Лабораторна робота 4

Тема: Структури даних у Python: списки, кортежі, множини та словники.

Мета: Навчитися обирати структури даних, опрацьовувати записи й накопичувати результати за ключами.

Зміст роботи

Завдання 1. Опрацювання списку чисел

Введіть цілі числа одним рядком через пробіли. Порожній рядок задає порожній список. Дані коректні; власні функції та винятки не потрібні.

1. Виведіть список, довжину й суму.
2. Створіть незалежний список, замінивши від’ємні числа нулями.
3. Виберіть додатні числа в порядку появи й виведіть також їхню відсортовану копію. Покажіть, що початковий список збережено.

Дозволено цикли, спискові включення, sum() і sorted(). Перевірте власні дані та порожній рядок.

Таблиця 1 — Контрольна перевірка списків

Вхід	**Виведення: показники**	**Виведення: нові списки**
-3 0 5 -1 5 2	Довжина: 6 Сума: 8 Початковий: [-3, 0, 5, -1, 5, 2]	Заміна: [0, 0, 5, 0, 5, 2] Додатні: [5, 5, 2] Відсортовані: [2, 5, 5]
-4 -2	Довжина: 2 Сума: -6 Початковий: [-4, -2]	Заміна: [0, 0] Додатні: [] Відсортовані: []
Порожній рядок	Довжина: 0 Сума: 0 Початковий: []	Заміна: [] Додатні: [] Відсортовані: []

Завдання 2. Опрацювання записів у кортежах

Введіть додатне ціле n. Для кожного запису введіть окремо непорожню назву та два числа за своїм варіантом таблиці 2. Збережіть записи як список кортежів (назва, a, b).

1. Переберіть записи з розпакуванням кортежів та обчисліть показник кожного.

2. Виведіть назви і показники з двома десятковими знаками, підсумок за правилом таблиці та назву запису з найбільшим показником. За рівності оберіть перший.

Числа дійсні, крім явно зазначених цілих полів; десятковий роздільник — крапка. Кортежі не змінюйте. Перевірте один запис, рівні показники й власні дані. У контролі n = 2; після назви кожного запису наведено a та b, які вводять окремо. Під «Виведенням» подано очікувані результати; формулювання підписів можна обрати самостійно.

Середнє в таблиці — арифметичне середнє показників окремих записів, не показник об’єднаного рейсу чи суміші. Одиниці зазначено біля полів. Введення залишається коректним.

Таблиця 2 — Кортежі записів і похідні показники

Варіант	**Поля та межі**	**Показник**	**Підсумок**	**Контрольний приклад**
1	Товар. a — ціна, грн > 0; b — кількість ≥ 0.	ab	Сума вартостей	**Вхід:** n = 2 А: a = 10, b = 3 Б: a = 5, b = 4 **Виведення:** А: 30.00; Б: 20.00 Підсумок: 50.00 Найбільший: А
2	Рейс. a — км > 0; b — год > 0.	a / b, км/год	Середнє показників	**Вхід:** n = 2 А: a = 120, b = 2 Б: a = 90, b = 3 **Виведення:** А: 60.00; Б: 30.00 Підсумок: 45.00 Найбільший: А
3	Оцінки a, b від 0 до 100.	(a + b) / 2	Середнє показників	**Вхід:** n = 2 А: a = 70, b = 90 Б: a = 60, b = 80 **Виведення:** А: 80.00; Б: 70.00 Підсумок: 75.00 Найбільший: А
4	Деталь. a — г > 0; b — см³ > 0.	a / b, г/см³	Середнє показників	**Вхід:** n = 2 А: a = 12, b = 3 Б: a = 10, b = 2 **Виведення:** А: 4.00; Б: 5.00 Підсумок: 4.50 Найбільший: Б

