Сложные задачи ЕГЭ по информатике и способы их решения

Исламов Ришат Габитович,
учитель информатики МБОУ СОШ № 10 с УИОП
Сургут 2022 год

Рекурсия. Рекурсивные процедуры и функции (Задание 16)

Что нужно знать:

1. **Для того, чтобы задать рекурсивную функцию, нужно определить условие окончания рекурсии, то есть значения параметров функции, для которых значение функции известно или вычисляется без рекурсивных вызовов;**
2. **Рекуррентную формулу (или формулы), с помощью которых значение функции для заданных значений параметров вычисляется через значение (или значения) функции для других значений параметров (то есть, с помощью рекурсивных вызовов)**

Пример:

$F(n) = 1$ при $n \leq 1$

$F(n) = 2*n + F(n-1)$, при $n > 1$

Ниже записана рекурсивная функция (процедура) F. Что выведет программа при вызове F(9)? В ответе запишите последовательность выведенных цифр слитно (без пробелов). 8

```
procedure F(n: integer);
begin
write(n);
if n >= 7 then
begin
F (n - 3);  F (n - 1)
end   end;
```

Сначала необходимо изучить текст программы на одном из языков программирования и понять, что выполняет данная функция. Функция получает на вход одно число n, выводит его на экран, затем при условии, что n >= 7 осуществляет два последовательных вызова F(n - 3) и F(n - 1), что приведет к печати меньших значений n и дальнейшим рекурсивным вызовам.

Выпишем рекуррентное соотношение для общего случая:

F(n) = n, F(n - 3), F(n - 1), при n >= 7;

F(n) = n, при n < 7.

Далее заполним таблицу, что выведет функция при вызове для разных значений n:

n	Рекуррентное	Результат вызова
	соотношение для F(n)	функции F(n)
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7, F(4), F(6)	746
8	8, F(5), F(7)	85746
9	9, F(6), F(8)	9685746

Дан рекурсивный алгоритм: (**Статград 2019 г**)

```
procedure F(n: integer);
begin
 writeln(n);
 if n < 6  then
       begin   F(n+2); F(n*3)  end
end;
```

Найдите сумму чисел, которые будут выведены при вызове F(1).

1.сначала определим рекуррентную формулу; обозначим через F(n) сумму чисел, которая выводится при вызове F(n)

2.при n >= 6 процедура выводит число n и заканчивает работу без рекурсивных вызовов:**F(n) = n при n >= 6**

3.при n < 6 процедура выводит число n и дважды вызывает сама себя:

F(n) = n + F(n+2) + F(3n) при n < 6

4.в результате вызова F(1) получаем

F(1) = 1 + F(3) + F(3); F(3) = 3 + F(5) + F(9) = 3 + F(5) + 9

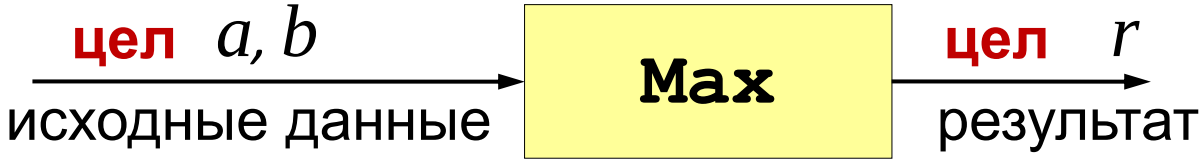
F(5) = 5 + F(7) + F(15) = 5 + 7 + 15 = 27

используем обратную подстановку:

F(3) = 3 + F(5) + 9 = 3 + 27 + 9 = 39 F(1) = 1 + F(3) + F(3) Ответ: 79.

Максимум из двух (трёх) чисел

Задача. Составить функцию, которая определяет наибольшее из двух целых чисел.




```
def Max(a, b):
    if a > b then
        return a
    else:
        return b
```

?

Как с её помощью найти максимум из трёх?

return – вернуть

```
def Max3(a, b, c):
    return Max( Max(a,b), c )
```

Функция — это вспомогательный алгоритм, который возвращает результат (число, строку символов и др.).

Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:
F(1) = 1
F(n) = F(n–1) * (n + 2), при n > 1
Чему равно значение функции F(5)? В ответе запишите только целое число.

```
def F( n ):
    if n == 1:
        return 1
    elif (n > 1):
        return F(n-1)*(n+2)
print (F(5))
```

Алгоритм вычисления функции F(n) задан следующими соотношениями:

F(n) = 1 при n = 1

F(n) = n + F(n–1), если n чётно,

F(n) = 2· F(n–2), если n > 1 и n нечётно.

