## #2 (аналитическое решение)


```
print('x y z w')
 for x in 0,1:
  for y in 0,1:
   for z in 0,1:
    for w in 0,1:
     f = # функция из задания
     if f == # 0 или 1:
      print(x, y, z, w)
```

## #2 (полное решение)

```
from itertools import *

def f(x, y, z): # меняем если больше переменных
    return (x == z) or (x <= (y and z)) # меняем функцию

for i in product([0, 1], repeat=3):  # меняем репит в зависимости от пропусков
    table = [(0, 0, i[0]), (1, i[1], i[2])] # меняем строки таблицы

    for p in permutations('xyz'):
        if [f(**dict(zip(p, r))) for r in table] == [0, 0]: # меняем только F
            if len(table) == len(set(table)): # проверка одинаковых строк
                print(p)
```

## #3 (excel)

=НАИБОЛЬШИЙ(МАССИВ; #)
=ОСТАТ(число; число)
=ВПР(что ищем; где ищем; какой по счету столбец подтягиваем; 0)
=ЕСЛИ(условие; если да; если нет)
=И(условие1; ...; условие N)
=ИЛИ(условие1; ...; условие N)

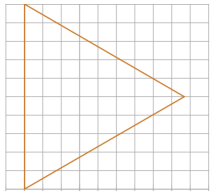
## #5

```
for n in range(1,70):
  s = bin(n)[2:]
  k = s.count("1")
  s += str(k%2)
  k = s.count("1")
  s += str(k%2)
  r = int(s,2)
  if r > 77:
  print(n)
  break
```

```
lst = []
for i in range(1, 100000):
  N = i
  n = bin(N)[2:]
  if N % 2 == 0:
    n = '1' + n + '10'
  else:
    n = '11' + n + '0'
  r = int(n, 2)
  if 800 <= r <= 1500:
    lst.append(r)
print(len(set(lst)))
```

## #6 (демо)

```
использовать Черепаха
алг
нач
  опустить хвост
  нц 7 раз
    вперед(10)
    вправо(120)
  кц
кон
```



## #6 (Python)

```
from turtle import *
k = 30 # масштаб
left(90)
for i in range(15):
    forward( k*4 )
    right(60)

up()
for x in range(0, 8):
  for y in range(0, 8):
    goto(x*k, y*k)
    dot(4)
```

## #7 (звук)

1 Мбайт = $2^{20}$ байт = $2^{23}$ бит,
1 Кбайт = $2^{10}$ байт = $2^{13}$ бит

объём музыкального файла вычисляется по формуле **I = k * f * B * t** , где f – частота дискретизации (Гц), B – разрешение (глубина кодирования, бит), k – количество каналов, t – время звучания (сек)

## #7 (картинки)

*Рисунок размером 512 на 256 пикселей занимает в памяти 64 Кбайт (без учёта сжатия). Найдите максимально возможное количество цветов в палитре изображения*

1.находим количество пикселей, используя для вычисления степени числа 2:

2.$N = 512 \cdot 256 = 2^9 \cdot 2^8 = 2^{17}$

3.объём файла в Кбайтах $64 = 2^6$

4.объём файла в битах $2^6 \cdot 2^{13} = 2^{19}$

5.глубина кодирования (количество битов, выделяемых на 1 пиксель):

6.$2^{19} : 2^{17} = 2^2 = 4$ бита на пиксель

7.максимальное возможное количество цветов $2^4 = 16$

8.Ответ: 16.