Варіант	Поля та межі	Показник	Підсумок	Контрольний приклад
5	Ділянка. a, b — сторони, м > 0.	ab, м²	Сума площ	**Вхід:** n = 2 A: a = 4, b = 3 Б: a = 5, b = 2 **Виведення:** A: 12.00; Б: 10.00 Підсумок: 22.00 Найбільший: A
6	Знижка. a — ціна, грн > 0; b — % від 0 до 100.	a(1 − b / 100), грн	Сума цін після знижки	**Вхід:** n = 2 A: a = 100, b = 20 Б: a = 60, b = 10 **Виведення:** A: 80.00; Б: 54.00 Підсумок: 134.00 Найбільший: A
7	Виконання. a — план > 0; b — виконано ≥ 0.	100b / a, %	Середнє показників	**Вхід:** n = 2 A: a = 20, b = 15 Б: a = 40, b = 36 **Виведення:** A: 75.00; Б: 90.00 Підсумок: 82.50 Найбільший: Б
8	Маршрут. a, b — етапи, км ≥ 0.	a + b, км	Сума довжин	**Вхід:** n = 2 A: a = 5, b = 8 Б: a = 4, b = 6 **Виведення:** A: 13.00; Б: 10.00 Підсумок: 23.00 Найбільший: A
9	Бюджет. a — дохід, b — витрати, грн ≥ 0.	a − b, грн	Сума залишків	**Вхід:** n = 2 A: a = 100, b = 40 Б: a = 90, b = 50 **Виведення:** A: 60.00; Б: 40.00 Підсумок: 100.00 Найбільший: A
10	Тренування. a — повторення, b — підходи; цілі ≥ 0.	ab	Сума повторень	**Вхід:** n = 2 A: a = 12, b = 3 Б: a = 10, b = 4 **Виведення:** A: 36.00; Б: 40.00 Підсумок: 76.00 Найбільший: Б
11	Суміш. a, b — кг ≥ 0; a + b > 0.	100a / (a + b), %	Середнє показників	**Вхід:** n = 2 A: a = 2, b = 3 Б: a = 3, b = 1 **Виведення:** A: 40.00; Б: 75.00 Підсумок: 57.50 Найбільший: Б

Варіант	Поля та межі	Показник	Підсумок	Контрольний приклад
12	Вимірювання. a, b — довільні дійсні в одній одиниці.	|a − b|	Середнє відхилень	**Вхід:** n = 2 А: a = 12, b = 9 Б: a = 4, b = 10 **Виведення:** А: 3.00; Б: 6.00 Підсумок: 4.50 Найбільший: Б
13	Робота. a — год ≥ 0; b — грн/год > 0.	ab; якщо a > 8, додати 0.5b(a − 8).	Сума оплат	**Вхід:** n = 2 А: a = 10, b = 20 Б: a = 8, b = 25 **Виведення:** А: 220.00; Б: 200.00 Підсумок: 420.00 Найбільший: А
14	Пакування. a — вміст, b — тара, кг ≥ 0; a + b > 0.	100b / (a + b), %	Середнє показників	**Вхід:** n = 2 А: a = 9, b = 1 Б: a = 8, b = 2 **Виведення:** А: 10.00; Б: 20.00 Підсумок: 15.00 Найбільший: Б
15	Друк. a — цілі сторінки ≥ 0; b — грн/сторінку > 0.	ab; якщо a ≥ 10, знижка 10%.	Сума вартостей	**Вхід:** n = 2 А: a = 10, b = 2 Б: a = 8, b = 3 **Виведення:** А: 18.00; Б: 24.00 Підсумок: 42.00 Найбільший: Б

Завдання 3. Операції над множинами слів

Введіть два рядки слів без розділових знаків. Через lower() та split() утворіть множини A й B. Порожній рядок задає порожню множину.

1. Виведіть перетин, об’єднання, різницю A − B й симетричну різницю.
2. Кожен результат подайте відсортованим списком і зазначте кількість елементів.

