверно, меньше чем за минуту. Если же от нас требуется найти два числа, *произведение* которых равно 7571, то, скорее всего, это займет у нас гораздо больше времени.

*Поиск* сомножителей числа n может вестись, например, перебором всех простых чисел до Описание: \sqrt n, как в рассмотренном выше примере с числом 2009. Однако, если множители – большие простые числа, то на их *поиск* уйдет достаточно много времени.

Таким образом, факторизация большого числа требует значительных затрат времени даже в том случае, когда известно, что оно является произведением двух больших простых чисел.

Сложность задачи факторизации используется в некоторых криптографических алгоритмах, например, в системе шифрования *RSA*.

### Основная теорема арифметики

Любое составное число можно составить из некоторого количества простых с помощью умножения. Например, составное число 2009 можно получить так:

2009 = 7 \* 7 \* 41

В математике рассматривается так называемая **основная теорема арифметики**, которая утверждает, что любое *натуральное число* ( n>1 ) либо само является простым, либо может быть разложено на *произведение* простых делителей, причем единственным способом (если не обращать внимания на порядок следования сомножителей).

Воспользовавшись обозначением степени, разложение числа 2009 на простые множители можно записать так:

2009 = 72 \* 41

Разложение на множители называется **каноническим**, если все множители являются простыми и записаны в порядке возрастания.

Например, запишем *каноническое разложение* числа 150 на множители:

150 = 2 \* 3 \* 52

### Взаимно простые числа и функция Эйлера

Два числа называются взаимно простыми, если они не имеют ни одного общего делителя кроме единицы.

Например, числа 11 и 12 взаимно просты (у них нет общих делителей кроме единицы), числа 30 и 35 — нет (у них есть общий делитель 5 ).

Исследованием закономерностей, связанных с целыми числами, долго занимался швейцарский математик Леонард Эйлер (Leonard Euler). Одним из вопросов, которым он интересовался, был следующий: сколько существует натуральных чисел, не превосходящих n и взаимно простых с n? Ответ на этот вопрос был получен Эйлером в 1763 году и этот ответ связан с каноническим разложением числа n на простые множители. Так, если

Описание: n=p_1^{a_1}\times p_2^{a_2}\times \dots \times p_n^{a_n}

где p1, p2, ..., pn – разные простые множители, то число Описание: \phi натуральных чисел, не превосходящих n и взаимно простых с n можно точно определить по формуле

Описание: \varphi (n) = n \times \left ( 1- \frac{1}{p_1} \right ) \times \left ( 1- \frac{1}{p_2} \right ) \times \dots \times \left ( 1- \frac{1}{p_n} \right )

Число натуральных чисел, не превосходящих n и, взаимно простых с n, называется **функцией Эйлера** и обозначается Описание: \phi(n).

Например, найдем количество натуральных чисел, не превосходящих 12 и взаимно простых с 12. Из ряда натуральных чисел

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

взаимно простыми (не имеющими общих делителей) с 12 будут только числа 1, 5, 7, 11. Их количество равно четырем. Таким образом Описание: \phi(12) = 4.

Теперь попробуем подсчитать

Описание: \phi(12)

по формуле, предложенной Эйлером. Для этого вначале запишем *каноническое разложение* числа 12:

12 = 22 \* 3.

Теперь подсчитаем функцию Эйлера Описание: \phi(12):

Описание: \varphi (12) = 12 \times \left ( 1- \frac{1}{2} \right ) \times \left ( 1- \frac{1}{3} \right ) = 4 \times 3 \times \frac{1}{2} \frac{2}{3} = 4

Значения, вычисленные путем простого перебора взаимно простых чисел и по формуле Эйлера, совпали. Это неудивительно, так как формула для вычисления функции Эйлера может быть доказана строго математически.

Формулу Эйлера удобно использовать для больших n, если известно разложение числа n на простые множители. Для криптографии формула Эйлера важна тем, что она позволяет легко получить число

Описание: \phi(n)

для простых и некоторых других чисел. В криптографии используются два следующих следствия формулы Эйлера.

**Следствие 1**. Если p – *простое число*, то Описание: \phi(p) = p - 1.

Действительно, если p – *простое число*, то его *каноническое разложение* состоит только из него самого. Тогда

Описание: \varphi (p) = p \times \left ( 1- \frac{1}{p} \right ) = \frac{p(p-1)}{p} = p-1

**Следствие 2**. Пусть р и q — два различных простых ( Описание: p\ne q). Тогда

Описание: \phi (p*q) = (p-1)(q-1)

Эта формула объясняется следующим образом. Пусть р \* q = N, где р и q — два различных простых ( Описание: p\ne q ). Тогда

Описание: \varphi (p \times q) = \varphi (N)=N \times \left ( 1- \frac{1}{p} \right ) \times \left ( 1- \frac{1}{q} \right )= \frac{p \times q \times (p-1) \times (q-1)}{p \times q} = (p-1) \times (q-1)

Рассмотрим несколько примеров использования следствий формулы Эйлера.

