Программа начинается с создания пустого словаря. После этого запускается цикл `while` с проверкой на количество пар ключ-значение в словаре. Поскольку словарь пока пуст, условное выражение вернет `True`, и будет запущено тело цикла.

На каждой итерации пользователь будет вводить строку с клавиатуры. После этого при помощи оператора `in` будет определено, присутствует ли введенный пользователем ключ в списке. Если да, то ассоциированный с этим ключом счетчик (значение) будет увеличен на единицу. В противном случае словарь пополнится новой парой ключ-значение. Цикл будет продолжаться, пока пользователь не введет пять уникальных строк с клавиатуры. После этого на экран будут выведены все пары ключ-значение из списка.

6.5. Словари как аргументы и возвращаемые значения функций

Словари могут быть переданы в функцию подобно значениям других типов. Как и в случае со списками, изменение переменной параметра, хранящей переданный словарь, может привести к модификации источника. Например, удаление, изменение или добавление пары ключ-значение неизбежно приведет к соответствующему изменению аргумента. В то же время присвоение переменной параметра другого значения (когда слева от знака равенства указывается только переменная без квадратных скобок) аргумента не затронет. Как и в случае с переменными других типов, функция может возвращать словарь при помощи ключевого слова `return`.

6.6. Упражнения

Несмотря на то что многие упражнения из данного раздела могут быть решены при помощи списков и выражений `if`, большинство из них имеют решения и с применением словарей. Поскольку мы в этой главе говорим главным образом о словарях, используйте их в своих программах вместе с другими конструкциями языка Python, которые вы изучили ранее в книге.

Упражнение 136. Поиск по значению

(*Решено. 45 строк*)

Напишите функцию с названием `reverseLookup`, которая будет осуществлять поиск всех ключей в словаре по заданному значению. Функция должна принимать в качестве параметров словарь и значение для поиска

и возвращать список ключей (он может быть пустым) из этого словаря, соответствующих переданному значению.

В основной программе продемонстрируйте работу функции путем создания словаря и поиска в нем всех ключей по заданному значению. Убедитесь, что функция работает корректно при наличии нескольких ключей для искомого значения, одного ключа и их отсутствии. Ваша программа должна запускаться только в том случае, если она не импортирована в виде модуля в другой файл.

Упражнение 137. Две игральные кости

(*Решено. 43 строки*)

В данном упражнении мы будем симулировать 1000 выбрасываний игральных костей. Начнем с написания функции, выполняющей случайное выбрасывание двух обычных шестигранных костей. Эта функция не будет принимать входных параметров, а возвращать должна число, выпавшее в сумме на двух костях.

В основной программе реализуйте симуляцию тысячи выбрасываний костей. Программа должна хранить все результаты с частотой их выпадения. После завершения процесса должна быть показана итоговая таблица с результатами, похожая на ту, что представлена в табл. 6.1. Выразите частоту выпадения каждого из чисел в процентах вместе с ожидаемым результатом согласно теории вероятностей.

Таблица 6.1. Выбрасывание игральных костей

Исход	Процент симуляции	Ожидаемый процент
2	2,90	2,78
3	6,90	5,56
4	9,40	8,33
5	11,90	11,11
6	14,20	13,89
7	14,20	16,67
8	15,00	13,89
9	10,50	11,11
10	7,90	8,33
11	4,50	5,56
12	2,60	2,78

Упражнение 138. Текстовые сообщения

(*21 строка*)

Если помните, на старых мобильных телефонах текстовые сообщения набирались при помощи цифровых кнопок. При этом одна кнопка была ас-

социирована сразу с несколькими буквами, а выбор зависел от количества нажатий на кнопку. Однократное нажатие приводило к появлению первой буквы в соответствующем этой кнопке списке, последующие нажатия меняли ее на следующую. Список символов, ассоциированных с цифровой панелью, приведен в табл. 6.2.

Таблица 6.2. Символы, соответствующие кнопкам на старых телефонах

Кнопка	Символы
1	. , ? ! :
2	A B C
3	D E F
4	G H I
5	J K L
6	M N O
7	P Q R S
8	T U V
9	W X Y Z
0	Пробел

Напишите программу, отображающую последовательность кнопок, которую необходимо нажать, чтобы на экране телефона появился текст, введенный пользователем. Создайте словарь, сопоставляющий символы с кнопками, которые необходимо нажать, а затем воспользуйтесь им для вывода на экран последовательности кнопок в соответствии с введенным пользователем сообщением по запросу. Например, на ввод строки `Hello, World!` ваша программа должна откликнуться следующим выводом: `443355 555566611096667775553111`1. Удостоверьтесь, что ваша программа корректно обрабатывает строчные и прописные буквы. При преобразовании букв в цифры игнорируйте символы, не входящие в указанный перечень, такие как точка с запятой или скобки.