Повторення не враховуйте. Перевірте власні дані, однакові множини та порожній рядок.

Таблиця 3 — Контрольна перевірка множин

Вхідні рядки	Операція	Очікуване виведення
A: Python код дані код B: дані МОВА Python	Перетин	Список: ['python', 'дані'] Кількість: 2
A: Python код дані код B: дані МОВА Python	Об’єднання	Список: ['python', 'дані', 'код', 'мова'] Кількість: 4

Вхідні рядки	Операція	Очікуване виведення
A: Python код дані код B: дані МОВА Python	Різниця A − B	Список: ['код'] Кількість: 1
A: Python код дані код B: дані МОВА Python	Симетрична різниця	Список: ['код', 'мова'] Кількість: 2
A: порожній рядок B: порожній рядок	Усі чотири операції	Список: [] Кількість: 0

Завдання 4. Частоти, групування і накопичення у словнику

Введіть дані за своїм варіантом таблиці 4 і побудуйте словник. Слова складаються з літер, регістр нормалізуйте через lower(). Цілі числа можуть бути від’ємними. Записи назва;число вводьте окремими рядками після невід’ємного n.

1. Накопичуйте значення за правилом варіанта; не створюйте ключів без даних.
2. Виведіть словник за зростанням ключів і кількість різних ключів. Для значень-списків збережіть порядок та повторення.

Для порожнього рядка або n = 0 виведіть «Даних немає» та кількість ключів 0. Ключі в одному словнику мають один тип. Перевірте повторення, порожні та власні дані. У контрольному стовпці окремо наведено вхідні дані, очікуваний словник і кількість його ключів.

Для варіантів із записами в контрольному прикладі спочатку вказано n, далі — кожний запис з нового рядка. Словник у результаті подано за зростанням ключів; підписи під час виведення можна обрати самостійно.

Таблиця 4 — Словники для різних способів опрацювання

Варіант	Вхідні дані	Ключ і значення	Контрольний приклад
1	Рядок слів	Ключ — слово, значення — частота.	**Вхід:** код дані код **Виведення:** {'дані': 1, 'код': 2} Кількість ключів: 2
2	Рядок слів	Ключ — довжина, значення — список слів.	**Вхід:** я код дані код **Виведення:** {1: ['я'], 3: ['код', 'код'], 4: ['дані']} Кількість ключів: 3

Варіант	Вхідні дані	Ключ і значення	Контрольний приклад
3	Рядок слів	Ключ — перша літера, значення — кількість слів.	**Вхід:** код кіт дані **Виведення:** {'д': 1, 'к': 2} Кількість ключів: 2
4	Рядок слів	Ключ — остання літера, значення — сума довжин слів.	**Вхід:** код мед мова **Виведення:** {'а': 4, 'д': 6} Кількість ключів: 2
5	Рядок цілих чисел	Ключ — число, значення — частота.	**Вхід:** 2 -1 2 0 **Виведення:** {-1: 1, 0: 1, 2: 2} Кількість ключів: 3
6	Рядок цілих чисел	Ключ — знак −1, 0, 1, значення — сума чисел.	**Вхід:** -3 -2 0 4 1 **Виведення:** {-1: -5, 0: 0, 1: 5} Кількість ключів: 3
7	Рядок цілих чисел	Ключ — модуль числа, значення — список початкових чисел.	**Вхід:** -2 2 -3 2 **Виведення:** {2: [-2, 2, 2], 3: [-3]} Кількість ключів: 2
8	Рядок слів	Ключ — перша літера, значення — перше найдовше слово з цією літерою.	**Вхід:** кіт код книга дані **Виведення:** {'д': 'дані', 'к': 'книга'} Кількість ключів: 2
9	Записи назва;ціла кількість ≥ 0	Ключ — назва, значення — сумарна кількість.	**Вхід:** n = 3 А;2 Б;3 А;4 **Виведення:** {'а': 6, 'б': 3} Кількість ключів: 2
10	Записи назва;ціла оцінка 0…100	Ключ — назва, значення — найбільша оцінка.	**Вхід:** n = 3 А;70 Б;90 А;85 **Виведення:** {'а': 85, 'б': 90} Кількість ключів: 2

