ЕГЭ-8 (4 минуты)[a]

**Знание основных понятий и методов,
используемых при измерении количества информации**

Содержание

[a]Автор: Е. В. Ширяева. 2025. https://t.me/EVShiryaeva. Нулевой курс мехмата ЮФУ (группы). Проект «Отличный код» (мини-группы, индивидуальные занятия).

8.1 Работа со словами, упорядоченными по алфавиту

8.1.1 Примеры решений без использования компьютера

Пример 8.1. Все пятибуквенные слова, составленные из букв И, К, Т, О, записаны в алфавитном порядке. Вот начало списка:

1. ИИИИИ
2. ИИИИК
3. ИИИИО
4. ИИИИТ
5. ИИИКИ

... ...

Запишите порядковый номер слова КОТИК.

Решение. Упорядочим буквы по алфавиту и занумеруем их: И — 0, К — 1, О — 2, Т — 3. Если представлять буквы цифрами видно, что имеем дело с возрастающей последовательностью чисел, записанных в 4-ричной системе счисления:

№ п.п.	Слово	Число_4	Число_{10}
1.	ИИИИИ	00000	0
2.	ИИИИК	00001	1
3.	ИИИИО	00002	2
4.	ИИИИТ	00003	3
5.	ИИИКИ	00010	4
...	...	...	...
?.	**КОТИК**		

Слову КОТИК соответствует 4-ричное число 12301_4. Переводим это число в десятичную систему счисления:

$$12301_4 = 4^4 + 2 \cdot 4^3 + 3 \cdot 4^2 + 1 = 256 + 128 + 48 + 1 = 433$$

или с помощью Python

```
>>> int('12301', 4)
```

№ п.п.	Слово	Число_4	Число_{10}
1.	ИИИИИ	00000	0
2.	ИИИИК	00001	1
...	...	...	...
?.	КОТИК	12301	433

Из первой и последней колонок таблицы видно, что порядковый номер слова связан с десятичным представлением числа соотношением:

$$\text{№ п.п.} = \text{Число}_{10} + 1.$$

Слово КОТИК имеет порядковый номер 434.

Ответ: 434.

Пример 8.2. Все пятибуквенные слова, составленные из букв Н, И, К, Т, О, записаны в алфавитном порядке. Вот начало списка:

1. ИИИИИ
2. ИИИИК
3. ИИИИН
4. ИИИИО
5. ИИИИТ
6. ИИИКИ

... ...

Запишите слово, которое стоит на 150-м месте от начала списка.

Решение. Упорядочим буквы по алфавиту и занумеруем их: И — 0, К — 1, Н — 2, О — 3, Т — 4. Если представлять буквы цифрами видно, что имеем дело с возрастающей последовательностью чисел, записанных в 5-ричной системе счисления:

№ п.п.	Слово	Число_5	Число_{10}
1.	ИИИИИ	00000	0
2.	ИИИИК	00001	1
3.	ИИИИН	00002	2
4.	ИИИИО	00003	3
5.	ИИИИТ	00004	4
6.	ИИИКИ	00010	5
...	...	...	...
150.	**?**		

Из первой и последней колонок таблицы видно, что порядковый номер слова связан с десятичным представлением числа соотношением:

$$\text{№ п.п.} = \text{Число}_{10} + 1.$$

То есть нас интересует 5-ричное представление числа 149: 1044_5. Заменяем цифры буквами и не забываем, что по условию слова пятибуквенные: ИКИТТ.

Ответ: ИКИТТ.

8.2 Практикум по программированию. Часть А

8.2.1 Важно

«Почему не рекомендуется заменять заглавные буквы из задания на строчные»

```
w = 'ЙОУЁЖИК'
w2 = 'йоуёжик'

print(sorted(w))   # sorted(w) - сортировка строки w
print(sorted(w2))
```

Сравните результаты сортировки двух строк:

Ежегодно 29 ноября отмечается День буквы «Ё»

```
['Ё', 'Ж', 'И', 'Й', 'К', 'О', 'У']
['ж', 'и', 'й', 'к', 'о', 'у', 'ё']
```

На самом деле, всё ещё хитрее: среди заглавных букв буква «Ё» всегда стоит первой, а среди строчных — последней.

Вредные ЁЖИКИ

```
w = 'ЁЖИКВТУМАНЕ'
w2 = 'ёжиквтумане'

print(sorted(w))   # sorted(w) - сортировка строки w
print(sorted(w2))
```

Сравните результаты сортировки двух строк:

Вредные ЁЖИКИ

```
['Ё', 'А', 'В', 'Е', 'Ж', 'И', 'К', 'М', 'Н', 'Т', 'У']
['а', 'в', 'е', 'ж', 'и', 'к', 'м', 'н', 'т', 'у', 'ё']
```

Совет. Если в заданном алфавите нет буквы «Е», то замените букву «Ё» на «Е».

8.2.2 Поиск слова по заданному порядковому номеру

Пример 8.3. Все пятибуквенные слова, составленные из букв К, О, Р, А записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. АААААА
2. ААААК
3. ААААО
4. ААААР
5. АААКА

… …

Запишите слово, которое стоит на 250-м месте от начала списка.

Решение (вар. 1.) Использование функции `product()`. Если уверены, что не ошибётесь в номере с учётом сдвига, то после построения списка слов (строки 1, 3, 4) решение сведётся к одной строке 6.

Решение (вариант 1). Ответ: АРРОК

```
1 from itertools import product
2
3 abc = sorted('КОРА')  # обязательно упорядочены по алфавиту
4 t = [''.join(w) for w in product(abc, repeat=5)]
5
6 print(t[249])
```

Если есть хоть малейшие сомнения при решении предыдущим способом, то пишем цикл и ищем подходящий номер (строки 6–10).

Решение (вариант 2). Ответ: АРРОК

```
1  from itertools import product
2
3  abc = sorted('КОРА')  # обязательно упорядочены по алфавиту
4  t = [''.join(w) for w in product(abc, repeat=5)]
5
6  num = 0  # порядковый номер слова
7  for w in t:
8      num += 1
9      if num == 250:
10         print(w)
```

Задания для самостоятельной работы

№ 8.1 (➔)**.** Все 4-буквенные слова, составленные из букв М, А, Я, К, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. АААА
2. АААК
3. АААМ
4. АААЯ
5. ААКА

… …

Запишите слово, которое стоит на 229-м месте от начала списка.

№ 8.2 (➔)**.** Некоторый алфавит составлен из разных букв, содержащихся в слове МАКАКА. Буквы алфавита были использованы при составлении пятибуквенных слов. Вот начало списка:

1. ААААА
2. ААААК
3. ААААМ
4. АААКА

… …

Запишите слово, которое стоит на 187-м месте от начала списка.

№ 8.3 (➔)**.** Михаил составляет список из 4-буквенных слов, в состав которых входят только буквы В, Е, С, Н, А. Михаил расположил слова в обратном алфавитном порядке. Вот начало списка:

1. СССС
2. СССН
3. СССЕ
4. СССВ
5. ССCА
6. ССНС

… …

Запишите слово, которое стоит в этом списке под номером 130.