Чему равно значение функции F(26)?

программа на языке Python:

```
def F( n ):
  if n == 1: return 1
  if n % 2 == 0:
    return n + F(n-1)
  else:
    return 2 * F(n-2)
print( F(26) )
```

Р-03. Определите наименьшее значение n, при котором сумма чисел, которые будут выведены при вызове $F(n)$, будет больше 500000. Запишите в ответе сначала найденное значение n, а затем через пробел – соответствующую сумму выведенных чисел.

Python	Паскаль	C++

def F(n):
  print(2*n)
  if n > 1:
  print(n-5)
  F(n-1)
  F(n-2)
	procedure F
	  (n: integer);
	begin
	  writeln(2*n);
	  if n > 1 then begin
	  writeln(n-5);
	  F(n-1);
	  F(n-2);
	  end;
	end;
		void F(int n)
		{
		  cout << 2*n << endl;
		  if(n > 1) {
		  cout << n-5 << endl;
		  F(n-1);
		  F(n-2);
		  }
		}

```
def f( n ):
  if n <= 1:
    return 2*n
  else:
    return 2*n+n-5+f(n-1)+f(n-2)
n = 0
while True:
  n += 1
  s = f(n)
  if s > 500000: break;
print( n, s )
```

76. Алгоритм вычисления функции F(n), где n – целое число, задан следующими соотношениями:
F(n) = 1, при n <= 1,
F(n) = 3 + F(n / 2 – 1), когда n > 1 и чётное,
F(n) = n + F(n + 2) , когда n > 1 и нечётное.
Назовите минимальное значение n, для которого F(n) = 19.

```
def F(n):
  if n <= 1: return 1
  if n % 2 == 0:
    return 3 + F(n//2 - 1)
  else:
    return n + F(n+2)
n=0
while True:
  n += 1
  r = F(n)
  if r == 19:
      break
print(n, r)
```

76. Алгоритм вычисления функции F(n), где n – целое число, задан следующими соотношениями:
F(n) = 1, при n <= 1,
F(n) = 3 + F(n / 2 – 1), когда n > 1 и чётное,
F(n) = n + F(n + 2) , когда n > 1 и нечётное.
Назовите минимальное значение n, для которого F(n) = 19.

Для обработки исключений используется конструкция try - except. В блоке try мы выполняем инструкцию, которая может породить исключение, а в блоке except мы перехватываем их.

```
def F(n):
  if n <= 1: return 1
  if n % 2 == 0:
    return 3 + F(n//2 - 1)
  else:
    return n + F(n+2)
n = 1
while True:
  try:
    r = F(n)
  except:
    pass
  else:
    print(n, r)
    if r == 19:
      break
  n += 1
```

81.Алгоритм вычисления функции F(n), где n – целое число, задан следующими соотношениями:

F(n) = n, при n <=5,

F(n) = n + F(n / 2 – 1), когда n > 5 и делится на 4,

F(n) = n + F(n + 2) , когда n > 5 и не делится на 4.

Назовите максимальное значение n, для которого возможно вычислить F(n).

```
import sys
sys.setrecursionlimit(100)
def F(n):
  if n <= 5: return 1
  if n % 5 == 0:
    return n + F(n//5 + 1)
  else:
    return n + F(n+6)
n = 1
while True:
  try:
    r = F(n)
  except:
    pass
  else:
    print(n, r)
    if r > 1000:
      break
  n += 1
```

83. Алгоритм вычисления функции F(n), где n – целое число, задан следующими соотношениями:
F(n) = n, при n ⍰ 5,
F(n) = n + F(n / 2 – 3), когда n > 5 и делится на 8,
F(n) = n + F(n + 4) , когда n > 5 и не делится на 8.
Назовите максимальное значение n, для которого возможно вычислить F(n).

```
def f(n):
    if n <= 5:
        return n
    elif n > 5 and n % 8 == 0:
        return n + f(n / 2 - 3)
    else:
        return n + f(n + 4)
maxx = 0
for i in range(0, 1000):
    try:
        m = f(i)
    except Exception:
        continue
    maxx = max(maxx, i)
print(maxx)
```