Варіант	Вхідні дані	Ключ і значення	Контрольний приклад
11	Записи категорія;ціле число	Ключ — категорія, значення — список чисел.	**Вхід:** n = 3 А;2 Б;-1 А;3 **Виведення:** {'а': [2, 3], 'б': [-1]} Кількість ключів: 2
12	Рядок слів	Ключ — довжина, значення — кількість слів такої довжини.	**Вхід:** код кіт мова **Виведення:** {3: 2, 4: 1} Кількість ключів: 2
13	Рядок цілих чисел	Ключ — остання цифра модуля, значення — список чисел.	**Вхід:** -12 2 15 22 **Виведення:** {2: [-12, 2, 22], 5: [15]} Кількість ключів: 2
14	Рядок слів	Ключ — слово, значення — список позицій від 1.	**Вхід:** код дані код **Виведення:** {'дані': [2], 'код': [1, 3]} Кількість ключів: 2
15	Записи назва;ціле число	Ключ — назва, значення — останнє введене число.	**Вхід:** n = 3 А;2 Б;3 А;5 **Виведення:** {'а': 5, 'б': 3} Кількість ключів: 2

Оформлення звіту

Звіт оформлюють за чинними вимогами курсу (документ «Вимоги до оформлення»).

Для кожного завдання у звіті подають формулювання, текст програми та скріншот результату в консолі.

Методичні рекомендації

Списки та незалежні копії

Список зберігає впорядковану послідовність елементів і дозволяє змінювати їх. list() створює список, [] — порожній список. Звернення values[0] вибирає перший елемент, а append(value) додає один елемент у кінець.

len(values) повертає кількість елементів; sum(values) — суму чисел, для порожнього списку 0.

Для введення чисел використайте text.split(), а потім перетворіть кожне поле через int(). Наприклад, [int(part) for part in "2 -1 5".split()] утворює [2, -1, 5]. Це спискове включення: спочатку записано вираз для нового елемента, далі — перебір. Звичайний цикл з append() дає той самий результат.

Присвоєння other = values не створює копії: обидві змінні посилаються на один список. Для незалежної копії простого списку чисел використайте values.copy() або values[:]. sorted(values) повертає новий відсортований список; values.sort() змінює початковий і повертає None. Якщо початкові дані потрібно зберегти, обирайте sorted().

У виразі [x for x in values if x > 0] умова визначає, які елементи потраплять до результату. Вираз [0 if x < 0 else x for x in values] змінює значення кожного елемента у новому списку. Порядок перебору зберігається; порожні дані утворюють порожній результат.

Код Python

```
values = [3, -1, 2]
copy = values.copy()
copy.append(7)
ordered = sorted(values)
print(values)
print(copy)
print(ordered)
```

Результат виконання

```
[3, -1, 2]
[3, -1, 2, 7]
[-1, 2, 3]
```

Отримаємо [3, -1, 2], [3, -1, 2, 7] та [-1, 2, 3]. Додавання до copy і сортування не змінили values.

Добір і перетворення у новому списку

Код Python

```
values = [-2, 0, 3]
positive = [x for x in values if x > 0]
doubled = [2 * x for x in values]
print(positive)
print(doubled)
print(values)
```

Результат виконання

```
[3]
[-4, 0, 6]
[-2, 0, 3]
```

Умова if визначає добір, вираз перед for — перетворення. Обидва включення створили нові списки, тому початковий values не змінився.

