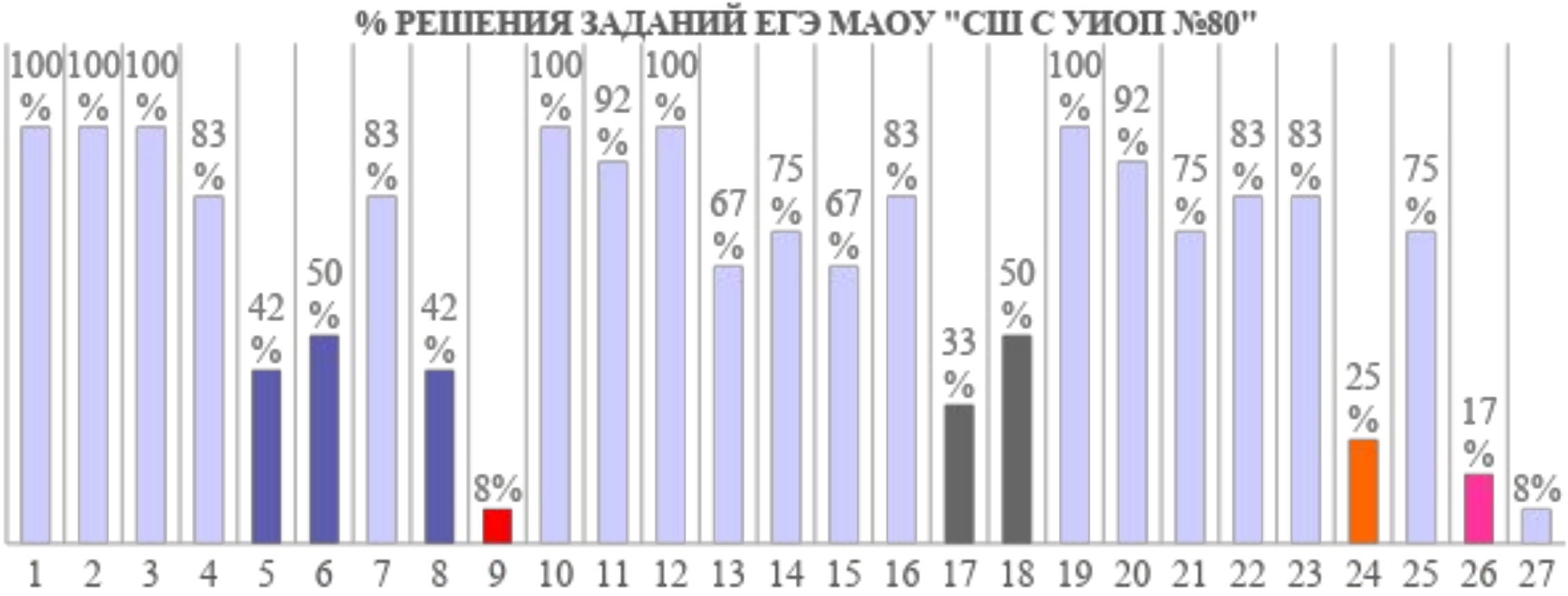
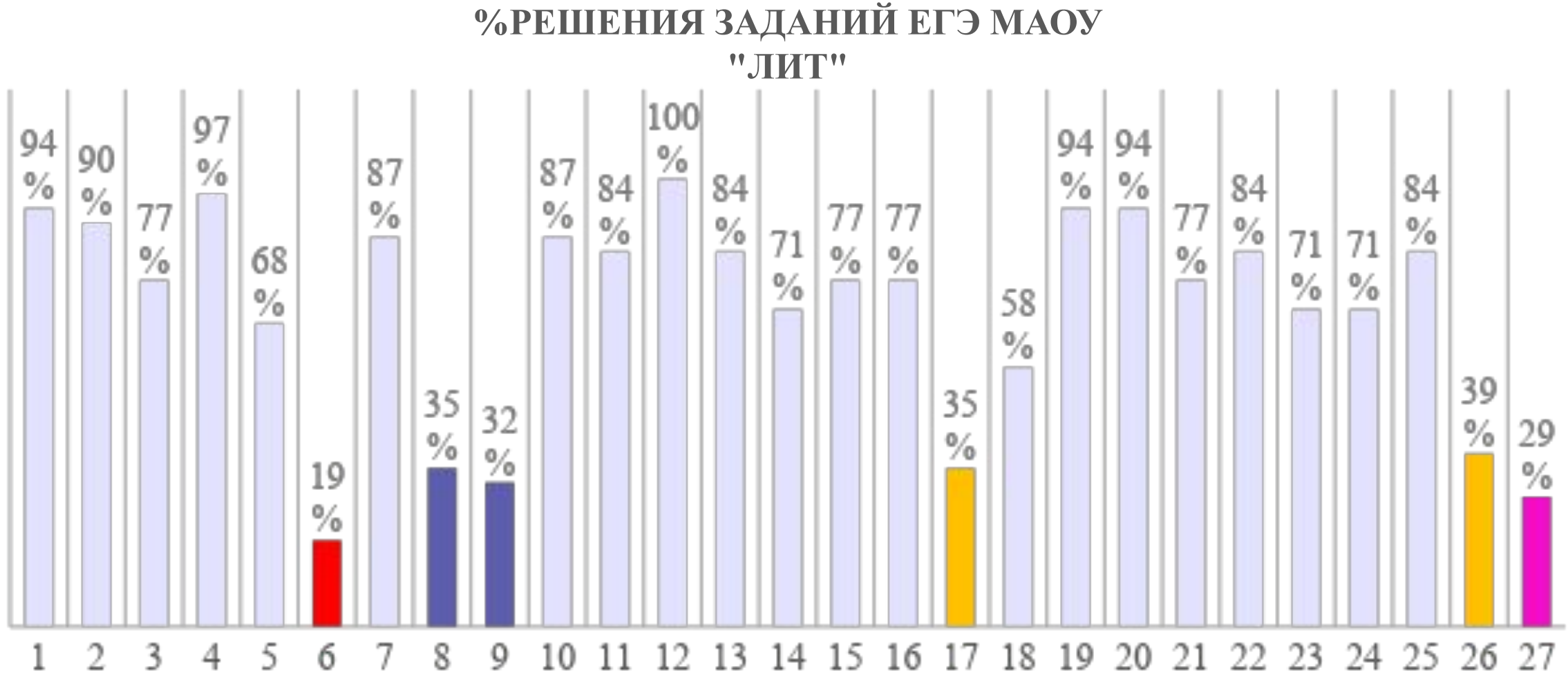
Линия 17 ЕГЭ
Перебор последовательности целых чисел. Проверка делимости

