

Решение задания №8 ЕГЭ по информатике с использованием языка программирования Python.
КОМБИНАТОРИКА.

(базовый уровень, время – 4 мин)

Тема: Кодирование данных, комбинаторика, системы счисления.

Что проверяется:

Знание о методах измерения количества информации (?)

1.6.1. Формализация понятия алгоритма (?)

1.1.4. Читать и отлаживать программы на языке программирования (?)

Что нужно знать:

В русском языке 33 буквы: 10 гласных букв (а, у, о, ы, и, э, я, ю, ё, е), 21 согласная буква (б, в, г, д, ж, з, й, к, л, м, н, п, р, с, т, ф, х, ц, ч, ш, щ) и два знака (ь, ъ).

Алфавит английского языка по написанию совпадает с латинским алфавитом и состоит из 26 букв.

принципы работы с числами, записанными в позиционных системах счисления

если слово состоит из L букв, причем есть n_1 вариантов выбора первой буквы, n_2 вариантов выбора второй буквы и т.д., то число возможных слов вычисляется как произведение

$$N = n_1 \cdot n_2 \cdot \dots \cdot n_L$$

если слово состоит из L букв, причем каждая буква может быть выбрана n способами, то число возможных слов вычисляется как $N = n^L$

если в программе L вложенных циклов и внешний цикл выполняется n_1 раз, следующий (вложенный) n_2 раз и т.д., то команды самого внутреннего цикла будут выполняться N раз, где

$$N = n_1 \cdot n_2 \cdot \dots \cdot n_L$$

Если $n_1 = n_2 = \dots = n_L = n$, то $N = n^L$.

при увеличении n или L значение N сильно возрастает, что приводит к существенному увеличению времени выполнения программы.

Выведем общую формулу для количества вариантов, когда символы могут повторяться!

$$N = m^i$$

Данную задачу нужно решать используя формулу **сочетаний** из раздела **комбинаторика**.

$$C_n^m = \frac{n!}{m!(n-m)!} \quad (m < n)$$

n - количество книг, из которых мы выбираем подарок, m - количество книг, которое мы хотим выбрать, C - количество вариантов (способов).

Решение с помощью вложенных циклов

Задача (Классическая)

Женя составляет 5-буквенные слова, в которых встречаются только буквы А, Б, В, Г, причём буква А появляется ровно 1 раз. Каждая из других допустимых букв может встречаться в слове любое количество раз или не встречаться совсем. Словом считается любая допустимая последовательность букв, не обязательно осмысленная. Сколько существует таких слов, которые может написать Женя?

Решение:

Напишем программу на языке **Python**.

```
k=0
for x1 in 'АБВГ':
    for x2 in 'АБВГ':
        for x3 in 'АБВГ':
            for x4 in 'АБВГ':
                for x5 in 'АБВГ':
                    s=x1+x2+x3+x4+x5
                    if s.count('А')==1:
                        k=k+1
print(k)
```

Т.к. слова состоят из **5-ти** символов, то мы формируем **пять** вложенных циклов! В каждом цикле перебираем все буквы, которые нам дали.

Внутри циклов мы составляем само слово в переменной **s**. Таким образом, в переменной **s** "прокрутятся" все возможные комбинации.

Но мы подсчитываем не все комбинации, а только те, где всего одна буква А.

Важно не перепутать русские и английские буквы.

Ответ: 405

Продолжим развивать навыки решения 8 задания из **ЕГЭ по информатике 2022**.

Задача (Развиваем навыки)

Артур составляет 5-буквенные коды из букв Е, С, А, У, Л. Каждую букву нужно использовать ровно один раз, при этом нельзя ставить рядом две гласные. Сколько различных кодов может составить Артур?

Источник задачи: <https://kpolyakov.spb.ru/>

Решение:

Запрограммируем решение этой задачи на Питоне.

```
k=0
for x1 in 'ЕСАУЛ':
    for x2 in 'ЕСАУЛ':
        for x3 in 'ЕСАУЛ':
            for x4 in 'ЕСАУЛ':
                for x5 in 'ЕСАУЛ':
                    s=x1+x2+x3+x4+x5
                    if s.count('Е')==1 and s.count('С')==1 and s.count('А')==1 and s.count('У')==1 and
s.count('Л')==1:
```

```

        if s.count('EA')==0 and s.count('AE')==0 and s.count('EY')==0 and
s.count('YE')==0 and s.count('AY')==0 and s.count('YA')==0:
            k=k+1
print(k)

```

В первом условии учли, что каждая буква встречается в слове только один раз. Второе условие говорит о том, что две гласные не стоят рядом (перебрали все возможные сочетания гласных).