Кортежі та розпакування записів

Кортеж tuple — впорядкована послідовність, елементи якої не можна переприсвоювати. Запис ("Зошит", 25.0, 3) зручно використати для трьох пов’язаних полів. Кортеж з одного елемента потребує коми: (5,). Список кортежів дозволяє додавати нові записи, зберігаючи склад кожного запису.

Присвоєння name, price, quantity = record розпаковує три поля у три змінні. Кількість змінних має відповідати кількості полів. Так само можна записати for name, price, quantity in records. Вартість обчислюють як price * quantity; формат .2f задає два десяткові знаки лише під час виведення.

Для вибору першого запису з найбільшим показником початковий результат установіть за першим записом, а далі оновлюйте лише за строго більшого значення. Оператор >= замість > обере останній запис за рівності. Нульова кількість не є помилкою й дає нульову вартість.

Поля запису розпаковуються в циклі

Код Python

```
records = [("Місто А", 12, 3), ("Місто Б", 10, 2)]
for name, amount, count in records:
    value = amount / count
    print(f"{name}: {value:.2f}")
```

Результат виконання

```
Місто А: 4.00
Місто Б: 5.00
```

Кортеж містить пов’язані поля. Кожна ітерація розпаковує їх у три змінні; формулу застосовано до поточного запису. Підсумок усіх записів обирайте за своїм варіантом.

Множини та операції над ними

Множина set зберігає унікальні елементи без установленого порядку. set([2, 2, 3]) містить 2 і 3. Для порожньої множини використайте set(): запис {} створює порожній словник. Індекси до множини не застосовують.

A & B — перетин, A | B — об’єднання, A - B — елементи лише першої множини, A ^ B — елементи рівно однієї множини. Наприклад, для A = {1, 2}, B = {2, 3} результати — {2}, {1, 2, 3}, {1}, {1, 3}. Різниця залежить від порядку операндів.

Перед створенням множин слів нормалізуйте регістр: set(text.lower().split()). Для відтворюваного виведення застосуйте sorted(result), що повертає список. len(result) рахує різні елементи, а не кількість їхніх повторень у початковому тексті.

Чотири операції множин

Код Python

```
A = {1, 2}
B = {2, 3}
print(sorted(A & B))
print(sorted(A | B))
print(sorted(A - B))
print(sorted(A ^ B))
```

Результат виконання

```
[2]
[1, 2, 3]
[1]
[1, 3]
```

sorted() робить результат відтворюваним. Різниця A − B не дорівнює B − A; симетрична різниця містить елементи рівно однієї множини.

Словники, лічильники та групи

Словник dict зберігає відповідність «ключ — значення». Ключі унікальні; значеннями можуть бути числа або списки. Для порожнього словника використайте {}. key in groups перевіряє наявність ключа, groups[key] читає його значення. Читання відсутнього ключа спричинить KeyError.

Для групування створюйте окремий порожній список для кожного нового ключа, а потім додавайте елемент через append(). Виклик groups.setdefault(key,

[]) повертає вже наявний список або створює його, якщо ключа немає. Усі групи не повинні посилатися на один спільний список.

Перебір for key in sorted(groups) дає ключі за зростанням. Для числового списку groups[key] використайте len() і sum(); для груп слів sum() без перетворення не підходить. Для підрахунку частот зручно counts[word] = counts.get(word, 0) + 1: get() повертає 0, коли слова ще немає у словнику.

Для додатного дільника d вираз x % d дає невід'ємну остачу, а x // d — частку з округленням вниз. Тому -5 % 3 дорівнює 1, -5 // 3 — -2. Для десяткових цифр спочатку застосуйте abs(x): abs(-12) % 10 дає 2. len(str(abs(x))) рахує цифри, зокрема для нуля повертає 1.

