НАО «Восточно-Казахстанский технический университет имени Д. Серикбаева»

Дисциплина «Теоретико-числовые методы в криптографии»

ПРАКТИЧЕСКАЯ РАБОТА 4

СРАВНЕНИЯ И КЛАССЫ ВЫЧЕТОВ

Методические указания и задание к выполнению

Аудиторная нагрузка: 3 академических часа. Сначала изучите методичку и разобранный пример, затем выполните свой вариант. Свойства взаимно простых чисел из лекции 4 используются для обоснования сокращения; определения сравнений и классов вычетов приведены ниже.

1 Цель работы

Научиться переходить от целых чисел к их классам вычетов, вычислять остатки выражений и обосновывать преобразования сравнений. Установить, когда можно сокращать общий множитель, как меняется модуль и почему одинаковые остатки не означают равенства исходных чисел.

2 Планируемые результаты

Студент умеет находить неотрицательные и симметричные представители класса, применять сложение и умножение сравнений, вычислять остаток многочлена, проверять полную систему вычетов и различать применение НОД и НОК при работе с разными модулями. Программная проверка сопровождается математическим обоснованием.

3 Программное обеспечение

Python 3.8 или новее. Используется стандартная функция `gcd` из модуля `math`; дополнительные пакеты не нужны. Все блоки Python из раздела 5 разместите последовательно в одном файле `practical04.py` и выполните сверху вниз. В командной строке используйте `python practical04.py` или `python3 practical04.py` в зависимости от настройки системы.

Аргументы функций — целые числа, модули — положительные. Для индивидуальных вариантов $m, n \geq 2$; модуль 1 отдельно используется для проверки граничного случая. Дробное деление `/` в вычислениях остатков и новых модулей не применяется.

4 Методичка с разобранными примерами

4.1 Сравнимость и представители класса

Для положительного целого m запись $a \equiv b \pmod{m}$ означает, что m делит разность $a - b$. Равносильно: числа дают одинаковый неотрицательный остаток при делении на m. **Класс вычетов** — всё множество

$$[a]_m = \{a + km : k \in \mathbb{Z}\}.$$

Класс бесконечен; его представитель — одно выбранное число. Всего классов ровно m, и их можно представить числами $0, 1, \ldots, m-1$. Отношение сравнимости рефлексивно, симметрично и транзитивно: это следует из делимости разностей и объясняет разбиение на классы.

Пример. Для $m = 12$ имеем $-17 = 12(-2) + 7$, поэтому $[-17]_{12} = [7]_{12} = [19]_{12}$. Но 10 относится к другому классу, поскольку $-17 - 10 = -27$ не делится на 12.

Кроме неотрицательного представителя r, удобно выбрать симметричный s из интервала $-m/2 < s \leq m/2$. Сначала вычислите $r = a \bmod m$, затем положите $s = r - m$, если $2r > m$, и $s = r$ иначе. При чётном m из двух равноудалённых представителей выбираем **положительный** $m/2$.

Для $a = -17$, $b = 10$, $m = 12$ получаем $(r_a, s_a) = (7, -5)$, $(r_b, s_b) = (10, -2)$. Граница проверяется целыми числами через $2r > m$, без вычисления вещественной половины модуля. Для $m = 1$ существует один класс всех целых с представителем 0.

4.2 Допустимые арифметические действия

Если $a \equiv b \pmod m$ и $c \equiv d \pmod m$, то

$$a + c \equiv b + d \pmod m, \qquad a - c \equiv b - d \pmod m, \qquad ac \equiv bd \pmod m.$$

Для суммы и разности утверждения следуют из делимости $(a - b) \pm (c - d)$. Для произведения используем $ac - bd = a(c - d) + d(a - b)$. Отсюда индукцией получаем допустимость одинаковых неотрицательных целых степеней и подстановки сравнимых аргументов в многочлен с целыми коэффициентами.

Это не разрешает произвольное деление сравнений или сокращение показателя степени. Например, из совпадения квадратов нельзя автоматически получить совпадение самих чисел: $1^2 \equiv 5^2 \pmod{12}$, но $1 \not\equiv 5 \pmod{12}$.