Ответ: 12

В задании 8 из **ЕГЭ по информатике** часто нужно проанализировать первую или последнюю букву в слове. Узнаем, как это можно сделать с помощью питона.

Задача (Проверяем первую букву слова)

Сколько слов длины 5, начинающихся с гласной буквы, можно составить из букв Е, Г, Э? Каждая буква может входить в слово несколько раз. Слова не обязательно должны быть осмысленными словами русского языка.

Источник задачи: <https://kpolyakov.spb.ru/>

Решение:

В этой тренировочной задаче из 8 задания **ЕГЭ по информатике 2022** нужно держать на контроле первую букву в слове.

```

k=0
for x1 in 'ЕГЭ':
    for x2 in 'ЕГЭ':
        for x3 in 'ЕГЭ':
            for x4 in 'ЕГЭ':
                for x5 in 'ЕГЭ':
                    s=x1+x2+x3+x4+x5
                    if x1=='Е' or x1=='Э':
                        k=k+1
print(k)

```

Подсчитываем только те комбинации, которые начинаются с гласных букв.

Ответ: 162

Интересный пример, где можно ошибиться в 8 задании из **ЕГЭ по информатике**.

Задача (Важный пример)

Сергей составляет 6-буквенные коды из букв С, О, Л, О, В, Е, Й. Буква Й может использоваться в коде не более одного раза, при этом она не может стоять на первом месте, на последнем месте и рядом с буквой Е. Все остальные буквы могут встречаться произвольное количество раз или не встречаться совсем. Сколько различных кодов может составить Сергей?

Источник задачи: <https://kpolyakov.spb.ru/>

Решение:

Эта задача примечательная тем, что буква "О" в слове "СОЛОВЕЙ" повторяется. В этом случае мы должны убрать повторение буквы из перебора.

```

k=0
for x1 in 'СОЛВЕЙ':
    for x2 in 'СОЛВЕЙ':
        for x3 in 'СОЛВЕЙ':
            for x4 in 'СОЛВЕЙ':

```

```

        for x5 in 'СОЛВЕЙ':
            for x6 in 'СОЛВЕЙ':
                s=x1+x2+x3+x4+x5+x6
                if s.count('Й')<=1 and x1!='Й' and x6!='Й' and s.count('ЕЙ')==0 and
s.count('ЙЕ')==0:
                    k=k+1
print(k)

```

Здесь также учитываем остальные условия.

Ответ: 23625

Задача (Количество гласных)

Василий составляет 4-буквенные коды из букв Г, А, Ф, Н, И, Й. Каждую букву можно использовать любое количество раз, при этом код не может начинаться с буквы Й и должен содержать хотя бы одну гласную. Сколько различных кодов может составить Василий?

Источник задачи: <https://kpolyakov.spb.ru/>

Решение:

Напишем программу.

```

k=0
for x1 in 'ГАФНИЙ':
    for x2 in 'ГАФНИЙ':
        for x3 in 'ГАФНИЙ':
            for x4 in 'ГАФНИЙ':
                s=x1+x2+x3+x4
                if x1!='Й' and s.count('А') + s.count('И') >= 1:
                    k=k+1

print(k)

```

Ответ: 888

Порешаем задачи из восьмого задания **ЕГЭ по информатике** на перебор чисел.

Задача (Перебор чисел)

Сколько существует чисел, восьмеричная запись которых содержит 7 цифр, причём все цифры различны и никакие две чётные и две нечётные цифры не стоят рядом.