## #8

```
# слова по порядку
from itertools import *

c = 1
for i in product('ЩОГБА', repeat=6):
    s = ''.join(i)
    if s == 'ОБЩАГА':
        print(c, s)
    c += 1
```

```
# количество подходящих
from itertools import *

c = 0
for i in permutations('КУСАТЬ', r=5):
    s = ''.join(i)
    if s[0] != 'Ь' and s.count('СУК') == 0:
        c += 1

print(c)
```

## #12

```
s = '>' + 39 * '0' + 5 * '1' + 39 * '2'
while '>1' in s or '>2' in s or '>0' in s:
    if '>1' in s:
        s = s.replace('>1', '22>', 1)
    if '>2' in s:
        s = s.replace('>2', '2>', 1)
    if '>0' in s:
        s = s.replace('>0', '1>', 1)
s = s.replace('>', '0')
su = sum([int(x) for x in s])
print(su)
```

## #14 (перевод в 10 сс)

```
for x in range(0, 36):
  s = (5 + 4*36 + x*36**2 + 2*36**3 +
36**4 + x + 12345)
  if s % 13 == 0:
    print(s // 13)
```

## #14 (перевод из 10 сс)

```
lst = []
n = 5**2026 + 7*5**1013 + 107 - x
base = 6
while n > 0:
  lst.append(n % base)
  n = n // base
lst = lst[::-1]
```

## #15 (первые три прототипа)

```
def f(A, x, y): # переписываем функцию из
задание
return # функция

for A in range(0, 100):
  if all(f(A, x, y) == True for x in
range(1, 100) for y in range(1, 100)):
    print(A)
```

## #15 (отрезки)

```
from itertools import *

def f(x):
    p = 5 <= x <= 30
    q = 14 <= x <= 23
    a = a1 <= x <= a2
    return (p == q) <= (not a)

ox = [i/4 for i in range(5*4, 31*4)]
lst = []

for a1, a2 in combinations(ox, 2):
    if all(f(x) == 1 for x in ox):
        lst.append(a2 - a1)
print(max(lst))
```

## #16

```
from functools import *

@lru_cache
def f(n):
  if n == 1: return 1
  if n >= 2: return f(n - 1) - 2*g(n-1)

@lru_cache
def g(n):
  if n == 1: return 1
  if n >= 2: return f(n - 1) + g(n-1) + n

print(sum([int(x) for x in str(g(36))]))
```

## #17 (пример задания)

```
f = open('17-1.txt')
lst = [int(x) for x in f]
ans = []
for i in range(1, len(lst) - 1):
  if lst[i] < lst[i - 1] and lst[i] < lst[i+1]:
    ans.append(lst[i])
print(len(ans), max(ans))
```

## #19-21 (одна куча)

```
def f(s, n): # s – колво камней, n – за сколько ходов закончится игра
  if s >= 25: return n % 2 == 0 # если колво камней норм, то спрашиваем у проги, кто победил (наш игрок всегда четный)
  if n == 0: return 0 # если камней еще не норм, а ходы уже закончились, значит скипаем
  h = [f(s + 2, n - 1), f(s * 2, n - 1)] # список всевозможных ходов
  return any(h) if (n - 1) % 2 == 0 else all(h) # если следующий ход наш (четный), то мы имеем право выбирать ход (any / любой), иначе мы должны учитывать все ходы соперника

print([s for s in range(1, 25) if f(s, 2)]) # 19
print([s for s in range(1, 25) if not f(s, 1) and f(s, 3)]) # 20
print([s for s in range(1, 25) if not f(s, 2) and f(s, 4)]) # 21
```

## #19-21 (две кучи)

```
def f(s, a, n):
  if s + a >= 69: return n % 2 == 0
  if n == 0: return 0
  h = [f(s+2, a, n-1), f(s*2, a, n-1), f(s, a+2, n-1), f(s, a*2, n-1)]
  return any(h) if (n - 1) % 2 == 0 else all(h) # если в 19 номере Петя сделал неудачный ход, то меняем all -> any (ДЛЯ 20 и 21 ВОЗВРАЩАЕМ ВСЕ НА СВОИ МЕСТА)

print([s for s in range(1, 60) if f(s, 9, 2)])
print([s for s in range(1, 60) if not f(s, 9, 1) and f(s, 9, 3)])
print([s for s in range(1, 60) if not f(s, 9, 2) and f(s, 9, 4)])
```

