

Пётр Алексеевич Леляев

*Кандидат физико-математических наук,
ведущий специалист ГАОУ ДПО «ТемоЦентр».*

**Владислав Сергеевич Попов**

Учитель информатики ГБОУ Цифровая школа, старший преподаватель кафедры «Информационные системы и телекоммуникации» МГТУ им. Н. Э. Баумана.

Способы решения задачи № 8 ЕГЭ по информатике

Согласно спецификации ЕГЭ по информатике 2021 года, задача № 8 проверяет такой элемент содержания, как знание о методах измерения количества информации. Обычно такая задача является комбинаторной и заключается в определении количества кодовых слов, удовлетворяющих определённым требованиям. Например, рассмотрим задачу из демоверсии ЕГЭ 2021 года:

Игорь составляет таблицу кодовых слов для передачи сообщений, каждому сообщению соответствует своё кодовое слово. В качестве кодовых слов Игорь использует трёхбуквенные слова, в которых могут быть только буквы Ш, К, О, Л, А, причём буква К появляется ровно 1 раз. Каждая из других допустимых букв может встречаться в кодовом слове любое количество раз или не встречаться совсем. Сколько различных кодовых слов может использовать Игорь?

Такую задачу несложно решить письменно. Заметим, что если буква К встречается в слове на первом месте, таких слов получается 16 — на любом из двух оставшихся мест могут находиться любые из 4 оставшихся букв. Аналогичная ситуация с буквой К на второй и третьей позиции в слове, таким образом, ответ равен 48. Проблемы в решении таких задач могут возникать при увеличении длины слова, мощности алфавита и усложнении условий отбора подходящих слов. Воспользуемся преимуществами языка программирования Python для решения подобных задач.

Принцип решения будет прост: будем перебирать все возможные слова нужной длины, составленные из указанных букв, и при помощи проверки условия выбирать из них подходящие. Для генерации списка всех возможных слов длины 3, образованных буквами слова ШКОЛА, воспользуемся функцией `product`

из модуля `itertools` (модуль не требует дополнительной установки и доступен везде, где есть Python):

```
>>> from itertools import product
>>> words = list(product('wkola', repeat=3))
```

Проверим, как выглядит начало списка:

```
>>> words[:10]
[('w', 'w', 'w'), ('w', 'w', 'k'), ('w', 'w', 'o'), ('w', 'w', 'l'),
 ('w', 'w', 'a'), ('w', 'k', 'w'), ('w', 'k', 'k'), ('w', 'k', 'o'),
 ('w', 'k', 'l'), ('w', 'k', 'a')]
```

Видно, что в `words` хранятся все кортежи длины 3, состоящие из необходимых букв. Осталось проверить, сколько из них содержат одну букву `К`. Создадим переменную `cnt`, где будем считать ответ, и будем увеличивать её на 1 с каждым найденным подходящим словом:

```
>>> cnt = 0
>>> for i in words:
    if i.count('k') == 1:
        cnt += 1
>>> cnt
48
```

Ещё одним способом решения задачи является перебор букв в слове с использованием вложенных циклов: в первом цикле перебирается значение первой буквы слова, во втором цикле — значение второй буквы, в третьем цикле — значение третьей буквы. Из отдельных букв в результате применения оператора «+» образуется строка, для которой подсчитывается количество вхождений буквы «К». Если количество вхождений равно 1, то значение переменной `cnt` увеличивается на 1.

```
letters = 'wkola'
cnt = 0
for a in letters:
    for b in letters:
        for c in letters:
            if (a + b + c).count('k') == 1:
                cnt += 1
print(cnt) # выводит 48
```

При решении подобных задач с использованием Python могут оказаться полезными такие методы, как `count` (подсчёт количества образцов в объекте), `index` (индекс первого вхождения), а также оператор вхождения `in`. Рассмотрим ещё одну задачу.