Источник задачи: <https://kpolyakov.spb.ru/>

Решение:

```

k1=0
k2=0
for x1 in '1234567':
    for x2 in '01234567':
        for x3 in '01234567':
            for x4 in '01234567':
                for x5 in '01234567':
                    for x6 in '01234567':
                        for x7 in '01234567':
                            s=x1+x2+x3+x4+x5+x6+x7
                            if s.count(x1)==1 and s.count(x2)==1 and s.count(x3)==1 and s.count(x4)==1
and s.count(x5)==1 and s.count(x6)==1 and s.count(x7)==1:
                                if int(x1)%2==0 and int(x2)%2==1 and int(x3)%2==0 and int(x4)%2==1 and
int(x5)%2==0 and int(x6)%2==1 and int(x7)%2==0:
                                    k1=k1+1

```

```

        if int(x1)%2==1 and int(x2)%2==0 and int(x3)%2==1 and int(x4)%2==0 and
int(x5)%2==1 and int(x6)%2==0 and int(x7)%2==1:
            k2=k2+1

print(k1+k2)

```

Число не может начинаться с нуля. Поэтому ноль был исключён из первого цикла.

Первое условие следит за тем, чтобы каждая цифра встречалась один раз в числе. Второе условие подсчитывает количество вариантов, когда первая цифра чётная. Второе условие следит за тем, чтобы чётность и нечётность цифр чередовалась. Третье условие, наоборот, подсчитывает варианты, когда первая цифра нечётная.

Операция **%** - остаток от деления. Если остаток от деления на 2 равен нулю, то число чётное. Если остаток от деления на 2 равен 1, то число нечётное.

Функция **int()** преобразует символ в число. Ведь мы работаем именно с символами, а не с реальными числами.

Ответ: 1008

Задача (Числа, Закрепление)

Сколько существует четырёхзначных чисел, записанных в восьмеричной системе счисления, в записи которых ровно две одинаковые цифры, причём стоящие рядом ?

Решение:

```

k=0
for x1 in '1234567':
    for x2 in '01234567':
        for x3 in '01234567':
            for x4 in '01234567':
                if (x1==x2 and x2!=x3 and x2!=x4 and x3!=x4) or (x2==x3 and x3!=x1 and x3!=x4 and x1!=x4)
or (x3==x4 and x3!=x2 and x3!=x1 and x1!=x2):
                    k=k+1
print(k)

```

Здесь следующий принцип составления условия. Два соседа должны быть равны. Берём одного соседа из пары, где цифры должны быть равны, и комбинируем его с другими цифрами. Пишем уже, чтобы цифры были не равны. Так же прописываем, чтобы две оставшиеся цифры также не были равны. Таким образом, перебираем все варианты.

Ответ: 882

Задача (Числа, важный приём)

Сколько существует различных трёхзначных чисел в шестнадцатеричной системе счисления, в записи которых цифры следуют слева направо в невозрастающем порядке?

Решение:

```

k=0
for x1 in '123456789ABCDEF':
    for x2 in '0123456789ABCDEF':
        for x3 in '0123456789ABCDEF':

```

```

if ord(x1)>=ord(x2)>=ord(x3):
    k=k+1

print(k)

```

Функция **ord()** - для символа вернет число из таблицы символов Unicode. Посмотрим на фрагмент таблицы кодировки ANSI (таблица ANSI - это часть таблицы Unicode) для цифр и нужных букв.

40	(00101000	64	@	01000000
41)	00101001	65	A	01000001
42	*	00101010	66	B	01000010
43	+	00101011	67	C	01000011
44	,	00101100	68	D	01000100
45	-	00101101	69	E	01000101
46	.	00101110	70	F	01000110
47	/	00101111	71	G	01000111
48	0	00110000	72	H	01001000
49	1	00110001	73	I	01001001
50	2	00110010	74	J	01001010
51	3	00110011	75	K	01001011
52	4	00110100	76	L	01001100
53	5	00110101	77	M	01001101
54	6	00110110	78	N	01001110
55	7	00110111	79	O	01001111

Левый столбик - это как раз то число, которое вернёт функция ord() для символа из среднего столбца. Видим, что цифры начинаются с 48. Буквы идут в алфавитном порядке, и для каждой буквы повышается код на 1.

Таким образом, в этой задачке функция ord() отлично подходит!

Ответ: 815

Следующий тип задач из задания 8 ЕГЭ по информатике лучше решать без программирования.

Задача (Со списками, классическая)

Все 4-буквенные слова, составленные из букв А, Е, И, О записаны в алфавитном порядке и пронумерованы. Вот начало списка:

1. АААА
2. АААЕ
3. АААИ
4. АААО
5. ААЕА
- ...