**Пример 1**. Найдем Описание: \phi(13). 13 – *простое число*, значит, используя следствие 1 Описание: \phi(13) = 13 – 1 =12. Мы можем проверить себя (и Эйлера), выписав все числа, меньшие 13:

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12

и подсчитав все взаимно простые с ним. Их действительно 12.

**Пример 2**. Найдем Описание: \phi(35). 35 – составное число, значит, первое следствие нам не подходит. Однако 35 является произведением двух простых чисел: 35 = 5 \* 7. Используя следствие 2, вычисляем Описание: \phi(35):

Описание: \phi (35) = (5 – 1) * (7 - 1) = 4 * 6 = 24.

Проверяем, выписывая все числа, меньшие 35 и не имеющие с ним общих делителей:

1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18, 19, 22, 23, 24, 26, 27, 29, 31, 32, 33, 34.

Их действительно оказалось 24. На последнем примере видно, что использовать формулу Эйлера гораздо удобнее, чем рассматривать все числа из довольно большого диапазона и проверять на взаимную простоту.

### Арифметика остатков и теория сравнений

Немецкий математик Карл Фридрих Гаусс предложил *запись*

Описание: a \equiv b (mod\ m),

для двух чисел a и b, если они имеют одинаковые остатки от деления на m (читается a сравнимо с b по модулю m ). Например,

Описание: 1997 \equiv 1(mod 4),\\
7k +1 \equiv 1(mod 7),\ где\ k - любое\ целое\ число.

Сравнения обнаруживают полезные для математиков и криптографов свойства, во многом похожие на свойства равенств. Эти свойства позволяют существенно упрощать арифметические вычисления, если нас интересует только *остаток от деления* на некоторое число m. Так, например, свойства сравнений полезны при расчетах в алгоритмах шифрования с открытым ключом.

Простейшими свойствами сравнений являются следующие.

**Свойство 1**. Если a-b делится на m, то Описание: a \equiv b (mod\ m).

Например, Описание: 15 \equiv 1 (mod\  7), так как 15 -1 =14, а 14 кратно 7.

**Свойство 2**. Если

Описание: a \equiv b (mod\ m)

и

Описание: c \equiv d (mod\ m)

то

Описание: a+c \equiv b+d (mod\ m)\\
 ac \equiv bd (mod\ m)

Например, так как Описание: 13\equiv5(mod\ 8) и Описание: 11\equiv3(mod\ 8), то Описание: 13+11\equiv5+3\equiv0(mod\ m), а также Описание: 13*11\equiv5*3\equiv7(mod\ 8).

**Свойство 3**. Если

Описание: a \equiv b (mod\ m),

то

Описание: то\ a^k \equiv b^k (mod\ m), k \subset  N.

Например, так как

Описание: 25\equiv4(mod\ 7),

то

Описание: 25^{40} \equiv 4^{40} \equiv (4^2)^{20} \equiv 2^{20} \equiv (2^4)^5 \equiv 2^5 \equiv 4 (mod\ m)

**Свойство 4**. Если

Описание: ac \equiv bc (mod\ m)

и c взаимно просто с m, то

Описание: a \equiv b (mod\ m)

Например, известно, что

Описание: 1200 \equiv 45 (mod\ 7),

а так как

Описание: 1200=15*80\ и\ 45 =15*3, то\\
80 \equiv 3 (mod\ 7)

**Свойство 5**. Если

Описание: ac \equiv bc (mod\ mc),

то

Описание: a \equiv b (mod\ m)

Например, если

Описание: 44 \equiv 4 (mod\ 4(10),

то

Описание: 11 \equiv 1 (mod\ 10).

### Малая теорема Ферма

В основе алгоритма шифрования по системе *RSA* лежит теорема, сформулированная в начале семнадцатого столетия без доказательства французским математиком Пьером Ферма (Pierre Fermat). Её часто называют "Малой *теоремой Ферма*", и её не следует путать с известной "Великой *теоремой Ферма*" - её он также сформулировал без доказательства, а доказана она была только в 1993-94 годах. Леонард Эйлер в 1760 году опубликовал *доказательство* Малой теоремы Ферма и получил ее *обобщение*, известное под названием теоремы Ферма-Эйлера. Именно эта теорема используется в алгоритме зашифрования/расшифрования *RSA*.

**Малая теорема Ферма** формулируется следующим образом. Если p - *простое число*, а m - любое число, которое не делится на p, то

Описание: m^{p-1} \equiv 1(mod\ p),

то есть число mp-1 при делении на p дает *остаток* 1.