99. Алгоритм вычисления функции F(n), где n – целое неотрицательное число, задан следующими соотношениями:
F(0) = 0,
F(n) = 1, когда 0 < n < 3,
F(n) = F(n – 2) + F(n – 1) , когда n ≥ 3.
Определите четыре последние цифры числа F(47).

```
from functools import lru_cache
@lru_cache(None)
def F(n):
    if n == 0:
        return 0
    elif 0 < n < 3:
        return 1
    else:
        return F(n-2) + F(n-1)
print(F(47))
# 2971215073
# 5073
```

Обработка символьных строк (Задание 24)

Что проверяется:

1. **Умение создавать собственные программы (10–20 строк) для обработки символьной информации.**
2. ***Цепочки (конечные последовательности), деревья, списки, графы, матрицы (массивы), псевдослучайные последовательности.***
3. ***Строить информационные модели объектов, систем и процессов в виде алгоритмов.***

Демоверсия варианта ЕГЭ по информатике 2021, ФИПИ:

Текстовый файл состоит не более чем из 10^6 символов **X, Y** и **Z**.

*Определите максимальное количество идущих подряд символов, среди которых **каждые два соседних различны**.*

Для выполнения этого задания следует написать программу.

```
f=open('24.txt') # открыть файл
s=f.readline() # считали строку
m=1 # макс длина цепочки
k=1 # начальная длина цепочки
for i in range(1, len(s)): # будем перебирать в цикле все символы, начиная с s[1] (второго по счёту) до конца строки, постоянно «оглядываясь назад», на предыдущий символ
    if s[i]!=s[i-1]: # обработать пару символов s[i] и s[i-1]
        k+=1
        m=max(k,m) # перезаписали Макс длину
    else:
        k=1 #  сбрасываем счетчик
print(m)
```

Ответ 35

Задание1

Задание выполняется с использованием прилагаемых файлов
В текстовом файле **k7-0.txt** находится цепочка из символов латинского алфавита **A**, **B**, **C**. *Найдите длину* ***самой длинной подцепочки****, состоящей из символов* ***C****.* ***Ответ: 0***

```
with open("k7-0.txt") as Fin:
  s = Fin.readline()
 c = 'C'
while c in s: # ищем CC, потом CCC и т.д
  c += 'C'
print(len(c)-1 )
# минус 1, чтобы убрать лишнюю (последнюю добавленную C)

f=open('k7-0.txt')
s = f.readline() # считали строку
m = 0 # макс длина цепочки
l = 0 # начальная длина цепочки
for i in range(0,len(s)):
    if s[i]=='C':
        l+=1
        m = max(l,m) # перезаписали Макс длину
    else:
        l = 0 #  сбрасываем счетчик
print(m)
```

Задание 21

В текстовом файле **k7a-1.txt** находится цепочка из символов латинского алфавита **A, B, C, D, E**. *Найдите длину* ***самой длинной подцепочки****, состоящей из символов* ***A****,* ***B*** *или* ***C*** *(в произвольном порядке).*

```
f = open('k7a-1.txt')
s = f.readline() # считали строку
m = 0 # макс длина цепочки из "A,B,C"
l = 0 # начальная длина цепочки из "A,B,C"
for i in range(0,len(s)):
    if s[i] in'ABC':
        l+=1
        m = max(l,m) # перезаписали Макс длину
    else:
        l = 0 # другая буква - сбрасываем счетчик
print(m)
```

Ответ: 16

Задание 26

В текстовом файле **k7a-6.txt** находится цепочка из символов латинского алфавита **A, B, C, D, E, F**.

*Найдите длину **самой длинной подцепочки**, **не содержащей гласных букв**.*

```
with open("k7a-6.txt") as F:
  s = F.readline() # считали строку
 k = 0 # начальная длина цепочки из "B,C,D,F"
Max = 0 # макс длина цепочки из "B,C,D,F"
for c in s:
  if c in 'BCDF':
    k += 1
    if k > Max:
      Max = k # перезаписали Макс длину
  else:
      k = 0 # другая буква - сбрасываем счетчик
print(Max)
```

Ответ: 20

Задание 33

В текстовом файле k7c-1.txt находится цепочка из символов латинского алфавита A, B, C, D, E.

Найдите количество цепочек длины 3, удовлетворяющих следующим условиям:

1-й символ – один из символов B, C или D;

2-й символ – один из символов B, D, E, который не совпадает с первым;

3-й символ – один из символов B, C, E, который не совпадает со вторым.