Запишите слово, стоящее на 248-м месте от начала списка.

Источник задачи: <https://inf-ege.sdamgia.ru/>

Решение:

Обозначим условно **А** - 0, **Е** - 1, **И** - 2, **О** - 3.

Важно: Нужно буквам присваивать цифры именно в том порядке, в котором они идут в самом правом столбце, потому что буквы могут дать в "перепутанном порядке" (например Е, А, И, О), и тогда ничего не получится.

1.	А	А	А	А	0
2.	А	А	А	Е	1
3.	А	А	А	И	2
4.	А	А	А	О	3
5.	А	А	Е	А	0
...					

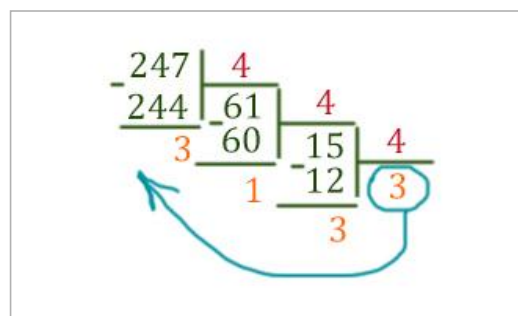
Теперь запишем список с помощью цифр.

1. 0000
2. 0001
3. 0002
4. 0003
5. 0010
- ...

Получился обычный счёт в **четверичной системе!!** (всего используются 4 цифры: 0, 1, 2, 3). А слева нумерация показывает соответствие нашей десятичной системе. Но **все числа десятичной системы** в этой таблице соответствия **сдвинуты на 1**, ведь мы должны были начать с нуля.

Нас просят записать слово стоящее на 248, т.е. если была обычная таблица соответствия чисел десятичной системы и четверичной системы, слово стоящее на 248 месте, находилось бы на 247 (248 - 1) месте. Значит, наше искомое **четверичное число** соответствует 247 в десятичной системе.

Переведём число 247 в четверичную систему!



Получилось число **33134** в четверичной системе. Сделаем обратное декодирование в буквы. Таким образом, ответ будет **ООЕО**.

Ответ: ООЕО

Ещё одна похожая задача 8 задания из примерных вариантов **ЕГЭ по информатике 2022**, но другой вариации.

Задача (Классика, Другая вариация)

Все 5-буквенные слова, составленные из букв А, Р, У, К записаны в алфавитном порядке. Вот начало списка:

1. ААААА
2. ААААК
3. ААААР
4. ААААУ
5. АААКА

.....

Укажите номер слова УКАРА

Источник задачи: <https://inf-ege.sdamgia.ru/>

Решение:

Закодируем буквы цифрами: **А** - 0, **К** - 1, **Р** - 2, **У** - 3. Здесь как раз буквы даны не в том порядке, как они идут в самом правом столбце. Но мы должны кодировать именно в том порядке, как буквы идут в самом правом столбце.

1. ААААА	0
2. ААААК	1
3. ААААР	2
4. ААААУ	3
5. АААКА	

У нас получилось четыре цифры! Значит снова можно слова превратить в таблицу соответствия между **десятичной** системой и **четверичной** системой. Но десятичная система смещена на 1 позицию.

1. 00000
2. 00001
3. 00002
4. 00003
5. 00010

.....

Выписываем данное нам слово и посмотрим, какое число в четверичной системе было бы, если бы у нас были в место слов числа в четверичной системе!

У	К	А	Р	А
3	1	0	2	0

Получили число в четверичной системе **310204**. Узнаем, какое число в десятичной системе соответствовало этому числу, если бы была обычная таблица соответствия. Для этого переведем число **310204** из **четверичной** системы в **десятичную**. Перевод делаем по аналогии перевода из двоичной системы в десятичную.

$$0 * 4^0 + 2 * 4^1 + 0 * 4^2 + 1 * 4^3 + 3 * 4^4 = 840 \text{ (в десят. системе)}$$

8
64
768

Но помним, что у нас нумерация идёт на 1 быстрее, нежели мы бы поставили десятичные числа, как в таблице соответствия, потому что нумерация начинается не с нуля, а с 1. Поэтому к числу **840** **нужно прибавить 1**, и в ответе будет **841**

Ответ: 841

Задача (Демонстрационный вариант ЕГЭ по информатике, 2020)

Все 4-буквенные слова, в составе которых могут быть буквы Н, О, Т, К, И, записаны в алфавитном порядке и пронумерованы, начиная с 1. Ниже приведено начало списка.

