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

Дисциплина «Теоретико-числовые методы в криптографии»

ПРАКТИЧЕСКАЯ РАБОТА 6

ФУНКЦИЯ ЭЙЛЕРА И ЕЁ ВЫЧИСЛЕНИЕ

Методические указания и задание к выполнению

Аудиторная нагрузка: 1 академический час. Методичка и программные примеры изучаются перед выполнением заданий. На занятии выполняются задания 1–4; задания 5–6 предназначены для самостоятельной работы и не увеличивают обязательный объём.

1 Цель работы

Научиться вычислять функцию Эйлера по определению и по разложению числа на простые множители, проверять условие её мультипликативности и связывать полученное значение с числом обратимых классов вычетов.

2 Планируемые результаты

После выполнения работы студент должен уметь составлять приведённую систему вычетов для малого модуля, обосновывать формулу для простой степени, выполнять целочисленное вычисление по различным простым делителям и проверять результат независимым подсчётом. В решении требуется явно указывать область определения функции и условие взаимной простоты множителей.

3 Программное обеспечение

Достаточно Python 3 и любой среды его запуска. Используется только стандартный модуль `math`. Примеры можно последовательно скопировать в один файл `phi_work.py` или выполнить по ячейкам в том же порядке. Перед программной проверкой запишите ручное решение. На занятии разрешается использовать подготовленные функции; переписывать весь код за отведённый час не требуется.

4 Методические сведения

4.1 Определение и граничный случай

Для положительного целого числа n **функция Эйлера** равна числу целых a от 1 до n, взаимно простых с n:

$$\varphi(n) = \#\{a \in \mathbb{Z} : 1 \le a \le n,\ \gcd(a, n) = 1\}.$$

При $n > 1$ число n не учитывается, поэтому можно перебирать $1, \dots, n-1$. При $n = 1$ единственное число 1 взаимно просто с 1, откуда $\varphi(1) = 1$. Граница перебора в коде должна сохранять этот случай.

Для $n \ge 2$ значение $\varphi(n)$ равно числу обратимых классов в $\mathbb{Z}_n$. Нулевой класс не обратим. Поэтому число **ненулевых необратимых** классов равно $n - 1 - \varphi(n)$. Слова «ненулевых» и «всех» здесь существенны: всего необратимых классов, включая нулевой, будет $n - \varphi(n)$.

Пример прямого подсчёта. Из чисел от 1 до 12 взаимно просты с 12 только 1,5,7,11, значит, $\varphi(12) = 4$. Ненулевых необратимых классов $12 - 1 - 4 = 7$; их представители 2,3,4,6,8,9,10.



Рисунок 1. Прямой подсчёт даёт четыре подходящих числа, поэтому значение функции Эйлера для 12 равно 4.

4.2 Простое число и простая степень

Если p простое, каждое число $1, \dots, p-1$ взаимно просто с p. Поэтому $\varphi(p) = p - 1$. Для степени простого числа при $k \geq 1$ имеем

$$\varphi(p^k) = p^k - p^{k-1} = p^{k-1}(p-1).$$

Обоснование. Число не взаимно просто с p^k ровно тогда, когда оно делится на p. Среди $1, \dots, p^k$ таких чисел p^{k-1}. Вычитая их из общего количества, получаем формулу.

Например, $\varphi(49) = 49 - 7 = 42$, а $\varphi(16) = 16 - 8 = 8$. Формула $n - 1$ применима к простому n, а не к произвольной степени простого числа.

4.3 Вычисление по разложению

Пусть $n = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$, где простые числа $p_1, \dots, p_k$ **различны**. Тогда

$$\varphi(n) = \prod_{i=1}^{k} p_i^{\alpha_i - 1}(p_i - 1) = n \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right).$$

Вторая формула подсчитывает числа, не делящиеся ни на один простой делитель n. При одном делителе исключается доля $1/p$. При двух различных простых делителях p, q включения и исключения дают $n - n/p - n/q + n/(pq)$. Для большего числа делителей аналогично учитываются все пересечения. Их количества целые, поскольку произведение выбранных различных простых делителей делит n. Раскрытие произведения даёт ту же сумму.

Если $n = 1$, разложение не содержит простых множителей; пустое произведение принимается равным 1. Обе записи согласуются с $\varphi(1) = 1$.

Разобранный пример. Для $N = 180 = 2^2 \cdot 3^2 \cdot 5$ получаем

$$\varphi(180) = 2^{2-1}(2-1) \cdot 3^{2-1}(3-1) \cdot 5^{1-1}(5-1) = 2 \cdot 6 \cdot 4 = 48.$$

В программе удобно использовать только целочисленные операции. Начинаем с $R = N$ и для каждого различного простого делителя выполняем $R \leftarrow R - R/p$ ровно один раз.