Например, пусть р=11, m = 3. Проверим, будет ли 310 mod 11 равно одному:

310 mod 11=32( ((32)2)2mod 11)) = 9(42 mod 11 )= 144 mod 11=1

*Обобщение*, сформулированное и доказанное Эйлером, справедливо для любого модуля, но в системе *RSA* используется частный случай, когда *модуль* является произведением только двух различных простых чисел. Поэтому рассмотрим формулировку теоремы для этого случая.

**Теорема Ферма-Эйлера** (для случая системы *RSA*). Если p и q - два различных простых числа, а m - любое число, которое не делится на p и q, то

Описание: m^{(p-1)(q-1)} \equiv 1(mod\ pq).

Например, пусть р=11, q = 5 (pq = 55), m = 3. Проверим, будет ли

Описание: 3^{40} \equiv 1(mod\ 55)

равно одному:

340 mod 55 = (35) 4mod 55 = 234 mod 55 = 279841 mod 55 = 1.

### Наибольший общий делитель

Пусть а и b — два целых положительных числа. Наибольший общий делитель чисел а и b есть наибольшее число с, которое делит и а, и b:

с = НОД(a, b).

Например, НОД(25,35) =5.

Для нахождения наибольшего общего делителя можно использовать следующий *алгоритм*, известный как **алгоритм Евклида**.

Алгоритм NOD (целые a, b, c);

Начало

1. Пока a<>b выполнять:

1.1.Если a>b то a:=a-b, иначе b:= b-a;

2. c:=a;

Конец.

После выполнения алгоритма результат будет содержаться в переменной с.

Поcмотрим, как с помощью алгоритма Евклида вычисляется НОД(18,9):

a: 18 9

b: 9 9

c: 9 9

Здесь каждый столбец представляет собой очередную итерацию алгоритма. Процесс продолжается до тех пор, пока b не станет равным a. Тогда в переменную с записывается ответ, в данном случае 9. Это и будет *значение* НОД(18,9).

### Обобщенный алгоритм Евклида

Для многих криптографических систем, рассмотренных в данной книге, актуален так называемый *обобщенный алгоритм Евклида*, с которым связана следующая теорема.

**Теорема**. Пусть а и b — два целых положительных числа. Тогда существуют целые (не обязательно положительные) числа х и у, такие, что

ах + by = НОД(a,b).

Обобщенный *алгоритм* Евклида служит для отыскания НОД(a, b) и х, у, удовлетворяющих записанному выше уравнению. Введем три строки U = (u1,u2,u3), V = (v1,v2,v3) и T = (t1,t2,t3).

*Алгоритм* записывается следующим образом (во входных параметрах должно соблюдаться условие a>=b ).

Алгоритм OAE (целые a, b);

Начало

1. U = (a,1,0), V = (b,0,1).

2. Пока v1 <> 0 выполнять:

2.1. q= u1 div v1;

2.2. T=(u1 mod v1, u2 – qv2, u3 – qv3);

2.3. U=V, V=T.

3. U=(НОД(a,b),x,y)).

Конец.

После окончания алгоритма результат будет содержаться в строке U. Операция div в алгоритме — это операция целочисленного деления.

**Пример**. Пусть а = 18, b = 9. Найдем числа х и у, удовлетворяющие уравнению

18х +9y = НОД(18,9).

Выполним *алгоритм* по шагам:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| q | U | | | V | | | T | | |
| u1 | u2 | u3 | v1 | v2 | v3 | t1 | t2 | t3 |
| 18 div 9 = 2 | 18 | 1 | 0 | 9 | 0 | 1 | 18 mod 9 = 0 | 1 - 2\*0 = 1 | 0 - 2\*1 = -2 |
|  | 9 | 0 | 1 | 0 | 1 | -2 |  |  |  |

В результате получили U=(НОД(a,b),x,y))= (9,0,1).

Выполним проверку: 18\*0 +9\*1 = 9 = НОД(18,9).

### Инверсия по модулю m

Во многих задачах криптографии для заданных чисел с, m требуется находить такое число d < m, что

cd mod m = 1

Такое d существует тогда и только тогда, когда числа с и m взаимно простые. Число d, удовлетворяющее равенству cd mod m = 1, называется **инверсией** с **по модулю** m и часто обозначается с-1 mod m. Данное обозначение для инверсии связано с тем, что *равенство* cd mod m = 1 можно переписать в виде

cс-1 mod m = 1.

Таким образом, *умножение* на с-1 соответствует делению на с при вычислениях по модулю m.

Инверсию по модулю m также можно вычислять с помощью обобщенного алгоритма Евклида.

Покажем, как это делается. *Равенство*, приведенное ниже означает, что для некоторого целого k имеет *место* *равенство* cd – km = 1. Учитывая, что с и d взаимно просты, можно преобразовать это *равенство* следующим образом:

m(-k) + cd = НОД(m,c).