Лавинова Татьяна Валерьевна,
учитель информатики и ИКТ МАОУ «ЛИТ»

Результаты ЕГЭ по информатике в 2023 г.



Результаты ЕГЭ по информатике в 2023 г.



Результаты ЕГЭ по информатике в 2023 г. (РФ)

Что проверяется:

- Умение создавать собственные программы (20–40 строк) для обработки целочисленной информации.
- *1.7.2. Основные конструкции языка программирования. Система программирования.*
- *1.1.5. Умение создавать программы на языке программирования по их описанию.*

Наиболее частые ошибки:

- выход за пределы массива;
- обработка отрицательных чисел;
- неправильное прочтение задания (ошибки в записи условия отбора значений).

Что нужно знать:

- в известных задачах этого типа (не олимпиадных) нет ограничения на время выполнения, по крайней мере, оно несущественно для отрезков, заданных для перебора; поэтому можно использовать простой перебор без оптимизации;
- задачи этого типа предлагается решать с помощью электронных таблиц или собственной программы; как правило, написать правильную программу значительно проще;
- проверку условия удобно оформить в виде функции, возвращающей логическое значение (True/False), но можно этого и не делать.

Что нужно знать:

Цикл с условием:

```
while <условие>:
    <тело цикла>
```

Цикл с параметром:

```
for i in range (n1, n2):
    <тело цикла>
```

Что нужно знать:

```
count = 0              #количество
summa=0                #сумма
for n in range(a, b+1):
   if условие выполнено:
      count += 1
      summa += n
print( count )
```

Что нужно знать:

Количество и сумма цифр многозначного числа:

```
count = 0
sum = 0
while a > 0:
      sum += a%10
      count += 1
      a = a // 10
```

Что нужно знать:

Поиск максимального и минимального элементов:

```
maxa = <наименьший из возможных>
mina = <наибольший из возможных>
if  a>maxx: maxa = a    # maxa = max(maxa, a)
if  a<minx: mina = a    # mina = min(mina, a)
```

Что нужно знать:

```
def   <имя_процедуры> (<список параметров>):
      <операторы>
```

Для того чтобы процедура заработала, её необходимо вызвать по имени; причём таких вызовов может быть сколько угодно.