4.3 Остаток многочлена

В индивидуальной работе вычисляется остаток выражения

$$F(a, b) = 3a^4 - 2a^2b + 5b^3 - 7.$$

Для образца $a = -17, b = 10, m = 12$ можно взять симметричные представители -5 и -2. Тогда $a^2 \equiv 25 \equiv 1$, $a^4 \equiv 1$, $b^3 \equiv -8 \equiv 4 \pmod{12}$. Следовательно,

$$F(a, b) \equiv 3 \cdot 1 - 2 \cdot 1 \cdot (-2) + 5 \cdot 4 - 7 = 20 \equiv 8 \pmod{12}.$$

Итоговый ответ — неотрицательный остаток 8. Полное вычисление исходного выражения даёт 249776; деление на 12 также оставляет 8. Для больших аргументов последовательное уменьшение промежуточных результатов сохраняет ответ и ограничивает их размер.

4.4 Сокращение общего множителя

Точное правило. Для $g = \gcd(c, m)$ и любого целого c справедливо

$$ca \equiv cb \pmod m \quad \Leftrightarrow \quad a \equiv b \pmod{m/g}.$$

Обоснование. Запишем $c = gc_1, m = gm_1$, где $\gcd(c_1, m_1) = 1$. Условие $m \mid c(a - b)$ равносильно $m_1 \mid c_1(a - b)$, а по взаимной простоте — $m_1 \mid (a - b)$. Обратное направление получается умножением на c.

Сохранить исходный модуль при сокращении можно как общее правило, если $g = 1$. При $g > 1$ исходное сравнение гарантирует только новый модуль m/g; дополнительное совпадение по m у конкретных чисел следует проверять отдельно.

Пример. Пусть $c = 8, m = 12, a = -17, b = 10$. Разность $8a - 8b = -216$ делится на 12. НОД равен 4, поэтому после сокращения получаем $-17 \equiv 10 \pmod 3$. По модулю 12 эти числа не сравнимы.

Универсальный контрпример неверному сокращению при $g > 1$: взять $a_0 = 0, b_0 = m/g$. Тогда $ca_0 \equiv cb_0 \pmod m$, но $a_0 \not\equiv b_0 \pmod m$, поскольку $0 < m/g < m$.

Если общий положительный множитель присутствует также в модуле, его можно сократить во всей записи: $da \equiv db \pmod{dm}$ равносильно $a \equiv b \pmod m$. Это правило отличается от удаления множителя только из двух частей сравнения.

4.5 Полная система вычетов и повторения

Полная система вычетов по модулю m содержит по одному представителю каждого класса. Поэтому для набора из m чисел достаточно и необходимо, чтобы все их остатки были различными.

Рассмотрим набор $T_j = cj + s$, где $j = 0,1,\dots,m-1$. Он является полной системой тогда и только тогда, когда $\gcd(c,m) = 1$.

Доказательство и более точный результат. Совпадение остатков $ci + s$ и $cj + s$ равносильно $c(i-j) \equiv 0 \pmod m$, то есть $i \equiv j \pmod{m/g}$, где $g = \gcd(c,m)$. Среди m индексов имеется m/g таких групп, по g индексов в каждой. Следовательно, различных остатков ровно m/g, и каждый встречается g раз. При $g = 1$ повторений нет; при $g > 1$ уже индексы 0 и m/g дают совпадение.

Для $c = 8, s = -17, m = 12$ имеем $g = 4$ и всего три разных остатка:

Индексы j	**Остаток** $T_j \bmod 12$	**Число появлений**
0, 3, 6, 9	7	4
1, 4, 7, 10	3	4
2, 5, 8, 11	11	4

Таким образом, набор не является полной системой. При том же модуле коэффициент 5, взаимно простой с 12, даёт полную систему при любом целом сдвиге s. Это свойство объясняет требование взаимной простоты коэффициента в обратимых аффинных преобразованиях классов.

4.6 Сравнения по разным модулям

Одна пара чисел и два модуля. Условия $u \equiv v \pmod m$ и $u \equiv v \pmod n$ вместе равносильны сравнению по $L = \operatorname{lcm}(m,n)$. Разность делится на оба модуля ровно тогда, когда делится на их НОК. Произведение mn можно использовать как общий гарантированный модуль только при $\gcd(m,n) = 1$.