1. ИИИИ
2. ИИИК
3. ИИИН
4. ИИИО
5. ИИИТ
6. ИИКИ
- ...

Под каким номером в списке идёт первое слово, которое начинается с буквы О?

Решение:

Закодируем буквы цифрами.

1. ИИИИ	0
2. ИИИК	1
3. ИИИН	2
4. ИИИО	3
5. ИИИТ	4
6. ИИКИ	

Получилось 5 цифр (0, 1, 2, 3, 4), значит, будем работать в **пятеричной системе**.

Нужно найти номер первого слова, которое начинается с буквы **О**. Если говорить на языке пятеричных чисел, то нужно найти номер числа **30005**. Мы "забываем нулями", чтобы число было четырёхразрядное, т.к. слова 4-х буквенные. Именно нулями, потому что нужно именно первое слово найти.

Теперь, как в предыдущей задаче, переведём число **30005** из **пятеричной** системы в **десятичную**.

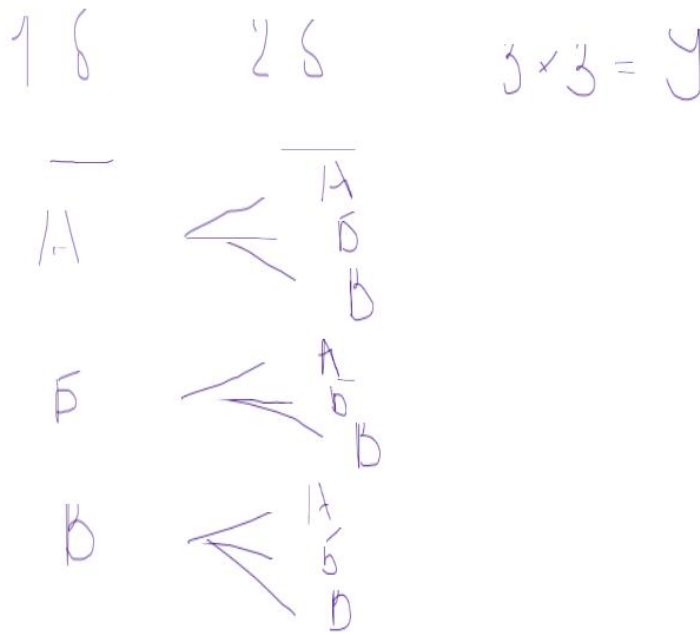
$$0 * 5^0 + 0 * 5^1 + 0 * 5^2 + 3 * 5^3 = 375 \text{ (в десят. системе)}$$

Но опять же должны прибавить 1 к числу **375**, т.к. нумерация отличается от десятичных чисел на 1 в большую сторону.

Ответ: 376

Решение первого типа с помощью множеств (Расширенный вариант)

Пример № 1. Сколько слов длины 2 буквы, можно составить из букв А,Б,В. Каждая буква может входить в слово несколько раз или не входить совсем. Слова не обязательно должны быть осмысленными словами русского языка.



Ответ: 9

При решении задач по комбинаторике с помощью программы на языке Python удобно использовать функции из модуля `itertools`:

`product` – декартово произведение (все возможные слова заданной длины, составленные из данного алфавита), например:

```
from itertools import product
k=0
# k – количество слов
for x in product('АБВ', repeat = 2):
    k+=1
    print(x)
print (k)
```

```
( 'A' , 'A' )
( 'A' , 'Б' )
( 'A' , 'В' )
( 'Б' , 'А' )
( 'Б' , 'Б' )
( 'Б' , 'В' )
( 'В' , 'А' )
( 'В' , 'Б' )
( 'В' , 'В' )
9
```

Как видно из этого примера, результат работы функции – массив отдельных букв.