Значит, мы можем вычислить с-1 mod m (или найти число d ) с помощью обобщенного алгоритма Евклида. При этом *значение* переменной k нас не интересует. Если число d получается отрицательным, то нужно прибавить к нему m, так как по определению число a mod m берется из *множества* {0,1,..., m - 1}.

Рассмотрим **пример**. Пусть m=9, c=5. Найдем 5-1 mod 9. Будем выполнять вычисления по обобщенному *алгоритму Евклида*, записывая все вычисления по шагам.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| q | U | | | V | | | T | | |
| u1 | u2 | u3 | v1 | v2 | v3 | t1 | t2 | t3 |
| 9 div 5 = 1 | 9 | 1 | 0 | 5 | 0 | 1 | 9 mod 5 = 4 | 1 - 1\*0 = 1 | 0-1\*1 = -1 |
| 5 div 4 =1 | 5 | 0 | 1 | 4 | 1 | -1 | 5 mod 4=1 | 0-1\*1=-1 | 1-1\*(-1)=2 |
| 4 div 1 =4 | 4 | 1 | -1 | 1 | -1 | 2 | 4 mod 1=0 | 1-4\*(-1)=5 | -1-4\*2=-9 |
|  | 1 | -1 | 2 | 0 | 1 | 7 |  |  |  |

Таким образом, получили, что 5-1 mod 9 = 2. Проверим: 5\*2 mod 9 =10 mod 9 = 1.

### Ключевые термины

**Алгоритм Евклида** – математический *алгоритм*, который может использоваться для нахождения наибольшего общего делителя двух чисел.

**Взаимно простые числа** – числа, не имеющие общих делителей (кроме единицы).

**Задача факторизации** – нахождение двух или более натуральных чисел, дающих при перемножении заданное число.

**Инверсия по модулю** – такое *натуральное число*, которое при умножении по модулю на данное число дает в результате единицу.

**Каноническое разложение на множители** – такое разложение на множители, при котором все множители являются простыми и записаны в порядке возрастания.

**Малая теорема Ферма** – известная теорема, сформулированная П. Ферма, лежащая в основе алгоритма шифрования по системе *RSA*

**Наибольший общий делитель** чисел а и b – наибольшее число с, которое делит и а и b: с = НОД(a, b).

**Основная теорема арифметики** – теорема утверждающая, что любое *натуральное число* большее единицы либо само является простым, либо может быть разложено на *произведение* простых делителей, причем единственным способом (если не обращать внимания на порядок следования сомножителей).

**Простое число** – *натуральное число*, которое не имеет делителей, кроме самого себя и единицы.

**Составное число** – *натуральное число*, которое делится, помимо самого себя и единицы, еще хотя бы на одно число.

**Функция Эйлера** позволяет подсчитать число натуральных чисел, не превосходящих n и, взаимно простых с n. Обозначается

Описание: \phi(n).

### Краткие итоги

Криптографические методы с открытым ключом, в большинстве своем, основаны на операциях с большими целыми числами.

Каждое *натуральное число*, большее единицы, делится по крайней мере на два числа: на 1 и на само себя. Если число не имеет делителей, кроме самого себя и единицы, то оно называется простым, а если у числа есть еще делители, то составным. *Единица* же не считается ни простым числом, ни составным. Для больших чисел задача проверки на простоту может быть достаточно сложной. Также вычислительно сложной является задача факторизации – нахождение двух или более натуральных чисел, дающих при перемножении заданное число.

Два числа называются взаимно простыми, если они не имеют ни одного общего делителя кроме единицы.

Наибольший общий делитель чисел а и b – наибольшее число с, которое делит и а и b. Для нахождения наибольшего общего делителя можно использовать *алгоритм* Евклида.

*Инверсия* по модулю – такое *натуральное число*, которое при умножении по модулю на данное число дает в результате единицу. Инверсию по модулю m можно вычислять с помощью так называемого обобщенного алгоритма Евклида.

#### Вопросы для самопроверки

1. Дайте определение простого и составного числа. Приведите по три примера простых и составных чисел.
2. Дать определение понятия "взаимно простые числа". Привести примеры взаимно простых чисел и чисел, не являющихся взаимно простыми.
3. Сформулируйте основную теорему арифметики.
4. В чем заключается задача факторизации?
5. Дайте определение наибольшего общего делителя.
6. Сформулируйте алгоритм Евклида для нахождения наибольшего общего делителя двух чисел.
7. Сформулируйте малую теорему Ферма.
8. Сформулируйте *теорема Ферма*-Эйлера (для случая системы RSA).
9. Сформулируйте обобщенный (расширенный) алгоритм Евклида.
10. Сформулировать принципы выполнения "операции взятия по модулю". Приведите примеры выполнения этой операции и поясните их.
11. Что такое инверсия по модулю n?