Для $m = 12, n = 18$ имеем $L = 36$. Пара $u = -17, v = 19$ сравнима по 12, 18 и 36, но не по $12 \cdot 18 = 216$. В общем случае пример $u = a$, $v = a + L$ показывает, почему без взаимной простоты произведение слишком велико.

Разные пары чисел. Если $u \equiv a \pmod m$, $v \equiv b \pmod n$, то сумму и произведение можно сравнивать по общему делителю $G = \gcd(m,n)$:

$$u + v \equiv a + b \pmod G, \qquad uv \equiv ab \pmod G.$$

Сначала оба исходных сравнения переходят к модулю G, затем применяют правила раздела 4.2. Например, $-5 \equiv -17 \pmod{12}$ и $28 \equiv 10 \pmod{18}$ дают $23 \equiv -7 \pmod 6$. По модулю 36 последнее сравнение неверно: разность равна 30.

Вообще, из сравнения по модулю m следует сравнение по каждому положительному делителю m. Переход к кратному модулю требует дополнительных условий.

5 Программные примеры

5.1 Представители и проверка сравнимости

Функция `residue_data` возвращает пару `(r, s)` согласно принятому интервалу $-m/2 < s \le m/2$. Функция `congruent` возвращает логический признак делимости разности. Общая проверка исключает нулевые и отрицательные модули.

```
from math import gcd

def check_modulus(m):
    if m < 1:
        raise ValueError("Модуль должен быть положительным")

def residue_data(a, m):
    check_modulus(m)
    r = a % m
    s = r - m if 2 * r > m else r
    return r, s

def congruent(a, b, m):
    check_modulus(m)
    return (a - b) % m == 0
```

5.2 Вычисление многочлена по модулю

Промежуточные квадраты и куб вычисляются целыми числами, после умножений берётся остаток. Выходное значение всегда принадлежит диапазону от 0 до $m - 1$.

```
def polynomial_mod(a, b, m):
    check_modulus(m)
    a, b = a % m, b % m
    a2 = a * a % m
    a4 = a2 * a2 % m
    b2 = b * b % m
    b3 = b2 * b % m
    return (3 * a4 - 2 * a2 * b + 5 * b3 - 7) % m
```

5.3 Проверка сокращения

Результат содержит НОД, новый модуль и истинность сравнений до и после правильного сокращения. Последние два значения обязаны совпадать, в том числе когда исходное сравнение ложно. Они не выражают сравнимость a, b по прежнему модулю.

```
def cancellation_data(c, a, b, m):
    check_modulus(m)
    g = gcd(c, m)
    reduced = m // g
    original = congruent(c * a, c * b, m)
    after = congruent(a, b, reduced)
    return g, reduced, original, after
```

5.4 Анализ полной системы

Возвращаются упорядоченный список остатков, словарь их частот и признак полноты. При вызове для своего варианта в качестве сдвига передавайте `a`.

```
def affine_residues(c, shift, m):
    check_modulus(m)
    values = [(c * j + shift) % m for j in range(m)]
    counts = {}
    for r in values:
        counts[r] = counts.get(r, 0) + 1
    full = len(counts) == m
    return values, counts, full
```

Словарь показывает, сколько раз появился каждый класс. Для проверки результата сопоставьте число ключей с m/g, а каждую частоту — с $g = \gcd(c, m)$.

5.5 Общий модуль для одной пары чисел

Используем формулу НОК через НОД. Деление выполняется до умножения и является точным.

```
def common_modulus(m, n):
    check_modulus(m)
    check_modulus(n)
    return (m // gcd(m, n)) * n
```

Для разных пар чисел из раздела 4.6 используйте `gcd(m, n)` как модуль проверки суммы и произведения. Функция `common_modulus` вычисляет НОК для случая, когда оба сравнения относятся к одной паре.