№ 8.4 (➔)**.** Все пятибуквенные слова, составленные из букв З, А, Я, Ц, Ё, Ж, М, Ы, Ш, Ь, записаны в алфавитном порядке. Вот начало списка:

1. ААААА
2. АААЁ
3. АААЖ
4. АААЗ
5. ААААМ

… …

Запишите слово, которое стоит на 47651-м месте от начала списка.

Задания для дополнительной отработки

№ 8.5 (➔). Все пятибуквеные слова, составленные из букв О, С, А записаны в алфавитном порядке. Вот начало списка:

1. ААААА
2. ААААО
3. ААААС
4. ААААОА
5. ААААОО

… …

Запишите слово, которое стоит на 145-м месте от начала списка.

№ 8.6 (➔). Все шестибуквеные слова, составленные из букв Ё, Ж, М, Ы, Ш, Ь записаны в алфавитном порядке. Вот начало списка:

1. ЁЁЁЁЁЁ
2. ЁЁЁЁЁЖ
3. ЁЁЁЁЁМ
4. ЁЁЁЁЁШ
5. ЁЁЁЁЁЫ
6. ЁЁЁЁЁЬ
7. ЁЁЁЁЖЁ

… …

Запишите слово, которое стоит на 21566-м месте от начала списка.

№ 8.7 (➔). Катя составляет список из 4-буквенных слов, в состав которых входят только буквы Б, У, Т, О, Н. Катя расположила слова в обратном алфавитном порядке. Вот начало списка:

1. УУУУ
2. УУУТ
3. УУУО
4. УУУН
5. УУУБ
6. УУТУ

… …

Запишите слово, которое стоит в этом списке под номером 517.

№ 8.8 (➔). Все пятибуквенные слова, составленные из букв А, Г, Д, Е, Ё, Ж, записаны в обратном алфавитном порядке. Вот начало списка:

1. ЖЖЖЖЖ
2. ЖЖЖЖЁ
3. ЖЖЖЖЕ
4. ЖЖЖЖД
5. ЖЖЖЖГ

… …

Запишите слово, которое стоит на 1461-м месте от начала списка.

8.2.3 Поиск по заданному слову

Пример 8.4. Все пятибуквенные слова, составленные из букв К, И, Н, О, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. ИИИИИ
2. ИИИИК
3. ИИИИН
4. ИИИИО
5. ИИИКИ

... ...

Запишите порядковый номер слова КОНИК.

Решение (вар. 1). Упорядочим буквы по алфавиту и занумеруем их, начиная с нуля:

$$\text{И} — 0;\ \text{К} — 1;\ \text{Н} — 2;\ \text{О} — 3.$$

Имеем 4-ую с.с. Закодируем с помощью цифр 4-ой с.с. слово «КОНИК»:

КОНИК

13201_4

Осталось перевести это число в 10-ую с.с. и вспомнить, что № п.п. $=$ Число$_{10} + 1$:

Решение (вариант 1). Ответ: 482

```
print(int('13201', 4) + 1)  # ответ 482
```

Решение (вар. 2). Составляем список всех слов и ищем позицию слова «КОНИК» в цикле.

Решение (вариант 2). Ответ: 482

```
from itertools import product

abc = sorted('КИНО')
t = [''.join(w) for w in product(abc, repeat=5)]

num = 0
for w in t:
    num += 1
    if w == 'КОНИК':
        print(num)
```

Решение (вар. 3). Составляем список всех слов и ищем позицию слова «КОНИК» с помощью метода `index()`. Не забываем про сдвиг № п.п. $=$ Число$_{10} + 1$.

Решение (вариант 3). Ответ: 482

```
from itertools import product

abc = sorted('КИНО')
t = [''.join(w) for w in product(abc, repeat=5)]

print(t.index('КОНИК') + 1)
```

Пример 8.5. Все пятибуквенные слова, составленные из букв А, Н, О, С, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. AAAAA
2. AAAAH
3. AAAAO
4. AAAAC
5. AAAHA

... ...

Какое количество слов находится между словами АНОНС и СОСНА (включая эти слова)?

Решение. Количество целых чисел на отрезке $[a; b]$ находится по формуле: $k = b - a + 1$.

Вспомогательные рассуждения к варианту 1 решения

АНОС	АНОНС	СОСНА
0123_4	01213_4	32310_4

Решение (вариант 1). Ответ: 846

```
>>> int('32310', 4) - int('01213', 4) + 1
    846
```

Решение (вариант 2). Ответ: 846

```
from itertools import product

abc = sorted('АНОС')
t = [''.join(w) for w in product(abc, repeat=5)]

num = 0
for w in t:
    num += 1
    if w == 'АНОНС':
        numAnons = num
    elif w == 'СОСНА':
        numSosna = num
        break  # прерывается цикл, т.к. все остальные слова не нужны

answer = numSosna - numAnons + 1
print(answer)   # ответ 846
```

Решение (вариант 3). Ответ: 846

```
from itertools import product

abc = sorted('АНОС')
t = [''.join(w) for w in product(abc, repeat=5)]

answer = t.index('СОСНА') - t.index('АНОНС') + 1
print(answer)
```

Пример 8.6. Все четырёхбуквенные слова, составленные из букв Б, Е, Л, К, А, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. ААААА
2. ААААБ
3. ААААЕ
4. ААААК
5. ААААЛ
6. ААБАА

… …

Под каким номером в списке стоит последнее слово с нечётным номером, которое не начинается с буквы Л и содержит ровно две буквы А?

Решение (вар. 1). Упорядочим буквы по алфавиту и занумеруем их, начиная с нуля:

А — 0; Б — 1; Е — 2; К — 3; Л — 4.

Имеем 5-ую с.с. Последнее четырёхбуквенное слово, удовлетворяющее «условиям не начинается с буквы Л и содержит ровно две буквы А»: «КЛАА»

КЛАА

3400_5

```
>>> int('3400', 5) + 1
476
```

но номер у этого слова чётный!

Вторая попытка подобрать последнее четырёхбуквенное слово, удовлетворяющее условиям «не начинается с буквы Л и содержит ровно две буквы А»: «ККАА»

ККАА

3300_5

```
>>> int('3300', 5) + 1
451
```

и номер у этого слова нечётный. Ответ: 451

Пример 8.7. Все шестибуквенные слова, в составе которых могут быть только буквы Р, О, Б, У, С, Т, А, К, Ф, М записаны в алфавитном порядке и пронумерованы, начиная с 1. Все буквы в слове различны.

Ниже приведено начало списка.

1. АБКМОР
2. АБКМОС
3. АБКМОТ
4. АБКМОУ
5. АБКМОФ
6. АБКМРО

...

Найти номер последнего слова, содержащего максимальное количество гласных букв.

Решение. Вариант 1: использование метода `index()`, что подразумевает знание разыскиваемого слова. Это слово легко определить, но также легко и промахнуться.

Определение разыскиваемого слова: отсортируем буквы заданного слова «РОБУСТАКФМ» в обратном алфавитном порядке: «ФУТСРОМКБА». Нужно отобрать «первые» шесть букв так, чтобы в слово входили все гласные буквы: «ФУТСОА».