Частота за ключем і окремі списки груп

Код Python

```
counts = {}
groups = {}
for word in "код кіт дані код".split():
    counts[word] = counts.get(word, 0) + 1
    groups.setdefault(len(word), []).append(word)
print(dict(sorted(counts.items())))
print(dict(sorted(groups.items())))
```

Результат виконання

```
{'дані': 1, 'код': 2, 'кіт': 1}
{3: ['код', 'кіт', 'код'], 4: ['дані']}
```

get() задає початковий нуль для частоти. setdefault() створює окремий список для нового ключа або повертає вже наявний. Повторення зберігаються у групі.

Часті помилки

Не змінюйте початковий список через його друге ім'я, якщо потрібна копія. Не очікуйте стабільного порядку від множини. Розрізняйте {} та set(). Під час групування зберігайте повторення: множина всередині групи їх втратить. Перевіряйте порожні дані до звернення до першого елемента.

Різні значення словника й похідні показники

Для частоти використайте counts[key] = counts.get(key, 0) + 1, для суми — totals[key] = totals.get(key, 0) + value. Для останнього значення достатньо values[key] = value. Для максимуму спочатку перевірте, чи ключ уже існує; відсутнє значення не слід підміняти нулем, якщо дозволено від'ємні числа.

Для першого найдовшого слова за ключем запам’ятайте перше, а замінюйте лише за строго більшої довжини. Для позицій перебирайте enumerate(words, start=1) і додавайте позицію до окремого списку ключа. Число різних ключів дорівнює len(dictionary).

Формули записів використовують звичайні операції +, −, *, / та abs(). Перед діленням знаменник має бути ненульовим згідно з умовою. Знижка 20% означає множник 1 − 20 / 100 = 0.8; додаткова оплата застосовується лише після перевірки порога.

Сума й максимум за назвою

Код Python

```
sums = {}
best = {}
for name, value in [("а", -2), ("б", 3), ("а", -1)]:
    sums[name] = sums.get(name, 0) + value
    if name not in best or value > best[name]:
        best[name] = value
print(sums)
print(best)
```

Результат виконання

```
{'а': -3, 'б': 3}
{'а': -1, 'б': 3}
```

Для суми початковий нуль допустимий; для максимуму його підставляти не можна, якщо числа можуть бути від’ємними. Умова перевіряє наявність ключа до читання best[name].

Приклади виконання завдань

Приклад до завдання 1

Навчальна задача. Для введених слів побудуйте список їхніх довжин, виберіть слова довжиною не більше чотирьох символів і покажіть відсортовану копію списку довжин.

Код Python

```
words = input("Слова: ").split()
lengths = [len(word) for word in words]
short = [word for word in words if len(word) <= 4]
ordered = sorted(lengths)
print(f"Початкові слова: {words}")
print(f"Довжини: {lengths}")
print(f"Короткі слова: {short}")
print(f"Відсортовані довжини: {ordered}")
```

Приклад виконання в консолі

```
Слова: мова Python код дані
Початкові слова: ['мова', 'Python', 'код', 'дані']
Довжини: [4, 6, 3, 4]
Короткі слова: ['мова', 'код', 'дані']
Відсортовані довжини: [3, 4, 4, 6]
```

Пояснення до прикладу

split() утворює чотири слова. Перше списокове включення перетворює кожне слово на його довжину: [4, 6, 3, 4]. Друге включення залишає «мова», «код» і «дані» у початковому порядку.

sorted(lengths) повертає [3, 4, 4, 6]; початковий lengths не змінюється. Для завдання 1 перенесіть ці прийоми на числа й самостійно визначте потрібне перетворення та умову добору.

Приклад до завдання 2

Навчальна задача. Збережіть назви міст та температури у кортежах. Визначте середню температуру й перше місто з найбільшою температурою. Кількість записів додатна; введення коректне.