5.6 Запуск разобранного примера

После всех определений добавьте следующий блок. Для своего варианта замените значения в первой строке. Списки с остатками нужно сопоставить с ручными расчётами, а не только скопировать в отчёт.

```
m, n, a, b, c = 12, 18, -17, 10, 8
print("Представители a:", residue_data(a, m))
print("Представители b:", residue_data(b, m))
print("a и b сравнимы:", congruent(a, b, m))
print("Остаток F:", polynomial_mod(a, b, m))
print("Сокращение:", cancellation_data(c, a, b, m))
values, counts, full = affine_residues(c, a, m)
print("Вычеты:", values)
print("Частоты:", counts)
print("Полная система:", full)
L = common_modulus(m, n)
u, v = a, a + L
print("НОК и пара:", L, u, v)
print("Проверки модулей:",
      [congruent(u, v, q) for q in (m, n, L, m * n)])
```

Ожидаемый вывод

```
Представители a: (7, -5)
Представители b: (10, -2)
a и b сравнимы: False
Остаток F: 8
Сокращение: (4, 3, True, True)
Вычеты: [7, 3, 11, 7, 3, 11, 7, 3, 11, 7, 3, 11]
Частоты: {7: 4, 3: 4, 11: 4}
Полная система: False
НОК и пара: 36 -17 19
Проверки модулей: [True, True, True, False]
```

В строке «Сокращение» указаны $g = 4$, модуль 3 и два истинных сравнения. Это согласуется с тем, что строка «a и b сравнимы» для исходного модуля 12 содержит `False`. В последней строке модули идут в порядке m, n, L, mn.

5.7 Проверки граничных случаев

Добавьте блок ниже после демонстрации. При успешном выполнении появится строка `Все проверки пройдены`. Числа в этих общих проверках при выборе своего варианта не меняют.

```
assert residue_data(-17, 12) == (7, -5)
assert residue_data(6, 12) == (6, 6)
assert residue_data(999, 1) == (0, 0)
assert congruent(-17, 19, 12)
assert not congruent(-17, 10, 12)
assert polynomial_mod(-17, 10, 12) == 8
assert cancellation_data(8, -17, 10, 12) == (4, 3, True, True)
assert cancellation_data(0, 2, 7, 12) == (12, 1, True, True)
assert affine_residues(5, 1, 12)[2]
assert not affine_residues(8, -17, 12)[2]
assert common_modulus(12, 18) == 36
assert common_modulus(8, 9) == 72
print("Все проверки пройдены")
```

Отдельный вызов с модулем 0 или отрицательным модулем должен вызвать `ValueError`. Случай `c = 0` разрешён: НОД равен m, после сокращения получается модуль 1, а аффинный набор при $m > 1$ содержит только один класс.

6 Порядок выполнения работы

1. Изучить раздел 4, проверить разобранный пример и объяснить каждое преобразование.
2. Получить номер варианта, выписать m, n, a, b, c из раздела 8.
3. Выполнить задания 1–5 вручную с обоснованиями и промежуточными остатками.
4. Собрать код, выполнить задание 6, проверить общие граничные случаи.
5. Сопоставить расчёты, оформить отчёт и подготовиться к защите. Ориентир на 135 минут: разбор — 20, ручные вычисления — 40, работа с системами вычетов и модулями — 30, программная проверка — 30, обсуждение — 15.

7 Индивидуальные задания

Задание 1. Классы и представители.

Для a и b запишите деление на m с неотрицательным остатком. Найдите представители r_a, r_b и симметричные s_a, s_b в оговорённом интервале. Определите, сравнимы ли a, b по модулю m, проверив и разность, и остатки. Запишите классы в виде $a + km$, $b + km$, $k \in \mathbb{Z}$.

Задание 2. Остаток многочлена.

Вычислите $F(a, b) = 3a^4 - 2a^2b + 5b^3 - 7$ по модулю m. Покажите уменьшение аргументов и промежуточных степеней. Объясните, почему замена аргументов их представителями сохраняет результат. Дайте неотрицательный остаток.

Задание 3. Правильное сокращение.

Найдите $g = \gcd(c, m)$ и проверьте сравнение $ca \equiv cb \pmod{m}$. Выпишите равносильное сравнение после сокращения и проверьте совпадение логических результатов. Укажите, позволяет ли общее правило сохранить модуль m. При $g > 1$ дополнительно приведите контрпример с числами $a_0 = 0, b_0 = m/g$.

Задание 4. Полная система или повторения.