Ответ: 1280

```
with open("k7c-1.txt") as Fin:
        s = Fin.readline()
k = 0
c1 = 'BCD‘ # строка проверки первого символа
c2 = 'BDE‘ # строка проверки второго символа
c3 = 'BCE‘ # строка проверки третьего символа
for i in range(len(s)-2):
    if s[i] in c1 and s[i+1] in c2 and s[i+2] in c3 \
       and s[i]!=s[i+1] and s[i+1]!=s[i+2]: # проверка повтора символов
      k += 1
print(k)
```

Задание 52

В текстовом файле k8-0.txt находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.

Ответ: 2 3

```
f=open('k8-0.txt')
s=f.readline() # считали строку
k=1 # начальная длина цепочки
m=0 # макс длина цепочки
for i in range (1,len(s)):
    if s[i]==s[i-1] :
        k+=1 # увеличиваем счетчик длины последовательности
        if k>m: # Запомнить символ, из которого строится эта подцепочка и
длину этой подцепочки
            m=k
            symb=s[i]
    else:
        k=1 # сбрасываем счетчик
 print(symb,m)
```

Задание 108

Текстовый файл 24.txt содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей убывающей подпоследовательности.

Ответ: 3

```
F= open("24.txt")
s = F.readline() # считали строку (весь текст файла в одной строке)
k = 1
maxim = 0
for i in range(1, len(s)):
    if s[i] < s[i - 1]:
        k += 1 # увеличиваем счетчик длины последовательности
        if k > maxim:
            maxim = k
    else:
        k = 1 # сбрасываем счетчик для работы со след. последовательностью
print(maxim)
```

Задание 137

Текстовый файл 24-s1.txt состоит не более чем из 10^6 заглавных латинских букв (A..Z). Текст разбит на строки различной длины. Определите количество строк, в которых буква J встречается чаще, чем буква E.

Ответ: 482

```
f= open("24-s1.txt")
k = 0 # счетчик строк
while True: # бесконечный цикл
    s = f.readline() # считываем очередную строку
    if not s: break # если строка пустая (т.е. достигнут конец файла), выходим
    if s.count("J") > s.count("E"):
        k +=1
print(k)
```

Обработка целых чисел. Проверка делимости (задание 25)

Что нужно знать:

В задачах этого типа нет ограничения на время выполнения, поэтому можно использовать простой перебор

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

Разбор 25 задания ЕГЭ по информатике с сайта К. Полякова № 1:
Напишите программу, которая ищет среди целых чисел, принадлежащих числовому отрезку [126849; 126871], числа, имеющие ровно 4 различных делителя. *Выведите эти четыре делителя для каждого найденного числа в порядке возрастания.*

1 способ:

```
for n in range(126849,126871+1):
    dl = [] # чистим список делителей
    for d in range(1,n+1): #
      if n % d == 0:
        dl = dl + [d] # добавляем
делитель в список
        if len(dl) > 4:
            break
    if len(dl) == 4:
      print(*dl)
```

2 способ:

```
for n in range(126849, 126871+1):
  a = [] # массив хранения делителей
  for d in range(1,n//2+1):
    if n % d == 0:
      a.append(d)
      if len(a) > 3: break
  if len(a) == 3: # добавим n
    a.append(n)
    print(*a)
```

Генерация списка делителей.

Общая идея: Для каждого числа указанного диапазона генерируем список делителей. Если длина списка равна четырем, выводим его.

```
For n in range(126849, 126871+1):
  dl = [d for d in range(1, n+1) if n % d == 0]
  if len(dl) == 4:
    print( *dl )
```

Разбор 25 задания ЕГЭ по информатике с сайта К. Полякова № 11:

Напишите программу, которая ищет среди целых чисел, принадлежащих числовому отрезку [164700; 164752], числа, имеющие ровно 6 различных делителей. Выведите эти делители для каждого найденного числа в порядке возрастания.

```
start, end = 164700, 164752

for n in range( start, end+1 ):
  a = [] # массив для хранения делителей
  for d in range(1,n+1):
    if n % d == 0:
      a.append(d)
      if len(a) > 6: break
  if len(a) == 6:
    print( *a )
```

Генерация списка делителей.

```
for n in range(164700, 164752+1):
   dl= [d for d in range(1, n+1) if n % d == 0]
   if len(dl) == 6:
      print( *dl )
```

Разбор 25 задания ЕГЭ по информатике с сайта К. Полякова № 21:

Напишите программу, которая ищет среди целых чисел, принадлежащих числовому отрезку [190201; 190230], числа, имеющие ровно 4 различных делителя. Выведите эти четыре делителя для каждого найденного числа в порядке убывания.

```
for n in range(190201,190230+1):
    divs = [] # чистим список делителей
    for d in range(1,n+1): #
      if n % d == 0:
        divs = divs + [d] # добавляем делитель в список
        if len(divs) > 4: break
    if len(divs) == 4:
      divs.reverse()
      print(*divs)
```

Генерация списка делителей.

```
for n in range(190201, 190230+1):
  divs = [d for d in range(1, n+1) if n % d == 0]
  if len(divs) == 4:
    divs.reverse() # реверсируем (по убыванию)
    print( *divs )
```

Разбор 25 задания ЕГЭ по информатике с сайта К. Полякова № 22:

Напишите программу, которая ищет среди целых чисел, принадлежащих числовому отрезку [190201; 190280], числа, имеющие ровно 4 различных четных делителя. Выведите эти четыре делителя для каждого найденного числа в порядке убывания.

```
for n in range(190201,190280+1):
    divs = [] # чистим список делителей
    for d in range(1,n+1): #
      if n % d == 0 and d%2==0:
        divs = divs + [d] # добавляем делитель в список
        if len(divs) > 4: break
    if len(divs) == 4:
      divs.reverse()
      print(*divs)
```

Генерация списка делителей.

```
for n in range(190201, 190280+1):
    divs = [d for d in range(1, n+1) if n % d == 0 and d % 2 == 0]
    if len(divs) == 4:
        divs.reverse()
        print( *divs )
```

Разбор 25 задания ЕГЭ по информатике с сайта К. Полякова № 32:

Напишите программу, которая ищет среди целых чисел, принадлежащих числовому отрезку [394441; 394505], числа, имеющие максимальное количество различных делителей. Если таких чисел несколько, то найдите минимальное из них. Выведите количество делителей найденного числа и два наибольших делителя в порядке убывания.

```
maxim = 0  # нужное количество делителей
divsMax = []
for n in range(394441, 394505 + 1):
  divs = [] # чистим список делителей
  for d in range(1,n+1): # перебор делителей
    if n % d == 0:
      divs.append(d) # добавляем делитель в список
  if len(divs) > maxim:
    maxim = len(divs)
    divsMax = divs
divsMax.reverse()
print(maxim,divsMax[0],divsMax[1])
```

Генерация списка делителей

```
maxim=0
divsmax=[]
for n in range(394441, 394505+1):
   divs = [d for d in range(1, n+1) if n % d == 0]
   if len(divs) > maxim:
      maxim = len(divs)
      divsmax = divs # сохраняем делители для числа с макс кол-вом дел-ей
divsmax.reverse()
print(maxim, divsmax[0], divsmax[1])
```

Разбор 25 задания ЕГЭ по информатике с сайта К. Полякова № 60:

Напишите программу, которая ищет среди целых чисел, принадлежащих числовому отрезку [3532000; 3532160], простые числа.

Выведите все найденные простые числа в порядке убывания, слева от каждого числа выведите его номер по порядку.

```
count = 0
for n in range(3532160, 3532000-1, -1): # цикл с конца и с шагом (-1)
  flag = True
  for d in range(2, n): # перебор делителей, начиная с двух
    if n % d == 0: # есть делитель помимо единицы и самого n
      flag = False # число не простое
      break
  if flag == True: # число простое
    count+=1
    print(count , n)
```

132. Назовём нетривиальным делителем натурального числа его делитель, не равный единице и самому числу. Найдите все натуральные числа, принадлежащие отрезку [247264322; 369757523] и имеющие ровно три нетривиальных делителя. Для каждого найденного числа запишите в ответе само число и его наибольший нетривиальный делитель. Найденные числа расположите в порядке возрастания.

start, end = 247264322, 369757523

```
def isPrime( x ):
  if x <= 1: return False
  d = 2
  while d*d <= x:
    if x % d == 0:
      return False
    d += 1
  return True

q4s = int(start**0.25)
q4e = int(end**0.25) + 1
for q4 in range(q4s, q4e+1):
  x = q4**4
  if start <= x <= end  and  isPrime(q4):
    print( x, [q4, q4**2, q4**3] )
```