Обработанный простой делитель	Вычисление	Новое значение R
До начала	$R = 180$	180
$p = 2$	$180 - 180/2$	90
$p = 3$	$90 - 90/3$	60
$p = 5$	$60 - 60/5$	48

На каждом шаге текущее R делится на ещё не обработанный простой делитель. В первоначальном разложении его степень присутствует; предыдущие шаги удаляли по одному экземпляру других простых делителей. Поэтому деление `//` здесь точное. Повторять поправку дважды для 2^2 или 3^2 нельзя: кратность уже учтена в исходном числе N.

4.4 Мультипликативность и её условие

Если $\gcd(r, s) = 1$, то разложения r и s не имеют общих простых множителей. Разделяя произведение в формуле для $\varphi(rs)$ на две части, получаем

$$\varphi(rs) = \varphi(r)\varphi(s), \qquad \gcd(r, s) = 1.$$

Пример с выполненным условием. Для $r = 8, s = 9$ имеем НОД $1, \varphi(8) = 4, \varphi(9) = 6$ и $\varphi(72) = 24 = 4 \cdot 6$.

Пример с общим делителем. Для $r = 6, s = 9$ НОД равен 3. Здесь $\varphi(6) = 2, \varphi(9) = 6$, а $\varphi(54) = 18$, поэтому $\varphi(54) \neq 2 \cdot 6$. Общий простой делитель 3 нельзя дважды включать в поправку для произведения.

Функция Эйлера также не является возрастающей на всех натуральных числах: $\varphi(11) = 10$, но $\varphi(12) = 4$. Размер значения зависит от простых делителей аргумента.

4.5 Количество допустимых множителей

Для аффинного преобразования $E_{a,b}(x) = ax + b \pmod m$ допустимы $\varphi(m)$ значений множителя a среди $0, \dots, m - 1$ и все m значений сдвига b. Число обратимых аффинных преобразований равно

$$K = m\varphi(m).$$

Разным парам (a, b) соответствуют разные преобразования: значение в 0 определяет b, а разность значений в 1 и 0 определяет a. При $m = 26$ имеем $\varphi(26) = 12$ и $K = 312$.

Если выбирать класс a равномерно из всех m классов, доля обратимых равна $\varphi(m)/m$. При выборе только из ненулевых классов знаменатель был бы $m - 1$. Указание множества выбора требуется для правильного подсчёта. Здесь оценивается число допустимых параметров, а не криптографическая стойкость схемы.

5 Программные примеры

5.1 Прямой подсчёт

Функция `coprimes(n)` возвращает подходящие числа из отрезка от 1 до `n` включительно. Функция `phi_direct(n)` подсчитывает их. Проверка входа отклоняет неположительные и нецелые значения.

```
from math import gcd

def require_positive(n):
    if type(n) is not int:
        raise TypeError("Требуется целое число")
    if n < 1:
        raise ValueError("Требуется n >= 1")

def coprimes(n):
    require_positive(n)
    return [a for a in range(1, n + 1)
            if gcd(a, n) == 1]

def phi_direct(n):
    return len(coprimes(n))
```

Список полезен при ручной сверке небольшого модуля. В частности, `coprimes(1)` возвращает `[1]`. Для $n > 1$ этот список является стандартной приведённой системой представителей.

5.2 Разложение пробными делениями

Словарь хранит простые делители как ключи, а их кратности как значения. После извлечения всех двоек проверяются нечётные делители. Переменная `n` внутри функции уменьшается, но исходное число у вызывающего кода сохраняется.

```
def factorize(n):
    require_positive(n)
    factors = {}
    d = 2
    while d * d <= n:
        while n % d == 0:
            factors[d] = factors.get(d, 0) + 1
            n //= d
        d = 3 if d == 2 else d + 2
    if n > 1:
        factors[n] = factors.get(n, 0) + 1
    return factors
```

Если после цикла остаток больше 1, он прост: при составном остатке существовал бы делитель не больше его квадратного корня, который уже должен был быть проверен. Для входа 1 функция возвращает пустой словарь.

5.3 Функция Эйлера по различным простым делителям

Перебор словаря `for p in factorize(n)` проходит по ключам, поэтому поправка выполняется один раз для каждого различного простого делителя. Показатели степеней используются при построении разложения; отдельно повторять поправку по ним не нужно.

```
def phi(n):
    require_positive(n)
    result = n
    for p in factorize(n):
        result -= result // p
    return result
```

Этот алгоритм не использует числа с плавающей точкой. Для одного аргумента пробные деления требуют не более порядка $\sqrt{n}$ проверок делимости. Прямой подсчёт выполняет n вычислений НОД. Это учебные оценки числа арифметических действий; стоимость операций с длинными целыми

зависит от длины числа. В частности, пробные деления не становятся быстрым методом факторизации больших чисел криптографического размера.