Упражнение 139. Азбука Морзе

(*15 строк*)

Азбука Морзе зашифровывает буквы и цифры при помощи точек и тире. В данном упражнении вам необходимо написать программу, в которой соответствие символов из азбуки Морзе будет храниться в виде словаря. В табл. 6.3 приведена та часть азбуки, которая вам понадобится при решении этого задания.

В основной программе вам необходимо запросить у пользователя строку. После этого программа должна преобразовать его в соответствующую последовательность точек и тире, вставляя пробелы между отдельными

символами. Символы, не представленные в таблице, можно игнорировать. Например, сообщение `Hello, World!` может быть представлено следующей последовательностью: …. . .–.. .–.. ––– .–– ––– .–. .–.. –..

Таблица 6.3. Азбука Морзе

Символ	Код	Символ	Код	Символ	Код	Символ	Код
A	.–	J	.–––	S	…	1	.––––
B	–…	K	–.–	T	–	2	..–––
C	–.–.	L	.–..	U	..–	3	…––
D	–..	M	––	V	…–	4	….–
E	.	N	–.	W	.––	5	…..
F	..–.	O	–––	X	–..–	6	–….
G	––.	P	.––.	Y	–.––	7	––…
H	….	Q	––.–	Z	––..	8	–––..
I	..	R	.–.	0	–––––	9	––––.

Примечание. Азбука Морзе была изобретена в XIX веке для передачи информации посредством телеграфа. Она широко используется и сегодня, более чем через 160 лет после ее создания.

Упражнение 140. Почтовые индексы

(*24 строки*)

Первый, третий и пятый символы в канадском почтовом индексе представляют собой буквы, а второй, четвертый и шестой – цифры. Провинцию или территорию, которой принадлежит индекс, можно определить по первому символу индекса, как показано в табл. 6.4. Символы D, F, I, O, Q, U, W и Z в настоящее время не используются в почтовых индексах Канады.

Второй символ в почтовом индексе определяет, расположен ли интересующий нас адрес в городе или в сельской местности. Если на этом месте стоит ноль, значит, это сельская местность, иначе город.

Напишите программу, которая будет запрашивать почтовый индекс у пользователя и отображать провинцию или территорию, которой он принадлежит, с указанием того, городская это территория или сельская. Например, если пользователь введет индекс `T2N1N4`, программа должна определить, что речь идет о городе на территории провинции Альберта. А индекс `X0A1B2` соответствует сельской местности в провинции Нунавут или в Северо-Западных территориях. Используйте словарь для хранения информации о соответствии первого символа индекса конкретной провинции или территории. Выведите на экран соответствующее сообщение

об ошибке, если индекс начинается с символа, который не используется для этих целей, или второй символ не является цифрой.

Таблица 6.4. Почтовые индексы Канады

Провинция/территория	Первая буква (буквы) индекса
Ньюфаундленд	A
Новая Шотландия	B
Остров Принца Эдуарда	C
Нью-Брансуик	E
Квебек	G, H и J
Онтарио	K, L, M, N и P
Манитоба	R
Саскачеван	S
Альберта	T
Британская Колумбия	V
Нунавут	X
Северо-Западные территории	X
Юкон	Y

Упражнение 141. Английская пропись

(*65 строк*)

Несмотря на то что популярность оплаты по чекам за последние годы серьезно снизилась, некоторые компании до сих пор используют этот способ для ведения взаиморасчетов с сотрудниками и поставщиками. Сумма на чеках обычно указывается дважды: один раз цифрами, второй – прописью на английском языке. Повторение суммы двумя разными формами записи призвано не позволить недобросовестным сотрудникам или поставщикам изменять сумму на чеках перед их обналичиванием.

В данном упражнении вам необходимо написать функцию, принимающую в качестве входного параметра число от 0 до 999 и возвращающую строку прописью. Например, если значение параметра будет равно 142, функция должна вернуть следующую строку: «one hundred forty two». Используйте один или несколько словарей вместо условных конструкций `if/elif/else` для выработки решения этой задачи. Напишите основную программу, в которой пользователь будет вводить числовое значение, а на экран будет выводиться соответствующая сумма прописью.

Упражнение 142. Уникальные символы

(*Решено. 16 строк*)

Напишите программу, определяющую и выводящую на экран количество уникальных символов во введенной пользователем строке. Например,

в строке `Hello, World!` содержится десять уникальных символов, а в строке `zzz` – один. Используйте словарь или набор для решения этой задачи.