Решение (вариант 1). Ответ: 151191

```
from itertools import permutations

abc = sorted('РОБУСТАКФМ')  # Сортировка букв слова обязательна.
t = [''.join(w) for w in permutations(abc, r=6)]

print(t.index('ФУТСОА') + 1)
```

Решение. Вариант 2 (не знаем разыскиваемое слово). В программе разыскиваем последнее слово, в котором есть все три гласные буквы.

Решение (вариант 2). Ответ: 151191

```
from itertools import permutations

abc = sorted('РОБУСТАКФМ')  # Сортировка букв слова обязательна.
t = [''.join(w) for w in permutations(abc, r=6)]

num = 0   # инициализация номера слова
for w in t:
    num += 1
    if 'У' in w and 'О' in w and 'А' in w:
        answer = num, w   # кортеж: номер, слово (для контроля)

print(answer)  # (151191, 'ФУТСОА')
```

Задания для самостоятельной работы

№ 8.9 (➔). Все четырёхбуквенные слова, составленные из букв Ж, И, Р, А, Ф, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. АААА
2. АААЖ
3. АААИ
4. АААР
5. АААФ

… …

Запишите порядковый номер слова ФАРА.

№ 8.10 (➔). Все шестибуквенные слова, составленные из букв Н, О, Р, К, А, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. ААААА
2. АААААК
3. АААААН
4. АААААО
5. АААААР
6. АААААКА

… …

Какое количество слов находится между словами КОРОНА и РОАНОК (включая эти слова)?

№ 8.11 (➔). Все пятибуквенные слова, составленные из букв Б, Е, Л, О, К, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. ББББ
2. ББББЕ
3. ББББК
4. ББББЛ
5. ББББО
6. ББББЕБ

… …

Под каким номером в списке стоит последнее слово с нечётным номером, которое не начинается с буквы О и содержит ровно две буквы Е?

№ 8.12 (➔). Все пятибуквенные слова, составленные из букв А, С, П, Е, К, Т, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. ААААА
2. ААААЕ
3. ААААК
4. ААААП
5. ААААС

… …

Сколько существует слов, стоящих в списке на позициях с чётными номерами, в которых нет двух подряд идущих гласных букв?

№ 8.13 (➔). Все шестибуквенные слова, составленные из букв слова «ПОЗДРАВИТЕЛЬНЫЙ», записаны в алфавитном порядке и пронумерованы начиная с 1. Все буквы в слове различны. Вот начало списка:

1. АВДЕЗИ
2. АВДЕЗЙ
3. АВДЕЗЛ
4. АВДЕЗН
5. АВДЕЗО

… …

Найти номер последнего слова, содержащего максимальное количество гласных букв и не начинающегося с мягкого знака.

№ 8.14 (★; ➔). Все пятибуквенные слова, составленные из букв слова «ПРАВИТЕЛЬ», записаны в алфавитном порядке и пронумерованы начиная с 1. Все буквы в слове различны. Вот начало списка:

1. АВЕИЛ
2. АВЕИП
3. АВЕИР
4. АВЕИТ
5. АВЕИЬ

… …

Сколько существует слов, стоящих на позициях с чётными номерами, в которых нет трёх подряд идущих согласных букв[b]?

Задания для дополнительной отработки

№ 8.15 (➔). Все четырёхбуквенные слова, составленные из букв Б, А, Р, О, Н, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. АААА
2. АААБ
3. АААН
4. АААО

… …

Запишите порядковый номер слова НОРА.

№ 8.16 (➔). Все шестибуквенные слова, составленные из букв К, А, М, И, Н, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. АААААА
2. АААААИ
3. АААААК
4. АААААМ

… …

Запишите порядковый номер слова ИММИКН[c].

[b] В алфавите русского языка 33 буквы: 10 гласных (а, е, ё, и, о, у, ы, э, ю, я); 21 согласная буква; ъ и ь.

[c] ИММ и КН — Институт математики, механики и компьютерных наук Южного федерального университета.

№ 8.17 (➔). Все шестибуквенные слова, составленные из букв К, О, Р, М, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. КККККК
2. КККККМ
3. КККККО
4. КККККР
5. ККККМК

… …

Какое количество слов находятся между словами ОКОРОК и РОКОКО (включая эти слова)?

№ 8.18 (➔). Все пятибуквенные слова, составленные из букв Л, Е, Т, О, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. ЕЕЕЕЕ
2. ЕЕЕЕЛ
3. ЕЕЕЕО
4. ЕЕЕЕТ
5. ЕЕЕЛЕ

… …

Под каким номером в списке стоит первое слово с чётным номером, которое не начинается с буквы Е и содержит ровно две буквы Т, не стоящие рядом?

№ 8.19 (➔). Все шестибуквенные слова, составленные из букв Б, И, Р, К, А, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. АААААА
2. АААААБ
3. АААААИ
4. АААААК
5. АААААР
6. АААААБА

… …

Под каким номером в списке стоит последнее слово с нечётным номером, которое не начинается с буквы Р и содержит ровно три буквы А?

№ 8.20 (➔). Все пятибуквенные слова, составленные из букв В, Е, К, Т, О, Р, записаны в алфавитном порядке и пронумерованы начиная с 1. Вот начало списка:

1. ВВВВВ
2. ВВВВЕ
3. ВВВВК
4. ВВВВО
5. ВВВВР

… …

Сколько существует слов, стоящих в списке на позициях с чётными номерами, в которых нет двух одинаковых подряд идущих букв?

№ 8.21 (➔)**.** Все пятибуквенные слова, составленные из букв П, Р, А, В, И, Т, Е, Л, Ь, записаны в алфавитном порядке и пронумерованы начиная с 1. Все буквы в слове различны. Вот начало списка:

1. АВЕИЛ
2. АВЕИП
3. АВЕИР
4. АВЕИТ
5. АВЕИЬ

… …

Найти номер первого слова, не содержащего гласных букв.

№ 8.22 (★; ➔)**.** Все пятибуквенные слова, составленные из букв слова «ТРЕУГОЛЬНИК», записаны в алфавитном порядке и пронумерованы начиная с 1. Все буквы в слове различны. Вот начало списка:

1. ГЕИКЛ
2. ГЕИКН
3. ГЕИКО
4. ГЕИКР
5. ГЕИКТ

… …

Сколько существует слов, стоящих на позициях с чётными номерами, в которых есть только три согласные буквы[d], при этом они стоят подряд.

Например, в слове ИЕКТР есть три стоящие подряд согласные буквы, а в неподходящем слове РОЛИК — их три, но они не стоят рядом.

[d] В алфавите русского языка 33 буквы: 10 гласных (а, е, ё, и, о, у, ы, э, ю, я); 21 согласная буква; ъ и ь.

8.3 Работа со словами (комбинаторика)

8.3.1 Примеры условий для отбора слов (Python)

В алфавите русского языка 33 буквы: 10 гласных (а, е, ё, и, о, у, ы, э, ю, я); 21 согласная буква; ъ и ь.