Полина составляет таблицу кодовых слов для передачи сообщений, каждому сообщению соответствует своё кодовое слово. В качестве кодовых слов Полина использует шестибуквенные слова, в которых могут быть только буквы П, О, Л, И, Н, А, причём каждая буква

С учётом возможности проверить наличие вхождения в слово буквосочетания «ЛА» есть и ещё одно решение — сгенерировать перестановки и перед подсчётом перевести каждую из них в обычную строку. После этого можно будет дословно сделать то, что требуется в условии:

```
>>> from itertools import permutations
>>> words = list(permutations('polina'))
>>> for i in range(len(words)):
    words[i] = ''.join(words[i]) # преобразование кортежей в строки
>>> words[:5] # смотрим первые 5 полученных элементов
['polina', 'polian', 'polnia', 'polnai', 'polain']
>>> cnt = 0
>>> for i in words:
    if 'la' not in i:
        cnt += 1
>>> cnt
600
```

В задачах перебора также встречаются задачи на перебор числовых последовательностей. Решим подобную задачу.

Сколько существует десятичных шестизначных чисел, в которых все цифры различны и никакие две чётные или две нечётные цифры не стоят рядом?

Возможно решение данной задачи при помощи перебора всех шестизначных чисел в цикле for. Для каждого числа с помощью вложенного цикла while определяется, что никакие две чётные или две нечётные цифры не стоят рядом. Длина множества numSet, полученного из числа, преобразованного к строке (len(set(str(number)))), позволяет получить количество различных цифр в данном числе. Программа, работающая по данной логике, получится сложнее:

```
cnt = 0
for number in range(100000, 1000000):
    numCopy = number
    isDifferent = True
    prevParity = numCopy % 2
    numCopy //= 10
    while numCopy > 0:
        if numCopy % 2 == prevParity:
            isDifferent = False
        prevParity = numCopy % 2
        numCopy //= 10
    numSet = set(str(number))
    if len(numSet) == 6 and isDifferent:
        cnt += 1
print(cnt) # выводит 6480
```

Однако и здесь возможно более простое переборное решение при помощи вложенных циклов. Первой цифрой шестизначного числа может быть любая цифра, кроме 0. В теле цикла максимальной вложенности создаётся множество из цифр s . Если количество цифр (длина множества) равно 6, и все пары цифр имеют разную чётность, счётчик количества подходящих чисел увеличивается на 1.

```
cnt = 0
for a in range(1, 10):
    for b in range(0, 10):
        for c in range(0, 10):
            for d in range(0, 10):
                for e in range(0, 10):
                    for f in range(0, 10):
                        s = {a, b, c, d, e, f}
                        if (len(s) == 6 and
                            a % 2 != b % 2 and
                            b % 2 != c % 2 and
                            c % 2 != d % 2 and
                            d % 2 != e % 2 and
                            e % 2 != f % 2):
                            cnt += 1

print(cnt) # выводит 6480
```

Возможное решение через функцию `permutations` модуля `itertools`, генерирующую перестановки без повторяющихся элементов:

```
from itertools import permutations
numbers = list(permutations('0123456789', 6))
cnt = 0
for number in numbers:
    strNum = ''.join(number) #конструирование строки из кортежа
    if (strNum[0] != '0' and
        int(strNum[0]) % 2 != int(strNum[1]) % 2 and
        int(strNum[1]) % 2 != int(strNum[2]) % 2 and
        int(strNum[2]) % 2 != int(strNum[3]) % 2 and
        int(strNum[3]) % 2 != int(strNum[4]) % 2 and
        int(strNum[4]) % 2 != int(strNum[5]) % 2):
        cnt += 1
print(cnt) # выводит 6480
```

Рассмотренные методы позволяют решать комбинаторные задачи на подсчёт количества слов и чисел. Наиболее полезными они могут быть для учеников, увлекающихся программированием. Однако самый надёжный способ получить правильный ответ — решить двумя способами (письменно и при помощи программы) и сверить ответы. ЕГЭ в новом компьютерном формате предоставляет такую возможность, которой авторы статьи настоятельно рекомендуют воспользоваться.