## #23

```
def f(a, b):
  if a == b: return 1
  if a > b or a == 10 or a == 15: return 0 # избегаемые числа – 10 и 15
  return f(a + 1, b) + f(a + 2, b) + f(a + 3, b)

print(f(5, 11) * f(11, 18)) # обязательное число – 11
```

## #23 (запоминание прошлого хода)

```
def f(a, b, r): # r – какой ход мы сделали в прошлый раз
  if a == b: return 1
  if a > b: return 0
  if a < b and r == '+1':
    return f(a + 2, b, '+2') + f(a * 2, b, '*2')
  if a < b and r != '+1':
    return f(a + 2, b, '+2') + f(a * 2, b, '*2') + f(a + 1, b, '+1')

print(f(1, 18, ''))
```

## #25 (поиск делителей)


```
def f(n):
  lst = []
  for i in range(1, int(n**0.5) + 1):
    if n % i == 0:
      lst.append(i)
      lst.append(n // i)
  return sorted(set(lst))

for i in range(126849, 126872):
  if len(f(i)) == 4: # сколько делителей надо
    print(f(i)[2:])
```

## #25 (маски)

```
from fnmatch import *

for i in range(0, 10**10, 1017): # ставим шаг, чтобы числа были кратные
  if fnmatch(str(i), '2?5432*1'):
    print(i, i // 1017)
```

## #26 (сисадмин)

```
f = open('26-1.txt')
s, n = map(int, f.readline().split())
a = sorted([int(x) for x in f])

s1 = 0
c = 0
b = 0
for i in range(n):
  if s1 + a[i] <= s:
    s1 += a[i]
    c += 1
    b = a[i]
    a[i] = 0
print(c, 'макс число')
s1 = s1 - b
a = a[::-1]
for i in range(n):
    if a[i] != 0 and s1 + a[i] <= s:
      print(a[i], 'макс файл')
      break
```

## #26 (2023 год ячейки)

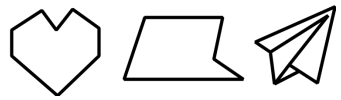
```
f = open('26-3.txt')
k = int(f.readline())
n = int(f.readline())
time = []
for i in range(n):
  a = [int(x) for x in f.readline().split()]
  time.append(a)
time.sort()
fin = [0] * k
c = 0
c1 = 0
for i in range(len(time)):
  for j in range(len(fin)):
    if time[i][0] > fin[j]:
      fin[j] = time[i][1]
      c += 1
      c1 = j + 1
      break
print(c, c1)
```

## #27 (выбрать число из пары так, чтобы все не делилось на 3)

```
f = open('271B.txt')
n = int(f.readline())
s_max = 0
d_min = 10**25
for i in range(n):
  a, b = map(int, f.readline().split())
  s_max += max(a, b)
  if abs(a - b) % 3 != 0:
    d_min = min(d_min, abs(a - b))
if s_max % 3 != 0:
  print(s_max)
else:
  print(s_max - d_min)
```

## #27 (досрок, халява)

```
f = open('272A.txt')
n = int(f.readline())
k = int(f.readline())
lst = [int(x) for x in f]
s, m = 0, 0
for i in range(k, n):
  m = max(m, lst[i - k])
  s = max(s, lst[i] + m)
print(s)
```



## #27 (пробирки)

```
f = open('273_B.txt')
n = int(f.readline())
data = []
for i in range(n):
  km, prob = map(int, f.readline(). split())
  data.append([km, (prob + 35) // 36])

k_do = n * [0]
k_do[0] = data[0][1]
for i in range(1, n):
  k_do[i] = k_do[i - 1] + data[i][1]

s0 = 0
for i in range(n):
  s0 = s0 + abs(data[i][0] - data[0][0]) * data[i][1]

prices = [s0]
for i in range(1, n):
  s0 = s0 + (data[i][0] - data[i-1][0]) * k_do[i-1] \
- (data[i][0] - data[i-1][0]) * (k_do[-1] - k_do[i-1])
  prices.append(s0)

print(min(prices))
```

Желаю всем успехов и халявного варианта!

💜