В заданиях ЕГЭ-8 обычно задаётся кириллический алфавит небольшой мощности, т. е. с небольшим количеством символов в алфавите. Например, мощность алфавита, составленного из букв слова ЧАСОВЩИК, равна восьми. Примеры условий ниже приведены для этого алфавита.

а) слово начинается на сочетание ЧА:

`w[0] == 'Ч' and w[1] == 'А'` или `w[:2] == 'ЧА'`

б) слово заканчивается на сочетание ЧА:

`w[-2] == 'Ч' and w[-1] == 'А'` или `w[-2:] == 'ЧА'`

в) в слове есть буква А: `'А' in w`

г) в слове нет буквы А: `'А' not in w`

д) количество букв А в слове равно двум: `w.count('А') == 2`

е) в слове не менее двух букв А: `w.count('А') >= 2`

ж) в слове все буквы согласные:

ж.1) `'А' not in w and 'И' not in w and 'О' not in w`
ж.2) `all(c not in w for c in 'АИО')` (нет «неприятеля» в слове)
ж.3) `all(c in 'ЧСВЩК' for c in w)` (все символы хорошие)

з) в слове не более двух гласных букв:

`w.count('А') + w.count('И') + w.count('О') <= 2`. Здесь воспользовались тем, что гласных букв в заданном алфавите мало. Иначе лучше было организовать цикл.

и) слово начинается и заканчивается одинаковыми символами: `w[0] == w[-1]`

к) слово начинается и заканчивается согласной буквой:

`w[0] in 'ЧСВЩК' and w[-1] in 'ЧСВЩК'`

л) слово-палиндром: `w == w[::-1]`

м) все буквы в слове упорядочены по алфавиту: `list(w) == sorted(w)`

н) одинаковые буквы не стоят рядом:

```
all(w[i] != w[i+1] for i in range(0, len(w)-1))
```

одинаковые буквы из заданного алфавита не стоят рядом:

`all(c + c not in w for c in 'ЧСВЩК')`.

8.4 Практикум по программированию. Часть B

8.4.1 Прямое (декартово) произведение

Функция `product(<алфавит>, repeat=<длина слова>)` используется, если в слове допускаются повторения букв.

`<длина слова>` может быть любой (> 0). Если `<длина слова>` не указывается, то составляются слова максимальной длины.

Пример 8.8. Из букв Б, Е, Л, К, А составляются трёхбуквенные слова, в которых буква К появляется ровно один раз. Каждая из других допустимых букв может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

Решение. Сортировка букв слова не требуется, так как важно количество подходящих слов, а не их порядок. Но сортировка букв поможет создавать «предсказуемые» списки — мы будем знать, что ожидать в начале списка, а что в конце.

Решение (вариант 1, рекомендовано). Ответ: 48

```
from itertools import product

abc = sorted('БЕЛКА')
t = [''.join(w) for w in product(abc, repeat=3)]

# собираем список подходящих слов
t = [w for w in t if w.count('К') == 1]

print(len(t))
```

Решение (вариант 2). Ответ: 48

```
from itertools import product

abc = sorted('БЕЛКА')
t = [''.join(w) for w in product(abc, repeat=3)]

cnt = 0  # счётчик подходящих слов
for w in t:
    if w.count('К') == 1:
        cnt += 1
print(cnt)
```

Пример 8.9. Из букв Т, Е, М, А составляются шестибуквенные слова, в которых буква М появляется хотя бы два раза. Каждая из других допустимых букв может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

Решение (вариант 1, рекомендовано). Ответ: 1909

```
from itertools import product

abc = sorted('TEMA')
t = [''.join(w) for w in product(abc, repeat=6)]

t = [w for w in t if w.count('M') >= 2]
print(len(t))
```

Решение (вариант 2). Ответ: 1909

```
from itertools import product

abc = sorted('TEMA')
t = [''.join(w) for w in product(abc, repeat=6)]

cnt = 0
for w in t:
    if w.count('M') >= 2:
        cnt += 1
print(cnt)
```

Пример 8.10. Из букв Б, А, Р, К, А, С составляются пятибуквенные слова. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

Решение. При использовании `product()` из алфавита следует убрать все повторения букв.

Решение. Ответ: 3125

```
from itertools import product

abc = 'БАРКС'  # при использовании product убрать все повторы букв
t = [''.join(w) for w in product(abc, repeat=5)]

print(len(t))  # Ответ: 3125
```

Пример 8.11. Составляются пятибуквенные слова из букв В, Л, А, С, Т, Ь. Слово не может начинаться на букву Ь и одинаковые буквы не могут стоять рядом. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

Решение (вариант 1). Ответ: 3125

```
from itertools import product

abc = 'ВЛАСТЬ'
t = [''.join(w) for w in product(abc, repeat=5)]
#print(len(t))  # 7776 все слова

t = [w for w in t if w[0] != 'Ь']
#print(len(t))  # 6480 слова, не начинающиеся на 'Ь'

# отбор слов, в которых одинаковые буквы не стоят рядом
t = [w for w in t if all(c + c not in w for c in abc)]

print(len(t))
```

Второй вариант решения вместо функции `all()` использует пользовательскую функцию для проверки условия «одинаковые буквы не стоят рядом». Это удобнее, так как даёт возможность отдельно протестировать функцию и попрактиковаться в написании подпрограмм.

Решение (вариант 2). Ответ: 3125

```
from itertools import product

def ok(w):  # Функция для проверки условия "одинаковые буквы не стоят рядом"
    for c in abc:  # идём по алфавиту
        if c + c in w:  # составляем пары одинаковых символов
            return False
    return True

# тестирование функции ok(w)
#print(ok('ВАЛЬС'))   # True
#print(ok('ВАЛЛЬС'))  # False

abc = 'ВЛАСТЬ'
t = [''.join(w) for w in product(abc, repeat=5)]

# список слов, удовлетворяющих сразу двум условиям
t = [w for w in t if w[0] != 'Ь' and ok(w)]

print(len(t))
```

Пример 8.12. Составляются пятибуквенные слова из букв Б, А, Р, С, И, К. Сколько можно составить различных слов, в которых гласные и согласные буквы чередуются? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

Решение. Идеи функций для проверки условия «гласные и согласные буквы в строке чередуются»:

ok(w)	ok2(w)
заменяем все гласные буквы из заданного алфавита на «Р», все согласные — на «А»; в новой строке не должно быть «АА» и «РР»	в цикле по индексам строки ищем «нарушителей»: соседние буквы согласные или гласные

Функции, проверяющие условие «гласные и согласные буквы чередуются»

```
def ok(w):
    w = w.replace('Б', 'Р').replace('С', 'Р').replace('К', 'Р')
    w = w.replace('И', 'А')
    return 'АА' not in w and 'РР' not in w

def ok2(w):
    # в цикле ищем "нарушителей": соседние буквы согласные или гласные
    for i in range(len(w)-1):
        if w[i] in 'БРСК' and w[i+1] in 'БРСК'\
           or w[i] in 'АИ' and w[i+1] in 'АИ':
               return False
    return True

# тестирование функций ok(w), ok(w)
print(ok('БАРСИК'), ok2('БАРСИК'))  # False False
print(ok('БАРИК'), ok2('БАРИК'))    # True True
print(ok('БАИК'), ok2('БАИК'))      # False False
print(ok('ИРИС'), ok2('ИРИС'))      # True True
```

Решение примера 8.12. Ответ: 384

```
from itertools import product

def ok(w):
    w = w.replace('Б', 'Р').replace('С', 'Р').replace('К', 'Р')
    w = w.replace('И', 'А')
    return 'АА' not in w and 'РР' not in w

abc = sorted('БАРСИК')
t = [''.join(w) for w in product(abc, repeat=5)]

t = [w for w in t if okV(w)]

print(len(t))
```

Пример 8.13. Составляются слова-палиндромы из букв М, У, Р, З, И, К, длиной не более пяти символов. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

Решение. Длина слов может быть 1, 2, 3, 4, 5. Например, К, ИИ, МИМ,...