Код Python

```python
n = int(input("Кількість міст: "))
records = []
for _ in range(n):
    city = input("Місто: ").strip()
    temperature = float(input("Температура: "))
    records.append((city, temperature))
total = 0.0
warmest, highest = records[0]
for city, temperature in records:
    total += temperature
    if temperature > highest:
        warmest, highest = city, temperature
print(f"Середня температура: {total / n:.1f}")
print(f"Найтепліше місто: {warmest}")
```

Приклад виконання в консолі

```
Кількість міст: 2
Місто: Житомир
Температура: 18
Місто: Львів
Температура: 16
Середня температура: 17.0
Найтепліше місто: Житомир
```

Пояснення до прикладу

Список records містить два кортежі: («Житомир», 18.0) і («Львів», 16.0). У циклі поля розпаковуються в city та temperature. Сума 34.0, тому середнє дорівнює 17.0.

Початковий кандидат — перше місто. Умова > змінює його лише за вищої температури; за рівності залишиться перше. У завданні 2 записи мають три поля, а порівнювати потрібно показник, обчислений за правилом свого варіанта.

Приклад до завдання 3

Навчальна задача. Для двох слів знайдіть спільні літери та літери, які є лише в першому слові. Регістр і повторення не враховуйте.

Код Python

```
first = set(input("Перше слово: ").strip().lower())
second = set(input("Друге слово: ").strip().lower())
common = first & second
only_first = first - second
print(f"Спільні літери: {sorted(common)}")
print(f"Лише в першому: {sorted(only_first)}")
print(f"Кількість спільних: {len(common)}")
```

Приклад виконання в консолі

```
Перше слово: мова
Друге слово: море
Спільні літери: ['м', 'о']
Лише в першому: ['а', 'в']
Кількість спільних: 2
```

Пояснення до прикладу

lower() вирівнює регістр, а set() утворює множину символів без повторень. Перетин first & second залишає «м» та «о». Різниця first - second містить «а» та «в», які є лише в першому слові; порядок операндів важливий.

sorted() подає результат упорядкованим списком, не змінюючи множини. len(common) рахує різні спільні літери — отримуємо 2. Для цілих слів у завданні 3 використайте set(text.lower().split()). Аналогічно виведіть об’єднання, симетричну різницю та кількість елементів кожного результату.

Приклад до завдання 4

Навчальна задача. Згрупуйте введені слова за останньою літерою без урахування регістру. Усередині груп збережіть початкове написання та порядок.

Код Python

```
words = input("Слова: ").split()
groups = {}
for word in words:
    key = word[-1].lower()
    if key not in groups:
        groups[key] = []
    groups[key].append(word)
for key in sorted(groups):
    print(f"{key}: {groups[key]}; кількість {len(groups[key])}")
if not groups:
    print("Слів немає")
```

Приклад виконання в консолі

```
Слова: мова школа код сад
а: ['мова', 'школа']; кількість 2
д: ['код', 'сад']; кількість 2
```

Пояснення до прикладу

word[-1].lower() утворює ключ за останньою літерою: «а» для «мова» й «школа», «д» для «код» і «сад». Умова key not in groups створює порожній список лише для нового ключа. Якби список створювався щоразу, попередні слова цієї групи втрачалися б.

append(word) виконується для кожного слова, зберігаючи його написання, порядок і повторення. sorted(groups) упорядковує ключі, а списки груп не змінює. Для порожнього введення словник залишається порожнім і спрацьовує if not groups.

У завданні 4 оберіть ключ і накопичення за своїм варіантом: список, частоту, суму, найбільше або останнє число. len(словник) дає кількість різних ключів; вона може відрізнятися від кількості введених елементів.

Контрольні питання

1. Коли потрібні список, кортеж, множина та словник?
2. Чим копія списку відрізняється від другого імені?
3. Як розпакувати кортеж у циклі?
4. Як зберегти перший запис за рівності показників?
5. Чим відрізняються sorted() і sort()?
6. Що означають &, |, − та ^ для множин?
7. Чому set() і {} задають різні типи?
8. Як get() допомагає рахувати частоти?
9. Як накопичувати список або суму за ключем?
10. Як забезпечити визначений порядок виведення?