В нём удобно работать с отдельными символами, но неудобно искать сочетания букв. Если нужно работать с сочетаниями букв, нужно «склеить» символы каждого кортежа в строки с помощью метода .join:

```
from itertools import product
k=0
#k-количество слов
for x in product('АБВ',repeat = 2):
    k+=1
    s="".join(x)
    print(s)
    print (k)
```

```
АА
АБ
АВ
БА
ББ
БВ
ВА
ВБ
ВВ
9
```

Пример № 2

Сколько слов длины 6 букв, начинающихся и заканчивающихся согласной буквой, можно составить из букв Г,О,Д. Каждая буква может входить в слово несколько раз или не входить совсем. Слова не обязательно должны быть осмысленными словами русского языка.

Решение 1 способ.

ГОДГОД

```
from itertools import product
k=0
#k-количество слов
for x in product('ГОД',repeat = 6):
    s="".join(x)
    if s[0] != 'О' and s[5] != 'О':
        k+=1
    #print(s)-вывод слов
print (k)
```

Ответ: 324

Решение 2 способ.

$$\begin{array}{r} 2 \\ \hline 1,2 \end{array} \quad \begin{array}{r} 3 \\ \hline 1,2,0 \end{array} \quad \begin{array}{r} 3 \\ \hline \end{array} \quad \begin{array}{r} 3 \\ \hline \end{array} \quad \begin{array}{r} 3 \\ \hline \end{array} \quad \begin{array}{r} 2 \\ \hline 1,2 \end{array}$$
$$2 \times 3^4 \times 2 = 324$$

Пример №1 для самостоятельного выполнения. 10-минут. Код программы и ответ скопировать в чат.

Например, добавить в текст программы, # Ответ ...

Сколько слов длины 4, начинающихся с согласной буквы, можно составить из букв Л, Е, Т, О?

Каждая буква может входить в слово несколько раз. Слова не обязательно должны быть осмысленными словами русского языка.

for $4 \times 4 \times 4 \times 4 \times 4 = 1024$

$$0.33333 = 243$$

$$5_A A A A A = 1$$

4A

A	A	A	A
—	—	—	<u>3</u>
—	—	3	—
<u>3</u>	3	—	—
—	—	—	—

= 3

→ 3

= 3

= 3

ОТВЕТ: $1024-243-3*5-1=765$

C/p №2

Какое количество 5-буквенных слов Вы можете составить из набора букв слова «ВЕКО»? В данной задаче нужно принять подходящими все возможные последовательности, вне зависимости имеет или нет данный набор букв смысловое содержание. Также в данной задаче есть ограничение: буква Е встречается в слове хотя бы один раз, каждая другая буква может встречаться любое количество раз, а может не встречаться совсем.

bce: $\underline{4} \cdot \underline{4} \cdot \underline{4} \cdot \underline{4} \cdot \underline{4} = 1024$ ~~✓~~

$$0! \cdot \underline{3} \cdot \underline{3} \cdot \underline{3} \cdot \underline{3} \cdot \underline{3} = 243 \text{ E:}$$

$$\begin{array}{r} 6ce - 0 = 1024 - 243 \\ 0, 1, 2, 3, 4, 5 \quad 0 \quad = 781 \end{array}$$

O best:

Пример № 4 (ДЕМО-2023, задание в варианте ЕГЭ 2022)

Определите количество пятизначных чисел, записанных в восьмеричной системе счисления, в записи которых только одна цифра 6, при этом никакая нечетная цифра не стоит рядом.

Решение.

$\{0,1,2,3,4,5,6,7\}$ – цифры в восьмеричной системе счисления

{0,2,4,6} – четные цифры.

Но по условию цифра 6 встречается только один раз в числе, поэтому будем рассматривать цифры {0,2,4}

Внимание!!! 0 не может стоять на первом месте в числе

1 способ решения.

1	3	7	7	7
6	(0, 2, 4)	(0, 1, 2, 3, 4, 5, 7)		
2	1	3	7	7
(2, 4)	6			
6	3	1	3	7
(1, 2, 3, 4, 5, 7)		6		
6	7	3	1	3
			6	
6	7	7	3	1
				6

Выполняем вычисления в Python

Python 3.6.2 (v3.6.2:5fd33b5, Jul 8 2017, 04:14:34) [MSC v.1900 32 bit (Intel)] on win32

Type "copyright", "credits" or "license()" for more information.

```
>>> 3*7*7*7+2*3*7*7+6*3*3*7+6*7*3*3+6*7*7*3
```

```
2961
```

```
>>>
```