Решение. Ответ: 300

```
from itertools import product

abc = sorted('МУРЗИК')
cnt = 0
for k in range(1, 6):
     t = [''.join(w) for w in product(abc, repeat=k)]
     t = [w for w in t if w == w[::-1]]  # палиндромы
     #print(t)  # для контроля отобранных слов
     #input()   # пауза; программа ждёт нажатия клавиши Enter
     cnt += len(t)

print(cnt)  # 300 ok
```

Пример 8.14. Составляются восьмибуквенные слова-палиндромы из букв Г, Р, А, Д, У, С, Н, И, К, не начинающиеся на гласную букву. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

Решение. Будем составлять слова, содержащие половину палиндрома, т. е. 4-буквенные слова. Сами палиндромы нам не нужны, можно в список добавлять любую информацию, например, единицу. Ниже показан способ создания палиндромов в учебных целях.

Решение (вариант 1). Ответ: 4374

```
from itertools import product

abc = 'ГРАДУСНИК'
t = [''.join(w) for w in product(abc, repeat=4)]
print(len(t))  # 6561 - длина списка 4-буквенных слов

t = [w + w[::-1] for w in t if w[0] in 'ГРДСНК']  # "вручную" составляем палиндромы

print(len(t))
```

Вариант решения без использования библиотеки `itertools`. Здесь тоже можно формировать слова-палиндромы, но для решения задачи достаточно использовать счётчик.

Решение (вариант 2). Ответ: 4374

```
abc = 'ГРАДУСНИК'
cnt = 0
for c1 in 'ГРДСНК':
    for c2 in abc:
        for c3 in abc:
            for c4 in abc:
                cnt += 1

print(cnt)
```

Решение с построением всего списка 8-буквенных слов здесь будет самым неудачным.

Решение (вариант 3). Ответ: 4374

```
from itertools import product

abc = 'ГРАДУСНИК'
t = [''.join(w) for w in product(abc, repeat=8)]  # ДОЛГО!

print(len(t)) # 43046721 - длина списка 8-буквенных слов

t = [w for w in t if w[0] in 'ГРДСНК' and w == w[::-1]]

print(len(t))
```

Задания для самостоятельной работы

№ 8.23 (➔)**.** Из букв С, Т, Р, Е, Л, К, А составляются четырёхбуквенные слова, в которых буква К появляется не более одного раза. Каждая из других допустимых букв может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.24 (➔)**.** Из букв В, Е, Н, И, К составляются трёхбуквенные слова, в которых буква Н появляется только один раз и не на последнем месте. Каждая из других допустимых букв может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.25 (➔)**.** Из букв Б, У, Л, К, А составляются пятибуквенные слова, в которых не более двух гласных букв. Каждая из других допустимых букв может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.26 (➔)**.** Из букв М, Ы, Ш, О, Н, О, К составляются четырёхбуквенные слова. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.27 (➔)**.** Составляются пятибуквенные слова из букв П, Р, У, С, А, К. Слово не может начинаться на букву С и не может заканчиваться двумя одинаковыми буквами. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.28 (➔)**.** Составляются шестибуквенные слова из букв Р, А, Д, И, У, С. Сколько можно составить различных слов, в которых гласные и согласные буквы чередуются? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.29 (➔)**.** Составляются слова из букв Б, О, Р, Т, И, К, длиной не более пяти символов. Допустимые слова начинаются с гласной буквы. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.30 (➔)**.** Составляются слова-палиндромы из разных букв слова РОСТОВЧАНЕ длиной не менее трёх и не более семи символов. Допустимые слова начинаются с гласной буквы и содержат не более двух согласных букв. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

Задания для дополнительной отработки

№ 8.31 (➔). Из букв Б, А, Т, О, Н составляются пятибуквенные слова, которые начинаются и заканчиваются одинаковыми символами. Каждая из допустимых букв может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.32 (➔). Из букв С, П, А, Р, Т, А составляются четырёхбуквенные слова. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.33 (➔). Составляются пятибуквенные слова из букв Р, А, Д, И, У, С. Сколько можно составить различных слов, в которых никакие две гласные и две согласные не стоят рядом? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.34 (➔). Из букв Б, У, Т, Ы, Л, К, А составляются пятибуквенные слова, в которых все буквы упорядочены по алфавиту. Каждая из допустимых букв может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.35 (➔). Составляются трёхбуквенные слова-палиндромы из букв Г, Р, А, Д, У, С. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.36 (➔). Составляются пятибуквенные слова-палиндромы из букв Г, Р, А, Д, У, С, Н, И, К. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

№ 8.37 (➔). Составляются слова-палиндромы из букв Р, У, Д, Н, И, К, длиной не более шести символов. Каждая буква может встречаться в слове любое количество раз или не встречаться совсем. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

8.4.2 Перестановки

Функция `permutations(<алфавит>, r=<длина слова>)` используется, если слова получаются перестановкой букв.

`<длина слова>` не может быть больше количества символов в `<алфавите>`. Если `<длина слова>` не указана, то составляются слова максимальной длины.

Пример 8.15. Составляются семибуквенные слова перестановкой букв Т, Ю, Л, Ь, П, А, Н. Слово не может начинаться на букву Ь. Сколько различных слов можно составить? Словом считается любая допустимая последовательность букв, не обязательно осмысленная.

Решение. Ответ: 3792

```
from itertools import permutations

abc = sorted('ТЮЛЬПАН')
t = [''.join(w) for w in permutations(abc, r=7)]

answer = [w for w in t if w[0] != 'Ь']

print(len(answer))  #  3792 ok
```

Пример 8.16. Составляются слова перестановкой букв Ф, И, А, Л, К, А. Сколько различных слов можно составить?

Решение. Ответ: 360

```
from itertools import permutations

abc = 'ФИАЛКА'  # повторяющиеся буквы убирать из алфавита нельзя!
t = [''.join(w) for w in permutations(abc)]
# В полученном списке будет два слова ФИАЛКА
print(t.count('ФИАЛКА'))

answer = set(t)  # преобразуем в множество => убираем повторения типа ФИАЛКа и ФИаЛКА

print(len(answer))
```

Задания для самостоятельной работы

№ 8.38 (➔). Составляются слова перестановкой букв Б, А, Л, А, Л, А, Й, К, А. Сколько различных слов можно составить?