Три нетривиальных делителя имеют только(!) простые числа в четвертой степени

Генерация списка

```
p = []
for x in range(2,1000):
    d = [o for o in range(2,x//2+1) if x % o == 0]
    if len(d) == 0:
        p += [x]
for i in p:
    if i**4 > 247264322 and i**4 < 369757524:
        print(i**4,i**3)
```

```
def prnum(m, n) :
    res = set()
    prime = [True] * (n+1)
    for i in range(3, n + 1, 2) :
        if not prime[i]:
            continue
        if i > m:
            res.add(i)
        for j in range(i * i, n+1, i):
            prime[j] = False
    return sorted(list(res))
a = 123456789
b = 223456789
num_a = int(a**0.25)
num_b = int(b**0.25) + 1
for num in prnum(num_a, num_b):
    print(num**4, '->', num**3)
```

```
MIN = 862346
MAX = 1056242

# Функция, проверяющая является ли число простым
def is_prime(n):
    a = 2
    while a ** 2 <= n:
        if n % a == 0:
            return False
        a += 1
    return True

# Все простые числа от 2 от MAX ** 0.5
primes = [2] + list(filter(is_prime, range(3, int(pow(MAX, 0.5)) + 1, 2)))
# Перебираем все простые числа
for prime in primes:
    # Не имеет сымсла рассматривать прогрессию, состоящую из 3 и более чисел
    # 862346 <= x(x + 100)(x + 200)(x + 300) <= 1056242 - не решается в натуральных числах => не более 3 чисел в прогрессии  при  x=1:  1*101*201*301=6110601
    # 862346 <= x(x + 100)(x + 200) <= 1056242 - так как x - натуральное, то
    # x = 30: 30*130*230=897000 ; 31; 32; 33, если x - не простое, то нетривиальных делителей будет больше и тогда прогрессия не получится
    # тогда интересующее нас значение x = 31
    # так как 31 + 200 = 231 - не простое число => невозможно составить прогрессию из 3 и более чисел
    # тогда будем рассматривать числа вида n = x(x + 100), где x и x + 100 - простые числа
    n = prime * (prime + 100)
    if is_prime(prime + 100):
        if MIN <= n <= MAX:
            print(n,  prime + 100)
```

142. Рассматривается множество целых чисел, принадлежащих числовому отрезку [862346; 1056242]. Найдите числа, нетривиальные делители которых образуют арифметическую прогрессию с разностью d = 100. В ответе для каждого такого числа (в порядке возрастания) запишите сначала само число, а потом – его максимальный нетривиальный делитель.

Генерация списка

```
p = []
for x in range(2,1100):
    d = [o for o in range(2,x//2+1) if x % o == 0]
    if len(d) == 0:
        p += [x]
for i in p:
    if (i+100) in p:
        if (i * (i+100)) > 862346 and  (i * (i+100)) < 1056242:
            print(i*(i+100),i+100)
```

Разбор 27 задания ЕГЭ по информатике (К.Ю. Поляков 2 вариант)

Дана последовательность из N натуральных чисел. Рассматриваются всевозможные пары различных элементов последовательности, между которыми есть хотя бы одно число, при этом сумма пары кратна трём, а сумма чисел между ними чётна. Найдите количество таких пар.

Входные данные. Даны два входных файла (файл A и файл B), каждый из которых содержит в первой строке количество чисел N (2 ≤ N ≤ 500000). Каждая из следующих N строк содержит натуральное число, не превышающее 10000.

Пример входного файла:

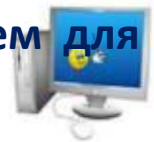
```
7
1
3
4
93
8
5
95
```

Решение 1 балл:

```
with open('27-76a.txt') as f:
    n = f.readline()
    m = [int(i.strip()) for i in f.readlines()] # заполнить массив
pairs = 0
for i in range(len(m)):
    for j in range(i + 2, len(m)):
        if (m[i] + m[j]) % 3 == 0 and sum(m[i + 1:j]) % 2 == 0:
            pairs += 1
print(pairs)
```

В этом наборе под условие подходят пары 1 и 8; 1 и 5; 3 и 93; 4 и 95. Ответ: 4.

В ответе укажите два числа: сначала искомое значение для файла A, затем для файла B.