Ответ:2961

```
from itertools import product
```

```
k=0
```

```
for x in product('01234567',repeat = 5):
```

```
    s="".join(x)
```

```
    if s.count('6') == 1 and s[0] != '0':
```

```
        if '61' not in s and '63' not in s and '65' not in s and '67' not in s:
```

```
            if '16' not in s and '36' not in s and '56' not in s and '76' not in s:
```

```
                k+=1
```

```
print(k)
```

С/р. № 3–

Определите количество пятизначных чисел, записанных в девятеричной системе счисления, в записи которых ровно одна цифра 1, при этом никакая чётная цифра не стоит рядом с цифрой 1.

Определите количество четырехзначных чисел, записанных в шестеричной системе счисления, в записи которых ровно одна цифра 3, при этом никакая нечетная цифра не может быть последней цифрой числа.

Дополнительно:

Методы Python:

Использование методов `extend()` и `append()` со строками

Строки, по сути своей, — последовательности символов, а значит, они относятся к итерируемым объектам. Давайте попробуем применить методы `extend()` и `append()` к `list_1`, а в качестве аргумента передать строку.

Добавим к списку `list_1` строку 'Happy' при помощи метода `append()`.

```
list_1 = [1, 2, 3, 4, 5]
list_1.append('Happy')
print(list_1)
# Output
[1, 2, 3, 4, 5, 'Happy']
```

А теперь давайте попробуем сделать то же самое при помощи метода `extend()`. Сейчас у нас не должно возникнуть ошибок. По идее, метод `extend()` переберет строку и добавит каждый символ к `list_1`. Давайте посмотрим, сработает ли.

```
list_1 = [1, 2, 3, 4, 5]
list_1.extend('Happy')
print(list_1)
# Output
[1, 2, 3, 4, 5, 'H', 'a', 'p', 'p', 'y']
```

Что касается **временной сложности** этих методов, мы интуитивно догадываемся, что `append()` имеет постоянное время выполнения, а время выполнения `extend()` пропорционально зависит от длины итерируемого объекта, который передается в качестве аргумента.

Добавление единичного элемента:

```
list_1 = [1, 2, 3, 4, 5]
list_1.append(True)
print(list_1)
# Output
[1, 2, 3, 4, 5, True]
```

`product(*iterables, repeat=1)`

Пакет `itertools` содержит небольшую, но очень полезную функцию, которая создает продукты **Cartesian** из ряда вложенных итерируемых. Да, эта функция является продуктом. Давайте посмотрим на то, как это работает!

Python

```
1 from itertools import product
2
3 arrays = [(-1,1), (-3,3), (-5,5)]
4 cp = list(product(*arrays))
5
6 print(cp)
```

Результат:

Python

```
1 [(-1, -3, -5),
2  (-1, -3, 5),
3  (-1, 3, -5),
4  (-1, 3, 5),
5  (1, -3, -5),
6  (1, -3, 5),
7  (1, 3, -5),
8  (1, 3, 5)]
```

Здесь мы импортируем **product**, затем устанавливаем список кортежей, который мы назначаем переменным массивам. Далее, мы вызываем **product** с этими массивами. Вы заметите, что мы вызываем их при помощи ***arrays**. Благодаря этому список будет «взорван» или применен к функции продукта в определенной последовательности. Это значит, что вы передаете 3 аргумента, вместо одного. Если хотите, попробуйте вызвать её, используя предварительно скопированную в массивы звездочку.

`permutations`

Наследуемый модуль `itertools` под названием **permutations** возвращает последовательные **перестановки элементов** из итерируемого вами значения той или иной длины. Как и функция **combinations**, `permutations` создает выдачу в лексикографическом порядке сортировки. Давайте посмотрим на пример:

Python

```
1 from itertools import permutations
2
3 for item in permutations('WXYZ', 2):
4     print("".join(item))
```

Результат:

Python

```
1 WX
2 WY
3 WZ
4 XW
5 XY
6 XZ
```

7YW
8YX
9YZ
10ZW
11ZX
12ZY

Обратите внимание на то, что выдача несколько длиннее, чем у функции **combinations**. Когда вы используете **permutations**, он перепробует все варианты перестановок в строке, но не будет повторять значения, если вложенные элементы являются уникальными