№ 8.39 (➔). Составляются пятибуквенные слова перестановкой букв Т, О, Р, М, О, З, О, К. Сколько различных слов можно составить?

№ 8.40 (➔). Составляются шестибуквенные слова перестановкой букв С, А, К, С, А, У, Л. При этом нельзя ставить рядом две гласные буквы. Сколько различных слов можно составить?

Задания для дополнительной отработки

№ 8.41 (➔). Составляются слова перестановкой букв К, О, Л, О, К, О, Л. Сколько различных слов, начинающихся на букву О, можно составить?

№ 8.42 (➔). Составляются пятибуквенные слова перестановкой букв К, У, К, О, Л, К, А. Сколько различных слов можно составить?

№ 8.43 (➔). Составляются шестибуквенные слова перестановкой букв О, С, К, О, Л, О, К. При этом нельзя ставить рядом две гласные буквы. Сколько различных слов можно составить?

8.5 Работа с числами

Важно помнить, что

- числа не должны иметь в старшем разряде (первом слева) нуля, это незначащий ноль. Обязательно делайте отбор таких чисел `t = [w for w in t if w[0] != '0']` сразу после формирования списка `t`. Это позволит избежать ошибки «опять забыл(а) убрать незначащий ноль»;
- понятия «возрастание» ($w_i < w_{i+1}$) и «неубывание» ($w_i \leqslant w_{i+1}$) — разные.

  Например, в числе 12578 цифры упорядочены по возрастанию (и по неубыванию).

  А в числе 125578 цифры упорядочены по неубыванию (но не по возрастанию!).

8.5.1 Примеры условий для отбора чисел (Python)

Будем рассматривать десятичные числа.

А) число начинается на чётную цифру: `w[0] in '2468'`.

Б) число заканчивается на две одинаковые цифры: `w[-2] == w[-1]`

В) в числе есть цифра 7: `'7' in w`

Г) в числе нет цифры 7: `'7' not in w`

Д) количество цифр 0 в числе равно двум: `w.count('0') == 2`

Е) в числе не менее двух цифр 0: `w.count('0') >= 2`

Ж) в числе все цифры нечётные:
Ж.1) `not ('0' in w or '2' in w or '4' in w or '6' in w or '8' in w)`
Ж.2) `all(x in '13579' for x in w)`
Ж.3) `all(x not in '02468' for x in w)`

З) в числе не более двух нечётных цифр:
З.1) `w.count('1') + w.count('3') + w.count('5') + w.count('7') + w.count('9') <= 2`
З.2) `len([x in '13579' for x in w]) <= 2`

И) число начинается и заканчивается одинаковыми цифрами: `w[0] == w[-1]`

К) число начинается и заканчивается нечётной цифрой:

```
w[0] in '13579' and w[-1] in '13579'
```

Л) сумма цифр числа чётная: `sum(int(x) for x in w) % 2 == 0`

М) все цифры в числе упорядочены по неубыванию (например, 1335):

```
list(w) == sorted(w)
```

Н) все цифры в числе упорядочены по возрастанию (например, 135):
`all(w[i] < w[i+1] for i in range(0, len(w)-1))`

О) одинаковые цифры не стоят рядом:

```
all(w[i] != w[i+1] for i in range(0, len(w)-1))
```

8.6 Практикум по программированию. Часть С

8.6.1 Примеры решения задач

Пример 8.17. Сколько существует десятичных пятизначных чисел, кратных пяти?

Решение. Ответ: 18000

```
from itertools import product

abc = '0123456789'  # цифры 10-ной с.с.
t = [''.join(w) for w in product(abc, repeat=5)]
t = [w for w in t if w[0] != '0']  # лучше сделать этот отбор отдельно

answer = [w for w in t if w[-1] == '0' or w[-1] == '5']

print(len(answer))
```

Пример 8.18 (тип «Пробный КЕГЭ-22»). Сколько существует семеричных шестизначных чисел, которые не могут начинаться с цифр 1 и 5 и не содержат пар 22 и 66 одновременно?

Решение. Ответ: 66838

```
from itertools import product

abc = '0123456'  # цифры 7-ной с.с.
t = [''.join(w) for w in product(abc, repeat=6)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if w[0] not in '15' and\
          not ('22' in w and '66' in w)]

print(len(answer))
```

Пример 8.19. Сколько существует девятеричных пятизначных чисел, которые не могут начинаться с цифр 6 и 8 и не должны содержать пары соседних одинаковых цифр.

Решение. Ответ: 24576

```
from itertools import product

abc = '012345678'  # цифры 9-ной с.с.
t = [''.join(w) for w in product(abc, repeat=5)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if w[0] not in '68' and\
          all(w[i] != w[i+1] for i in range(0, len(w)-1))]

print(len(answer))
```

Пример 8.20 (тип «ЕГЭ-22»). Сколько существует девятеричных пятизначных чисел, которые не начинаются с нечётной цифры, не оканчиваются цифрой 3 или 8, а также содержат в своей записи не более одной цифры 1?

Решение. Ответ: 18944

```
from itertools import product

abc = '012345678'
t = [''.join(w) for w in product(abc, repeat=5)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if w[0] not in '1357' and\
          w[-1] not in '38' and w.count('1') <= 1]

print(len(answer))
```

Пример 8.21 (тип «ЕГЭ-22»). Сколько существует девятеричных шестизначных чисел, в записи которых ровно одна цифра 4 и никакая нечётная цифра не стоит рядом с цифрой 4?

Решение. Первый вариант программы использует легко тестируемую функцию ok(w) для проверки факта «никакая нечётная цифра не стоит рядом с цифрой 4».

Решение (вариант 1). Ответ: 58368

```
from itertools import product

def ok(w):
    for c in '1357':
        if c + '4' in w or '4' + c in w:
            return False
    return True

abc = '012345678'  # цифры 9-ной с.с.
t = [''.join(w) for w in product(abc, repeat=6)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if w.count('4') == 1 and ok(w)]

print(len(answer))
```

Второй вариант программы использует генератор списка условий. Не рекомендуется новичкам.

Решение (вариант 2). Ответ: 58368

```
from itertools import product

abc = '012345678'  # цифры 9-ной с.с.
t = [''.join(w) for w in product(abc, repeat=6)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if w.count('4') == 1 and\
          all('4' + c not in w and c + '4' not in w for c in '1357')]

print(len(answer))
```

Пример 8.22. Сколько существует шестнадцатеричных четырёхзначных чисел, в записи которых никакие две чётные и нечётные цифры не стоят рядом?

Решение. В примере 8.12 приведены функции, которые почти решают задачу по проверке факта «никакие две чётные и нечётные цифры не стоят рядом». «Почти решают», потому что в примере 8.12 речь шла о чередовании гласных и согласных букв. Эти функции легко переделать под цифры. Ниже приведены разные варианты проверки основного условия задачи.

Первый вариант программы использует легко тестируемую функцию `ok(w)` для проверки факта «никакие две чётные и нечётные цифры не стоят рядом».