Решение 2 балла:

```
def solve(filename):
    osts = {0: [], 1: [], 2: []}  # Словарь со списками для индексов чисел с одинаковыми остатками на 3
    sums = []  # Частичные суммы
    answer = s = 0  # Ответ и сумма от первого до i-го элемента
    with open(filename) as f:
        n = int(f.readline())
        for i in range(n):
            ch = int(f.readline())
            # Чтобы сумма двух чисел делилась на 3, сумма их остатков должна нацело делится на 3
            if ch % 3 == 1:
                check = osts[2]
            elif ch % 3 == 2:
                check = osts[1]
            else:
                check = osts[0]
            # Перебирая все суммы, ищем чётные
            for j in check:
                sm = s - sums[j]
                if sm > 0 and sm % 2 == 0:
                    answer += 1
            # Частичные суммы
            s += ch
            sums.append(s)
            # Дополнение словаря osts
            osts[ch % 3].append(i)
    return answer
print(solve("27a.txt"))
print(solve("27b.txt"))  # Считается за 2 мин
```

Разбор 27 задания ЕГЭ по информатике (К.Ю. Поляков 2 вариант)

Дана последовательность из N натуральных чисел. Рассматриваются всевозможные пары различных элементов последовательности, между которыми есть хотя бы одно число, при этом сумма пары кратна трём, а сумма чисел между ними чётна. Найдите количество таких пар.

Пример входного файла:

```
7
1
3
4
93
8
5
95
```

```
f = open('27-76b.txt')
n = int(f.readline())
prev = [ [0,0],[0,0],[0,0] ]
count, chet_sum = 0,0
a = int(f.readline())
for i in range(n-1):
    b = int(f.readline())
    chet_sum += a % 2
    ost = b % 3
    count += prev[(3-ost)%3][chet_sum%2]
    prev[a%3][chet_sum%2]+=1
    a = b
print(count)
```

В этом наборе под условие подходят пары 1 и 8; 1 и 5; 3 и 93; 4 и 95. Ответ: 4.

В ответе укажите два числа: сначала искомое значение для файла A, затем для файла B.

Апробация ЕГЭ по информатике 19 февраля 2022 – задание №27

Дана последовательность из N натуральных чисел. Рассматриваются все её непрерывные подпоследовательности, такие что сумма элементов каждой из них кратна k = 67. Найдите среди них подпоследовательность с максимальной суммой. Укажите в ответе найденную максимальную сумму.

Входные данные

Даны два входных файла (файл A и файл B), каждый из которых содержит в первой строке количество чисел N (1 < N < 10 000 000). Каждая из следующих N строк содержит одно натуральное число, не превышающее 10 000.

Пример организации исходных данных во входном файле:

```
7
1
3
4
93
8
5
95
```

Решение 1 балл:

```
f = open('27_a.txt')
n = int(f.readline()) # чтение данных
a = [int(s) for s in f] # заполнение массива
m = 0 # начальное значение максимальной суммы
for i in range(n):
    s = 0 # обнулить сумму под последовательности
    for j in range(i, n):
        s += a[j] # сумму под последовательности
        if s % 67==0 and s>m:
            m=s
print(m)
```

В ответе укажите два числа: сначала значение искомой суммы для файла A, затем — для файла B.

```
from collections import defaultdict
# defaultdict() представляет собой словарь со значениями по умолчанию
k = 67
def solve(filename):
    # Мы знаем, что все числа - натуральные => каждая следующая префиксая сумма > преыдущей
    # Тогда максимальная разность сумм с одинаковым остатком - разность крайней левой перфексой суммы с таким остатком и крайней правой
    # Поэтому будем хранить для каждого остатка максимальную и минимальную префиксную сумму
    min_ost = defaultdict(int)
    max_ost = defaultdict(int)
    s = 0
    with open(filename) as f:
        n = int(f.readline())
        for _ in range(n):
            s += int(f.readline())
            # Если нет минимума, то не имеет смысла заполнять максимум
            if s % k not in min_ost:
                min_ost[s % k] = s
            else:
                max_ost[s % k] = s
    max_diff = max_ost[0]  # задаём начальное значение равное максимальной префиксной сумме с остатком 0
    # находим максимальную сумму, как разность максимальной и минимальной
    for ost, val1 in min_ost.items():
        if max_ost[ost]:
            val2 = max_ost[ost]
            if val2 - val1 > max_diff:
                max_diff = val2 - val1
    return max_diff
print(solve("27_a.txt"))
print(solve("27_b.txt"))
```

Апробация ЕГЭ по информатике 19 февраля 2022 – задание №27

Дана последовательность из N натуральных чисел. Рассматриваются все её непрерывные подпоследовательности, такие что сумма элементов каждой из них кратна k = 67. Найдите среди них подпоследовательность с максимальной суммой. Укажите в ответе найденную максимальную сумму.

Входные данные

Даны два входных файла (файл A и файл B), каждый из которых содержит в первой строке количество чисел N (1 < N < 10 000 000). Каждая из следующих N строк содержит одно натуральное число, не превышающее 10 000.