5.4 Запуск и ожидаемый результат

Выполните предыдущие блоки, затем следующий пример. В последних двух строках сравниваются случай взаимно простых аргументов и случай с общим простым делителем.

```
m, n = 12, 180
print("Взаимно простые с 12:", coprimes(m))
print("phi(12) =", phi_direct(m))
print("Разложение 180:", factorize(n))
print("phi(180) =", phi(n))
for r, s in [(8, 9), (6, 9)]:
    print(r, s, "НОД =", gcd(r, s),
          "phi(rs) =", phi(r * s),
          "phi(r)phi(s) =", phi(r) * phi(s))
```

Ожидаемый вывод

```
Взаимно простые с 12: [1, 5, 7, 11]
phi(12) = 4
Разложение 180: {2: 2, 3: 2, 5: 1}
phi(180) = 48
8 9 НОД = 1 phi(rs) = 24 phi(r)phi(s) = 24
6 9 НОД = 3 phi(rs) = 18 phi(r)phi(s) = 12
```

5.5 Самопроверка программы

Проверки охватывают единицу, простое число, простые степени, несколько различных простых множителей, условие мультипликативности и недопустимый вход. В цикле два способа вычисления сравниваются для первых ста положительных аргументов.

```
assert coprimes(1) == [1]
assert factorize(1) == {}
assert phi(1) == 1
assert phi(13) == 12
assert phi(16) == 8
assert phi(27) == 18
assert phi(180) == 48
assert phi(8 * 9) == phi(8) * phi(9)
assert phi(6 * 9) != phi(6) * phi(9)
for n in range(1, 101):
    assert phi(n) == phi_direct(n)
for bad in [0, -5]:
    try:
        phi(bad)
    except ValueError:
        pass
    else:
        raise AssertionError("Не отклонён неверный вход")
print("Все проверки пройдены")
```

Успешный запуск заканчивается строкой `Все проверки пройдены`. Если появляется `AssertionError`, найдите первый не прошедший пример и проверьте границы перебора, обработку остаточного простого множителя и кратность поправок. Численные проверки подтверждают работу программы на выбранных данных; формулы обосновываются математически в разделе 4.

5.6 Дополнительная проверка по делителям

Для любого положительного n справедливо тождество

$$\sum_{d \mid n} \varphi(d) = n.$$

Сумма берётся по всем положительным делителям. Для доказательства разобьём числа $1, \dots, n$ по значению НОД с n. Если этот НОД равен n/d, то число имеет вид $a = (n/d)u$, где $1 \le u \le d$ и $\gcd(u, d) = 1$. Таких чисел ровно $\varphi(d)$. Каждый исходный элемент попал ровно в одну группу, поэтому сумма равна n.

Например, для 12 делители равны 1,2,3,4,6,12, а значения функции — 1,1,2,2,2,4. Их сумма равна 12. Код для самостоятельной работы:

```
def divisor_check(n):
    require_positive(n)
    divisors = [d for d in range(1, n + 1)
                if n % d == 0]
    values = [phi(d) for d in divisors]
    return divisors, values, sum(values)

assert divisor_check(12) == (
    [1, 2, 3, 4, 6, 12], [1, 1, 2, 2, 2, 4], 12)
```

Функция возвращает три объекта: список делителей, список значений функции и их сумму. Подсчёт всех делителей полным перебором здесь предназначен для небольших учебных аргументов.

6 Порядок выполнения работы

Для занятия продолжительностью 45 минут предлагается распределение: 8 минут на разбор примера и условий формул, 10 минут на задание 1, 10 минут на задание 2, 8 минут на задание 3, 6 минут на программную проверку и 3 минуты на итоговую сверку. Методичку и код следует просмотреть до занятия.

Получите номер варианта и выпишите m, N, r, s из раздела 8. Во всех обязательных заданиях сначала покажите ручное решение, затем используйте программу. Таблица вариантов задаёт разные аргументы для разных действий: m нужен для списка классов, N — для разложения, а r, s — для проверки мультипликативности.

7 Индивидуальные задания

Задание 1. Прямой подсчёт. Составьте приведённую систему по модулю m, найдите $\varphi(m)$ и число ненулевых необратимых классов. Для каждого включённого представителя подтвердите взаимную простоту с m; для исключённых укажите общий простой делитель с модулем. Можно группировать исключённые числа по делителю.

Задание 2. Разложение и вычисление. Разложите N на простые множители и проверьте разложение умножением. Вычислите $\varphi(N)$ по произведению $p^{\alpha-1}(p-1)$. Отдельно запишите последовательность целочисленных поправок $R \leftarrow R - R/p$ и убедитесь, что результат совпал. Укажите, почему каждый простой делитель обрабатывается один раз.