Для вызова функции достаточно указать её имя со списком фактических параметров в любом выражении, в условиях (после слов **if, while**) или в операторе **print** главной программы.

Что нужно знать:

Определить, является ли число ***n*** простым

…

```
count=0
for a in range (2, n//2+1):
    if (n % a) == 0:
        count += 1
if count = 0:
    p = True
```

Что нужно знать:

Число ***a*** кратно числу ***c***	a%c==0
Последняя цифра числа ***a*** в системе счисления с основанием ***b***	a % b
Две последние цифры числа в системе счисления с основанием ***b***	$a \% b^2$
Предпоследняя цифра числа	a %100//10
Число ***a*** содержит ***n*** цифр в системе счисления с основанием ***b***	$b^{n-1}<=a<b^n$
Перевод числа из десятичной системы счисления в двоичную строку 0b…	bin (x)
Перевод числа из десятичной системы счисления в восьмеричную строку 0o…	oct (x)
Перевод числа из десятичной системы счисления в шестнадцатеричную строку 0x…	hex (x)

Что нужно знать:

Перевод десятичного числа в систему счисления с основанием ***b***:

```
while a > 0:
        d = a % b      #последняя цифра числа
        a = a // b
```

Что нужно знать:

Обработка отрицательных чисел

1) округление «вниз» при делении:

-15 // 2 = -8

-15 % 2 = 1

2) количество цифр числа:

$10^{n-1} <= abs(x) < 10^{n}$

Что нужно знать:

Чтение данных из файла:

1 способ

```
F = open("….txt")
a = [int(x) for x in F.readlines()]
```

2 способ

```
a=[int(x) for x in open('….txt')]
```

Что нужно знать:

Работаем с парами чисел массива ***a***:

```
for i in range (len(a)-1):
```

…

Работаем с тройками чисел массива ***a*** :

```
for i in range (len(a)-2):
```

…

Что нужно знать:

Модуль для работы с масками:

```
from fnmatch import fnmatch
…
if fnmatch( str(c), 'маска' )
```

Что нужно знать:

`zip ()`

– это функция Python, которая позволяет объединить в кортежи элементы нескольких списков, кортежей или других итерируемых объектов, чтобы потом можно было обработать все кортежи в цикле

```
for a, b, c in zip(data, data[1:], data[2:]):
```

№ 17-152 (В. Шубинкин)

В файле **17-1.txt** содержится последовательность целых чисел. Элементы последовательности могут принимать целые значения от -10 000 до 10 000 включительно. Определите и запишите в ответе сначала количество пар элементов последовательности, в которых ровно одно число делится на 9, а другое при этом заканчивается на 3 в восьмеричной системе счисления. Затем - максимальное число в паре среди всех таких пар. В данной задаче под парой подразумевается два идущих подряд элемента последовательности.

```
max_a=-100000
count=0
a = [int(x) for x in open("17-1.txt")]
for i in range (0, len(a)-1):
    if (a[i]%9==0 and a[i+1]%9!=0 and a[i+1]%8==3
        or a[i+1]%9==0 and a[i]%9!=0 and a[i]%8==3):
        count+=1
        max_a=max(max_a, a[i], a[i+1])
print (count, max_a)
```

```
max_a=-100000
count=0
a = [int(x) for x in open("17-1.txt")]
for b, c in zip(a, a[1:]):
    if (b%9==0 and c%9!=0 and c%8==3
        or c%9==0 and b%9!=0 and b%8==3):
        count+=1
        max_a=max(max_a, b, c)
print (count, max_a)
```

№ 17-158 (В. Шубинкин)