Решение (вариант 1, рекомендую). Ответ: 7680

```
from itertools import product

def ok(w):
    for c in '0468ace':  # все чётные цифры заменяем на 2
        w = w.replace(c, '2')
    for c in '3579bdf':  # все нечётные цифры заменяем на 1
        w = w.replace(c, '1')
    return '11' not in w and '22' not in w

# тестирование функции ok(w)
#print(ok('16341')) # True
#print(ok('1631'))  # False
#print(ok('2221'))  # False

abc = '0123456789abcdef'  # цифры 16-ной с.с.
t = [''.join(w) for w in product(abc, repeat=4)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if ok(w)]

print(len(answer))
```

Второй вариант программы содержит вариант функции ok(w) с проверкой соседних элементов.

Решение (вариант 2). Ответ: 7680

```
from itertools import product

def ok(w):
    # в цикле ищем ”нарушителей”: соседние цифры имеют одинаковую чётность
    for i in range(len(w)-1):
        if int(w[i], 16) % 2 == int(w[i+1], 16) % 2:
            return False
    return True

# тестирование функции ok(w)
#print(ok(’16341’)) # True
#print(ok(’1631’))  # False
#print(ok(’2221’))  # False

abc = ’0123456789abcdef’  # цифры 16-ной с.с.
t = [’’.join(w) for w in product(abc, repeat=4)]
t = [w for w in t if w[0] != ’0’]

answer = [w for w in t if ok(w)]

print(len(answer))
```

Третий вариант программы использует генератор списка условий. Не рекомендуется новичкам.

Решение (вариант 3). Ответ: 7680

```
from itertools import product

abc = ’0123456789abcdef’  # цифры 16-ной с.с.
t = [’’.join(w) for w in product(abc, repeat=4)]
t = [w for w in t if w[0] != ’0’]

answer = [w for w in t if all(int(w[i], 16) % 2 != int(w[i+1], 16) % 2 \
                             for i in range(len(w)-1))]

print(len(answer))
```

Пример 8.23. Сколько существует десятичных шестизначных чисел, в которых все цифры идут в порядке убывания, при этом чётные и нечётные цифры чередуются?

Решение. Так как чётные и нечётные цифры чередуются, то условие «в порядке убывания» можно проверить с помощью `list(w) == sorted(w, reverse=True)`. Функция `sorted()` **здесь** сортирует в порядке невозрастания, т.е. подразумевает, что две рядом стоящие цифры могут быть одинаковыми, но этот случай в задаче исключён.

Первый вариант программы использует легко тестируемую функцию `ok(w)` для проверки факта «никакие две чётные и нечётные цифры не стоят рядом».

Решение (вариант 1). Ответ: 35

```
from itertools import product

def ok(w):
    # в цикле ищем "нарушителей": соседние цифры имеют одинаковую чётность
    for i in range(len(w)-1):
        if int(w[i]) % 2 == int(w[i+1]) % 2:
            return False
    return True

# тестирование функции ok(w)
#print(ok('16341')) # True
#print(ok('1631'))  # False
#print(ok('2221'))  # False

abc = '0123456789'  # цифры 10-ной с.с.
t = [''.join(w) for w in product(abc, repeat=6)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if ok(w) and list(w) == sorted(w, reverse=True)]

print(len(answer))
```

Второй вариант программы использует генератор списка условий. Не рекомендуется новичкам.

Решение (вариант 2). Ответ: 35

```
from itertools import product

abc = '0123456789'  # цифры 10-ной с.с.
t = [''.join(w) for w in product(abc, repeat=6)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if all(int(w[i]) % 2 != int(w[i+1]) % 2 \
                             and list(w) == sorted(w, reverse=True) \
                             for i in range(len(w)-1))]

print(len(answer))
```

Пример 8.24. Сколько существует восьмеричных пятизначных чисел, не содержащих в своей записи цифру 1, в которых все цифры различны и никакие две чётные или две нечётные цифры не стоят рядом?

Решение. Условие «все цифры различны» подразумевает построение списка перестановок. Ниже для этой цели использована функция `permutations()`. Ниже приведены два варианта решения.

Решение (вариант 1, рекомендую). Ответ: 252

```
from itertools import permutations

def ok(w):
    # все чётные 8-ные цифры заменяем на 2
    w = w.replace('0', '2').replace('4', '2').replace('6', '2')
    # все нечётные 8-ные цифры заменяем на 1
    w = w.replace('3', '1').replace('5', '1').replace('7', '1')
    return '11' not in w and '22' not in w

# тестирование функции ok(w)
#print(ok('16341')) # True
#print(ok('1631'))  # False
#print(ok('2221'))  # False

abc = '01234567'  # цифры 8-ной с.с.
t = [''.join(w) for w in permutations(abc, r=6)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if '1' not in w and ok(w)]

print(len(answer))
```

Решение (вариант 2). Ответ: 252

```
from itertools import permutations

abc = '01234567'  # цифры 8-ной с.с.
t = [''.join(w) for w in permutations(abc, r=6)]
t = [w for w in t if w[0] != '0']

answer = [w for w in t if '1' not in w \
          and all(int(w[i], 8) % 2 != int(w[i+1], 8) % 2 \
                  for i in range(len(w)-1))]

print(len(answer))
```

Пример 8.25. Сколько существует шестнадцатеричных пятизначных чисел, в записи которых все цифры различны и чётность соседних цифр не совпадает?

Решение очень похоже на решение, приведённое в примере 8.22 (существенное отличие — разные методы, здесь нужен `permutations()`, так как все цифры шестнадцатеричного пятизначного числа различны).

Решение (вариант 1). Ответ: 35280

```
from itertools import permutations

def ok(w):
    # в цикле ищем ”нарушителей”: соседние цифры имеют одинаковую чётность
    for i in range(0, len(w)-1):
        if int(w[i], 16) % 2 == int(w[i+1], 16) % 2:
            return False
    return True

# тестирование функции ok(w)
#print(ok(’16341’)) # True
#print(ok(’1631’))  # False
#print(ok(’2221’))  # False

abc = ’0123456789abcdef’  # цифры 16-ной с.с.
t = [’’.join(w) for w in permutations(abc, r=5)]
t = [w for w in t if w[0] != ’0’]

answer = [w for w in t if ok(w)]

print(len(answer))
```

Решение (вариант 2). Ответ: 35280

```
from itertools import permutations

abc = ’0123456789abcdef’  # цифры 16-ной с.с.
t = [’’.join(w) for w in permutations(abc, r=5)]
t = [w for w in t if w[0] != ’0’]

answer = [w for w in t if all(int(w[i], 16) % 2 != int(w[i+1], 16) % 2 \
                              for i in range(0, len(w)-1))]

print(len(answer))
```

Пример 8.26. Сколько существует различных семеричных чисел длиной не больше шести, в записи которых все цифры чётные?

Решение. Длина чисел может быть 1, 2, 3, 4, 5, 6.

Решение. Ответ: 4096

```
from itertools import product

abc = '0246'  # "чётные" цифры 7-ной с.с.
cnt = 0
for k in range(1, 7):
    t = [''.join(w) for w in product(abc, repeat=k)]
    t = [w for w in t if w[0] != '0']
    cnt += len(t)

print(cnt + 1)  # Пояснение: + 1, так как отдельно добавляется 0
```

Задания для самостоятельной работы

№ 8.44 (➔). Сколько существует шестеричных пятизначных чисел, которые не могут начинаться с цифр 3 и 5 и не содержат пар 11 и 44 одновременно?