Задание 3. Условие мультипликативности. Найдите $g = \gcd(r, s)$, $\varphi(r)$, $\varphi(s)$ и $\varphi(rs)$. Сравните $\varphi(rs)$ с $\varphi(r)\varphi(s)$. При $g = 1$ объясните применимость свойства; при $g > 1$ укажите общие простые множители и вычислите обе стороны независимо.

Задание 4. Программная проверка. Используя готовые функции, сравните прямой подсчёт и вычисление по разложению для m, N и rs. Сохраните вывод. Проверьте дополнительно входы 1, 13 и 0, объяснив ожидаемый результат и ошибку для нуля. Сопоставьте каждый численный результат с ручным решением.

Задание 5. Тождество по делителям — самостоятельно. Найдите все положительные делители N, вычислите для каждого $\varphi(d)$ и проверьте, что сумма равна N. Приложите список делителей и соответствующих значений, а не только итог программы. Объясните, почему в сумме требуется $\varphi(1) = 1$.

Задание 6. Подсчёт преобразований — самостоятельно. Для модуля m найдите число $K = m\varphi(m)$ обратимых аффинных преобразований и долю $\varphi(m)/m$ допустимых множителей среди всех классов. Сократите дробь. Докажите, что разные допустимые пары (a, b) задают разные преобразования.

8 Варианты индивидуальных заданий

Во всех вариантах $m \geq 2$ и $N > 1$. Для задания 3 варианты с нечётными номерами содержат взаимно простые r, s, а варианты с чётными — аргументы с общим делителем. НОД всё равно требуется вычислить и привести в решении.

Вариант	m	N	r	s
1	8	72	4	9
2	9	90	6	10
3	10	100	8	15
4	12	108	9	12
5	14	126	7	16
6	15	132	10	15
7	16	140	9	20
8	18	150	12	18
9	20	156	11	15
10	21	168	14	21
11	22	180	8	21
12	24	198	15	25
13	25	200	5	24
14	26	210	16	24
15	27	216	9	28
16	28	220	18	27
17	30	225	11	20
18	32	234	20	30
19	33	242	13	18
20	35	270	21	28

9 Требования к отчёту

Укажите тему, номер варианта и исходные данные. Затем приведите список взаимно простых представителей, значение $\varphi(m)$ и число ненулевых необратимых классов. Для аргумента N запишите проверенное разложение, формулу и все промежуточные поправки. Для пары (r, s) оформите вычисления обеих сторон и вывод о применимости свойства.

Программная часть включает используемые функции или ссылку на соответствующие блоки методички, фактический вывод для своего варианта и объяснение совпадения с ручным решением. Вывод без исходных данных и без математического обоснования не заменяет решение. Дополнительные задания оформляются после обязательных.

10 Критерии оценивания

Составляющая обязательной работы	Баллы
Приведённая система, прямой подсчёт и число ненулевых необратимых классов	25
Разложение N, вычисление по формуле и последовательность поправок	35
НОД пары, обе стороны равенства и объяснение условия мультипликативности	20
Программная сверка и граничные случаи	10
Ясная запись решения и ответы при защите	10
Всего	100

Задания 5–6 служат для углубления и обсуждения. Их отсутствие не уменьшает оценку за полностью выполненные обязательные задания. При арифметической ошибке оценивается также верность способа решения; ошибка в условии применимости формулы рассматривается отдельно от счётной ошибки.

11 Контрольные вопросы

1. Что подсчитывает функция Эйлера и для каких аргументов она определена в работе?
2. Почему $\varphi(1) = 1$, хотя привычный список от 1 до $n - 1$ здесь пуст?
3. Почему для простого p получается $p - 1$, а для p^k нужно исключить p^{k-1} чисел?
4. Почему в произведении поправок участвуют различные простые делители?
5. Как вычислить функцию Эйлера без дробной арифметики?
6. Зачем проверять $\gcd(r, s)$ перед применением мультипликативности?
7. Может ли функция Эйлера уменьшаться при увеличении аргумента?
8. Чем отличаются $n - \varphi(n)$ и $n - 1 - \varphi(n)$ при $n \geq 2$?
9. Почему остаток после цикла пробных делений, если он больше 1, является простым?
10. Какие случаи следует проверить, кроме одного удачного численного примера?
11. Как объясняется тождество $\sum_{d|n} \varphi\,(d) = n$?
12. Почему число обратимых аффинных преобразований равно $m\varphi(m)$?

Литература

Мартынов Л. М. Алгебра и теория чисел для криптографии. СПб.: Лань, 2020. §4.3, с. 106–112 — функция Эйлера, формула вычисления, мультипликативность и сумма по делителям; §4.4, с. 112–114 — обратимые классы. Дополнительно используются определения и тождество Безу из ранее подготовленных материалов курса.

Примеры, варианты и программы составлены для этой практической работы. Функция Эйлера и теорема Эйлера — разные понятия: теорема о степенях обратимых классов относится к следующей лекции.