Для $j = 0, \dots, m - 1$ постройте список $(cj + a) \bmod m$. Определите, образуют ли исходные m чисел полную систему вычетов. Укажите все различные остатки, их число и кратности. Докажите ответ через $\gcd(c, m)$. При неполноте предъявите разные индексы 0 и m/g, дающие одинаковый класс.

Задание 5. Работа с разными модулями.

Найдите $G = \gcd(m, n)$ и $L = \operatorname{lcm}(m, n)$. Для одной пары чисел объясните объединение двух сравнений по m, n в сравнение по L. Проверьте пару $u = a, v = a + L$ по модулям m, n, L, mn и поясните результат для произведения модулей.

Затем рассмотрите две разные пары: $u_1 = a + m \equiv a \pmod{m}$ и $v_1 = b + n \equiv b \pmod{n}$. Проверьте $u_1 + v_1 \equiv a + b \pmod{G}$ и $u_1 v_1 \equiv ab \pmod{G}$. Объясните, почему здесь используется общий делитель модулей.

Задание 6. Программная проверка.

Выполните программу со своими данными и общими проверками. Для задания 2 отдельно вычислите исходный многочлен целыми числами и сравните его остаток с результатом `polynomial_mod`. Для второй части задания 5 добавьте два вызова `congruent` с модулем `gcd(m, n)`. Кратко разберите ошибки: обычное деление сравнений, неверный знак отрицательного остатка, замена НОК произведением без проверки взаимной простоты.

8 Варианты индивидуальных заданий

Здесь m — основной модуль, n — второй модуль задания 5, a, b — аргументы, c — коэффициент сокращения и построения системы вычетов. В образце раздела 4 используются другие исходные данные.

Вариант	m	n	a	b	c
1	8	9	-20	28	3
2	9	15	-33	22	6
3	10	21	-46	24	4
4	12	18	-59	37	5
5	14	15	-72	41	6
6	15	25	-85	35	4
7	16	27	-98	54	6
8	18	24	-111	52	9
9	20	21	-124	56	6
10	21	28	-137	55	7
11	22	35	-150	49	5
12	24	36	-163	53	10
13	25	26	-176	79	10
14	26	39	-189	72	9
15	27	35	-202	68	6
16	28	42	-215	79	10
17	30	49	-228	73	7
18	12	20	-241	35	8
19	18	25	-254	70	5
20	24	36	-267	70	9

9 Требования к отчёту

Укажите название работы, Ф. И. О., группу, вариант и исходные данные. Приложите ручные решения с обоснованиями, таблицу остатков аффинного набора, код и вывод, результаты общих проверок и краткое объяснение обнаруженных различий между равенством и сравнением. Все конечные остатки должны лежать в указанном диапазоне. Утверждение о полной системе или допустимости сокращения нужно доказать, а не только подтвердить программой.

10 Критерии оценивания

Составляющая работы	Баллы
Классы, неотрицательные и симметричные представители	15
Остаток многочлена и допустимость преобразований	15
Сокращение, новый модуль и контрпример при необходимости	20
Система вычетов, число классов, кратности и доказательство	20
Разные модули, корректное использование НОД и НОК	20
Работающий код, сопоставление результатов и оформление	10
Итого	100

11 Контрольные вопросы

1. Чем класс вычетов отличается от своего представителя?
2. Почему модуль должен быть положительным в принятом соглашении?
3. Почему симметричный представитель при чётном модуле требует выбора для середины?
4. Почему сложение и умножение сравнений допустимы?
5. Почему совпадение квадратов не гарантирует совпадения самих чисел по модулю?
6. Как меняется модуль после сокращения общего множителя c?
7. Когда набор $cj + s$ даёт полную систему вычетов?
8. Почему при НОД $g > 1$ каждый полученный класс встречается g раз?
9. Когда для двух модулей нужен НОК, а когда достаточно общего делителя?
10. Почему нельзя заменять модуль его кратным без дополнительных условий?

Литература

Мартынов Л. М. **Алгебра и теория чисел для криптографии**. Лань, 2020. §§ 4.1–4.2, с. 102–106: свойства сравнений и полные системы вычетов; § 3.1.3: сокращение при взаимной простоте. Trappe W., Washington L. C. **Introduction to Cryptography with Coding Theory**. 2nd ed., 2006. § 3.3: сравнения и арифметика по модулю.