№ 8.45 (➔). Сколько существует восьмеричных шестизначных чисел, которые не могут начинаться с цифр 4 и 6 и не должны содержать пары соседних одинаковых цифр.

№ 8.46 (➔). Сколько существует девятеричных пятизначных чисел, которые не начинаются с чётной цифры, не оканчиваются цифрой 1 или 7, а также содержат в своей записи не более двух цифр 3?

№ 8.47 (типа ЕГЭ-2024; [➔]). Сколько существует двенадцатеричных семизначных чисел, в записи которых ровно одна цифра 5 и не более трёх цифр с числовым значением, превышающим 8.

№ 8.48 (➔). Сколько существует девятеричных пятизначных чисел, в записи которых ровно одна цифра 7 и никакая нечётная цифра не стоит рядом с цифрой 7?

№ 8.49 (➔). Сколько существует десятичных шестизначных чисел, в записи которых никакие две чётные и нечётные цифры не стоят рядом?

№ 8.50 (➔). Сколько существует шестнадцатеричных пятизначных чисел, в которых все цифры идут в порядке возрастания, при этом чётные и нечётные цифры чередуются?

№ 8.51 (➔). Сколько существует десятичных чисел, кратных пяти, в записи которых все цифры различны.

Задания для дополнительной отработки

№ 8.52 (➔). Сколько существует десятичных шестизначных чисел, кратных пяти?

№ 8.53 (➔). Сколько существует восьмеричных пятизначных чисел, которые не могут начинаться с цифр 2 и 4 и не содержат пар 33 и 77 одновременно?

№ 8.54 (➔). Сколько существует семеричных шестизначных чисел, которые не могут начинаться с цифр 1 и 3, и не должны содержать пары соседних одинаковых цифр.

№ 8.55 (➔). Сколько существует восьмеричных семизначных чисел, которые начинаются с чётной цифры и не оканчиваются парой цифр 17?

№ 8.56 (➔). Сколько существует шестнадцатеричных пятизначных чисел, в записи которых ровно одна цифра 9 и никакая нечётная цифра не стоит рядом с цифрой 9?

№ 8.57 (➔). Сколько существует восьмеричных семизначных чисел, в записи которых никакие две чётные и нечётные цифры не стоят рядом?

№ 8.58 (➔). Сколько существует шестнадцатеричных пятизначных чисел, в которых все цифры идут в порядке убывания, при этом чётные и нечётные цифры чередуются?

№ 8.59 (➔). Сколько существует восьмеричных пятизначных чисел, не содержащих в своей записи последовательность 10, в которых все цифры различны и никакие две чётные или две нечётные цифры не стоят рядом?

№ 8.60 (➔). Сколько существует шестнадцатеричных пятизначных чисел, в записи которых все цифры различны и расположены по возрастанию?

№ 8.61 (➔). Сколько существует различных семизначных чисел, получающихся перестановкой цифр числа 1051515.

№ 8.62 (➔). Сколько существует различных восьмеричных чисел, оканчивающихся на чётную цифру, в записи которых все цифры различны.

8.7 Новые задачи. Часть D

Задания для самостоятельной работы

№ 8.63 (➔). Все пятибуквенные слова, составленные из букв Б, Р, Е, Л, О, К, записаны в определённом порядке и пронумерованы начиная с 1. Вот начало списка:

1. ББББББ
2. ББББР
3. ББББЕ
4. ББББЛ
5. ББББО
6. ББББК
7. БББРБ

… …

Под каким номером в списке стоит слово «БЕЛОК»?

№ 8.64 (➔). Все пятибуквенные слова, составленные из букв У, Р, О, К, В, Ы записаны в определённом порядке и пронумерованы начиная с 1. Вот начало списка:

1. УУУУУ
2. УУУУР
3. УУУУО
4. УУУУК
5. УУУУВ
6. УУУУЫ
7. УУУРУ

… …

Под каким номером в списке стоит слово «РЫВОК»?

№ 8.65 (➔). Все шестибуквенные слова, составленные из букв Р, О, Л, И, К записаны в определённом порядке и пронумерованы начиная с 1. Вот начало списка:

1. РРРРРР
2. РРРРРО
3. РРРРРЛ
4. РРРРРИ
5. РРРРРК
6. РРРРОР

… …

Под каким номером в списке стоит слово «КРОЛИК»?

№ 8.66 (➔). Все пятибуквенные слова, составленные из букв П, Е, Т, Р, У, Ш, К, А записаны в определённом порядке и пронумерованы начиная с 1. Вот начало списка:

1. ППППП
2. ПППЕ
3. ПППТ
4. ПППР
5. ПППУ
6. ППППШ
7. ПППК
8. ПППА
9. ПППЕП

… …

Сколько слов стоит между словами «РЕПКА» и «ШТУКА»?

Ответы

Раздел 8.2: «Практикум по программированию. Часть А»

№ 8.1: ЯМКА

№ 8.2: МАММА

№ 8.3: НССА

№ 8.4: МЫШЦА

№ 8.5: ОСОАА

№ 8.6: МЫШЬЁЖ

№ 8.7: БУНТ

№ 8.8: ЁЖГДЕ

№ 8.9: 516

№ 8.10: 8857

№ 8.11: 2457 (слово «ЛОЛЕЕ»)

№ 8.12: 2624

№ 8.13: 3358661 (слово «ЫЬОИЕА»)

№ 8.14: 5456

№ 8.15: 346

№ 8.16: 5415

№ 8.17: 1387

№ 8.18: 308 (слово «ЛЕТЕТ»)

№ 8.19: 12251 (слово «КРКААА»)

№ 8.20: 1875

№ 8.21: 2419 (слово «ВЛППРТ»)

№ 8.22: 3560

Раздел 8.4: «Практикум по программированию. Часть B»

№ 8.23: 2160

№ 8.24: 32

№ 8.25: 2133

№ 8.26: 1296

№ 8.27: 5400

№ 8.28: 1458

№ 8.29: 3110

№ 8.30: 297

№ 8.31: 625

№ 8.32: 625

№ 8.33: 486

№ 8.34: 462

№ 8.35: 36

№ 8.36: 729

№ 8.37: 516 Подсказка: см. пример 8.13

№ 8.38: 7560

№ 8.39: 1520

№ 8.40: 504

№ 8.41: 90

№ 8.42: 480

№ 8.43: 168

Раздел 8.6: «Практикум по программированию. Часть C»

№ 8.44: 3862

№ 8.45: 84035

№ 8.46: 19952

№ 8.47: 10824448

№ 8.48: 8880

№ 8.49: 28125

№ 8.50: 378

№ 8.51: 1863219

№ 8.52: 180000

№ 8.53: 20442

№ 8.54: 31104

№ 8.55: 774144

№ 8.56: 91680

№ 8.57: 28672

№ 8.58: 504

№ 8.59: 432

№ 8.60: 3003

№ 8.61: 120

№ 8.62: 48929