В файле **17-1.txt** содержится последовательность целых чисел. Элементы последовательности могут принимать целые значения от -10 000 до 10 000 включительно. Определите и запишите в ответе сначала наибольшую длину убывающей подпоследовательности, затем количество убывающих подпоследовательностей такой длины. Под убывающей подпоследовательностью подразумевается последовательность подряд идущих элементов, каждый из которых меньше предыдущего. Например, в последовательности 7; -12; 10; 4; 7; -12; 10; -12; 3 наибольшая длина убывающей подпоследовательности равна 2, количество таких подпоследовательностей равно 4.

```
a=[int(x) for x in open("17-1.txt")]
count=0
maxlen=0
k=1
for i in range (1, len(a)):
    if a[i-1]>a[i]:
        k+=1
        if maxlen<k:
            maxlen=k
            count=0
        if k==maxlen:
            count+=1
    else:
        k=1
print (maxlen, count)
```

№ 17-198 (Л. Шастин)

В файле **17-10.txt** содержится последовательность целых чисел. Элементы последовательности могут принимать значения от 0 до 10000 включительно. Определите сначала количество пар, сумма элементов которых при переводе в систему счисления с основанием 7 образует число-палиндром, а затем наибольшую сумму-палиндром **в семеричной системе** счисления. Под парой чисел подразумевается два идущих подряд элемента последовательности.

```
def f(n):
    s=''
    while n>0:
        s+=str(n%7)
        n//=7
    return s[::-1]

maxa=0
count=0
smax=0
a = [int(x) for x in open("17-10.txt")]
for i in range(0, len(a)-2):
    if f(a[i]+a[i+1])==f(a[i]+a[i+1])[::-1]:
        count+=1
        smax=max(smax, int(f(a[i]+a[i+1])))

print (count, smax)
```

№ 17-290 (А. Брейк)

В файле **17-290.txt** содержится последовательность целых чисел. Элементы последовательности могут принимать целые значения от 0 до 10 000. Запишите в ответе количество троек элементов последовательности, в которых хотя бы одно число оканчивается на 1 в пятеричной системе счисления и все числа имеют длину 4 в своей шестеричной записи. Затем запишите максимальную разность между максимальным и минимальным числами в таких тройках. В данной задаче под тройкой подразумевается три идущих подряд элемента последовательности.

```
q=[int(x) for x in open('17-290.txt')]
count=0
maxr=0
for a, b, c in zip(q, q[1:], q[2:]):
    if (a%5==1)+(b%5==1)+(c%5==1)>0:
        if (6**3<=a<6**4)*(6**3<=b<6**4)*(6**3<=c<6**4)==1:
            count+=1
            maxr=max(maxr, max(a, b, c)-min(a, b, c))
print (count, maxr)
```

№ 17-380 (ЕГЭ-2023)

В файле **17-380.txt** содержится последовательность целых чисел, не превышающих по модулю 100 000. Определите количество троек элементов последовательности, в которых не более двух из трёх элементов являются четырёхзначными числами, а сумма элементов тройки не больше максимального элемента последовательности, оканчивающегося на 25. В ответе запишите количество найденных троек чисел, затем максимальную из сумм элементов таких троек. В данной задаче под тройкой подразумевается три идущих подряд элемента последовательности.

```
a=[int(x) for x in open('17-380.txt')]
count=0
maxsum=0
max25=max(x for x in a if abs(x)%100==25)
for x, b, c in zip(a, a[1:], a[2:]):
    if (1000<=abs(x)<10000)+(1000<=abs(b)
          <10000)+(1000<=abs(c)<10000)<=2 and x+b+c<=max25:
        count+=1
        maxsum=max(maxsum, x+b+c)
print (count, maxsum)
```

№ 17-347 (П. Финкель)

В файле **17-346.txt** содержится последовательность целых чисел. Элементы последовательности могут принимать целые значения от 1 до 200 000 включительно. Определите количество троек последовательности, для которых произведение всех цифр трёх чисел не превосходит $2 \cdot 10^9$ и удовлетворяет маске «53*7*». В качестве ответа укажите количество таких троек и наибольшее произведение их цифр. В данной задаче под тройкой подразумевается три идущих подряд элемента последовательности.

```
from fnmatch import fnmatch

def f(a):
    p=1
    while a>0:
        p*=a%10
        a//=10
    return p

a=[int(x) for x in open("17-346.txt") ]
count=0
maxp=0
for i in range (len(a)-2):
    c=f(a[i])*f(a[i+1])*f(a[i+2])
    if c<=2*(10**9) and fnmatch( str(c), '53*7*' ):
        count+=1
        if c>maxp:
            maxp=c
print (count, maxp)
```