Пример организации исходных данных во входном файле:

```
7
1
3
4
93
8
5
95
```

Решение 2 балла:

```
f = open('27_b.txt')
n = int(f.readline()) # чтение данных
s = 0
res = []
maxs = [-1]*67
maxs[0]=0
for i in range(n):
    s += int(f.readline())
    ost = s%67
    if maxs[ost] != -1:
        res.append(s-maxs[ost])
    else:
        maxs[ost] = s
print(max(res))
```

Решение 2 балла:

```
input=open('27_b.txt').readline
k=67
p=[float('inf')]*k
p[0]=ans=sm=0
for _ in range(int(input())):
    sm+=int(input())
    ans=max(ans,sm-p[sm%k])
    p[sm%k]=min(p[sm%k],sm)
print(ans)
```

Разбор 27 задания ЕГЭ по информатике (К.Ю. Поляков 6 вариант)

Набор данных представляет собой последовательность натуральных чисел. Необходимо найти количество подпоследовательностей подряд идущих чисел, сумма которых делится на 71. Гарантируется, что такие подпоследовательности существуют.

Входные данные. Даны два входных файла (файл A и файл B), каждый из которых содержит в первой строке количество чисел N (2 ≤ N ≤ 108). Каждая из следующих N строк содержит натуральное число, не превышающее 10000.

Пример входного файла:

```
6
12
59
45
13
31
27
```

```
with open('27_6Va.txt') as f:
    line = f.readlines()
def r(o):
    return int(line[o])
c = 0
for i in range(1,50):
    n = r(i)
    for l in range(i+1, 51):
        if (n + r(l)) % 71 == 0:
            c += 1
        n += r(l)
print(c)
```

В этом наборе можно выбрать последовательности 12+59 (сумма 71), 13+31+27 (сумма 71). Ответ: 2. В ответе укажите два числа: сначала искомое значение для файла A, затем для файла B.

```
def solve(filename):
    answer = s = 0
    osts = {i: 0 for i in range(71)}  # Словарь количества сумм с определённым остатком
    with open(filename) as f:
        n = int(f.readline())
        for _ in range(n):
            ch = int(f.readline())
            s += ch
            answer += osts[s % 71]  # К ответу прибавляем количество сумм с таким же остатком
            osts[s % 71] += 1  # Увелличиваем количество сумм с остатком
            # Если сумма уже делится нацело, нужно её учесть
            if s % 71 == 0:
                answer += 1
    return answer
print(solve("27_6Va.txt"))
print(solve("27_6Vb.txt"))
```

Разбор 27 задания ЕГЭ по информатике 2021 года

Имеется набор данных, состоящий из пар положительных целых чисел. Необходимо выбрать из каждой пары ровно одно число так, чтобы сумма всех выбранных чисел не делилась на 3 и при этом была минимально возможной. Гарантируется, что искомую сумму получить можно. Программа должна напечатать одно число – минимально возможную сумму, соответствующую условиям задачи.

Входные данные:

Даны два входных файла: файл A (27-1a.txt) и файл B (27-1b.txt), каждый из которых содержит в первой строке количество пар N (1 ≤ N ≤ 100000). Каждая из следующих N строк содержит два натуральных числа, не превышающих 10 000.

*** Не требуется учитывать эффективность алгоритма (с 2021 года)!**

Пример входных данных:

6
1 3
5 12
6 9
5 4
3 3
1 1

Пример выходных данных для приведённого выше примера входных данных:

20

```
f = open ('27-1b.txt') # для первого ответа - 27-1a.txt
n=int(f.readline())
data=f.readlines()
summa=0
minim=10001 # для минимальной разницы
for i in range(0, n):
    s = data[i].split()
    a=int(s[0])
    b=int(s[1])
    summa+=min(a,b) # сумма максимумов из пар
    raznitsa = abs(a-b) # разность
    if raznitsa % 3 != 0:
        minim=min(minim, raznitsa)
if summa % 3 != 0:
    print(summa)
else:
    print(summa + minim) # здесь добавляем! т.к. иначе берем наибольший из пары
```

Исламов Ришат Габитович, учитель информатики
МБОУ СОШ № 10 с УИОП
islamov.rishat86@mail.ru

Решение ЕГЭ_ 2022 год _Задание 27_ К.Ю. Поляков 20 вариантов
https://drive.google.com/drive/folders/1QoUaf_ZGGAZttVD2o_rurw6ctiCKC6Wy?usp=sharing