Упражнение 143. Анаграммы

(*Решено. 39 строк*)

Анаграммами называются слова, образованные путем взаимной перестановки букв. В английском языке, например, анаграммами являются слова «live» и «evil», а в русском – «выбор» и «обрыв». Напишите программу, которая будет запрашивать у пользователя два слова, определять, являются ли они анаграммами, и выводить на экран ответ.

Упражнение 144. Снова анаграммы

(*48 строк*)

Понятие анаграмм не ограничивается словами, а может быть расширено до целых предложений. Например, строки «William Shakespeare» и «I am a weakish speller» являются полными анаграммами, если игнорировать пробелы и заглавные буквы.

Расширьте свою программу из упражнения 143, добавив возможность проверки на анаграммы целых фраз. При анализе не обращайте внимания на знаки препинания, заглавные буквы и пробелы.

Упражнение 145. Эрудит

(*Решено. 18 строк*)

В известной игре Эрудит (Scrabble™) каждой букве соответствует определенное количество очков. Общая сумма очков, которую получает игрок, составивший это слово, складывается из очков за каждую букву, входящую в его состав. Чем более употребимой является буква в языке, тем меньше очков начисляется за ее использование. В табл. 6.5 приведены все соответствия букв и очков из английской версии игры.

Таблица 6.5. Стоимость букв в английской версии игры Эрудит

Очки	Буквы
1	A, E, I, L, N, O, R, S, T и U
2	D и G
3	B, C, M и P
4	F, H, V, W и Y
5	K
8	J и X
10	Q и Z

Напишите программу, рассчитывающую и отображающую количество очков за собранное слово. Создайте словарь для хранения соответствий между буквами и очками и используйте его в своем решении.

Примечание. На игровом поле Эрудита присутствуют специальные клетки, удваивающие и утраивающие стоимость буквы или всего слова. В данном упражнении мы для простоты реализации проигнорируем этот факт.

Упражнение 146. Карточка лото

(*Решено. 58 строк*)

Карточка для игры в лото состоит из пяти колонок, в каждой из которых – пять номеров. Колонки помечены буквами `B`, `I`, `N`, `G` и `O`. Под каждой буквой могут быть номера в своем диапазоне из 15 чисел. А именно под буквой `B` могут присутствовать числа от 1 до 15, под `I` – от 16 до 30, под `N` – от 31 до 45 и т. д.

Напишите функцию, которая будет создавать случайную карточку лото и сохранять ее в словаре. Ключами словаря будут буквы `B`, `I`, `N`, `G` и `O`, а значениями – списки из пяти чисел, располагающихся в колонке под каждой буквой. Создайте еще одну функцию для отображения созданной карточки лото на экране со столбцами с заголовками. В основной программе создайте карту лото случайным образом и выведите ее на экран. Ваша программа должна запускаться только в том случае, если она не импортирована в виде модуля в другой файл.

Примечание. Как вы знаете, в настоящих карточках для игры в лото присутствуют пустые клетки в столбцах. Мы не будем реализовывать эту особенность в нашей программе.

Упражнение 147. Проверка карточки

(*102 строки*)

Карточка для игры в лото считается выигравшей, если в ней на одной линии расположились пять выпавших номеров. Обычно игроки зачеркивают номера на своих карточках. В данном упражнении мы будем обнулять в словаре выпавшие номера.

Напишите функцию, принимающую на вход карточку в качестве параметра. Если карточка содержит последовательность из пяти нулей (по вертикали, горизонтали или диагонали), функция должна возвращать `True`, в противном случае – `False`.

В основной программе вы должны продемонстрировать на примере работу функции, создав и отобразив несколько карточек с указанием того, какие из них выиграли. В вашем примере должно быть как минимум по одной карточке с выигрышем по вертикали, горизонтали и диагонали, а также карточки, на которые выигрыш не выпал. При решении этой задачи воспользуйтесь функциями из упражнения 146.

Подсказка. Поскольку отрицательных чисел на карточке лото быть не может, нахождение линии со всеми нулями сводится к поиску последовательности чисел, сумма которых равна нулю. Так вам будет легче решить это упражнение.

Упражнение 148. Играем в лото

(*88 строк*)

В данном упражнении мы напишем программу, выполняющую симуляцию игры в лото с одной картой. Начните с генерирования списка из всех возможных номеров для выпадения (от B1 до O75). После этого перемешайте номера в хаотичном порядке, воспользовавшись функцией `shuffle` из модуля `random`. Вытаскивайте по одному номеру из списка и зачеркивайте номера, пока карточка не окажется выигравшей. Проведите 1000 симуляций и выведите на экран минимальное, максимальное и среднее количество извлечений номеров, требующееся для выигрыша. При решении этой задачи вы можете воспользоваться функциями из упражнений 146 и 147.