Всероссийская олимпиада школьников по информатике, 2014-15 уч. год
Первый (школьный) этап, г. Москва
Разбор заданий для 9-11 классов

Каждая задача оценивается в 100 баллов.

Ограничение по времени работы в каждой задаче — 1 секунда.

Задача 1. Шахматная доска

Шахматная доска состоит из $n \times m$ клеток, покрашенных в черный и белый цвет в «шахматном» порядке. При этом клетка в левом нижнем углу доски покрашена в черный цвет. Определите, сколько всего на доске черных клеток.

Программа получает на вход два числа n и m, записанных в отдельных строках. Все числа — натуральные, не превосходящие 30 000.

Программа должна вывести одно целое число — количество черных клеток на доске.

Пример входных и выходных данных

Ввод	**Вывод**
3 4	6

Система оценивания

Решение, правильно работающее только для случаев, когда входные числа не превосходят 10, будет оцениваться в 40 баллов.

Решение

Если на доске — четное число клеток, то черных и белых клеток поровну, поэтому ответом будет $nm/2$. Если же произведение nm нечетно, то черных клеток будет на одну больше, поэтому можно поделить nm на 2 нацело и прибавить к результату 1.

Решения задач будут приведены на языке Python версии 3. В языке Python для целочисленного деления используется операция «//» (аналог операции div в языке Паскаль, функции div в алгоритмическом языке или операции «\» в Бейсике). Для проверки четности используется в языке Python используется операция «%» (аналог операции mod в языках Паскаль и Бейсик и функции mod в алгоритмическом языке).

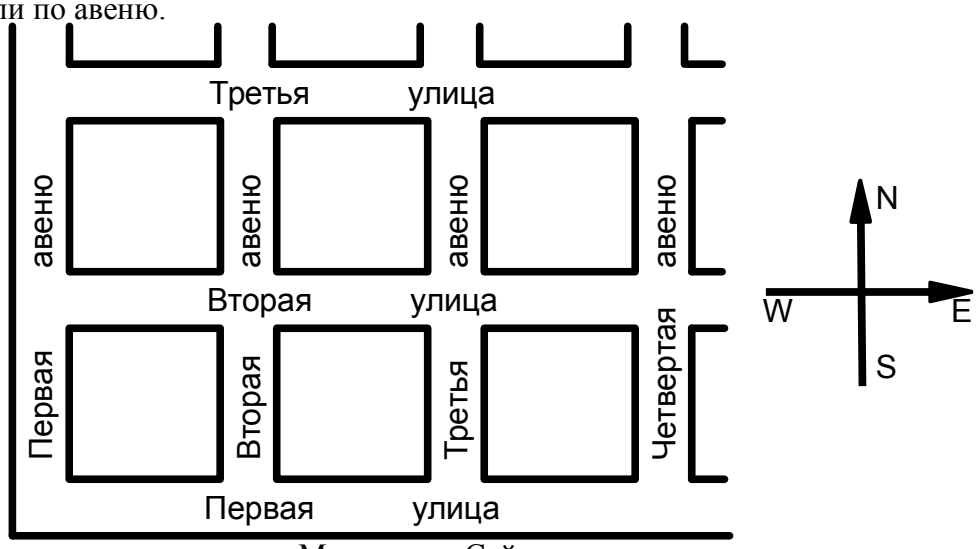
```
n = int(input())
m = int(input())
if n * m % 2 == 0:
    print(n * m // 2)
else:
    print(n * m // 2 + 1)
```

Можно решить эту задачу и без использования инструкции if, если заметить, что результатом является частное от деления nm на 2, округленное вверх:

```
n = int(input())
m = int(input())
print((n * m + 1) // 2)
```

Задача 2. Манхэттен

Кварталы Манхэттена состоят из авеню, направленных с юга на север и улиц, направленных с запада на восток. Все улицы и авеню пронумерованы числами, начиная с 1 подряд (первая улица, вторая улица, третья улица и т. д.). Передвигаться можно только по улицами или по авеню.



Миша впервые попал на Манхэттен. Сейчас он стоит на пересечении авеню номер x_1 и улицы номер y_1. Ему нужно попасть на перекресток авеню номер x_2 и улицы номер y_2. Определите маршрут, который он должен пройти.

Программа получает на вход 4 числа: x_1, y_1, x_2, y_2, записанных в отдельных строках. Все числа — натуральные, не превышают 10^3. Начальное и конечное расположение Миши не совпадают.

Программа должна вывести последовательность из латинских заглавных букв, описывающих маршрут, которому должен следовать Миша. Буква «N» обозначает перемещение на один квартал на север, «S» — на юг, «W» — на запад, «E» — на восток. Программа должна вывести самый короткий из всех возможных маршрутов, если же кратчайших маршрутов существует несколько, то программа должна вывести любой из них (но только один).

Программа может выводить последовательность ходов не в одну строку (как в примере), а каждый символ ответа — в отдельной строке (например, если каждый символ ответа выводится при помощи отдельной команды вывода внутри цикла).

Пример входных и выходных данных

Ввод	Вывод	Примечание
1 3 4 1	EEESS	Миша стоит на пересечении первой авеню и третьей улицы и должен попасть на пересечение четвертой авеню и первой улицы. Ему нужно пройти три квартала на восток и два квартала на юг. Возможны и другие ответы, например, «SSEEE», «ESESE» и прочие. Возможен вывод ответа не в одну строку, а каждый символ ответа — в отдельной строке.

Решение

Если $x_2 > x_1$, то Миша должен переместиться на $x_2 - x_1$ квартала на восток, поэтому программа должна вывести $x_2 - x_1$ символ «E», что можно сделать при помощи цикла. Иначе Миша должен переместиться на $x_1 - x_2$ квартала на запад, поэтому программа должна вывести $x_1 - x_2$ символ «W». Аналогично рассматривается передвижение на север и на юг. Пример решения на языке Python, использующие циклы для вывода последовательности

перемещений:

```
x1 = int(input())
y1 = int(input())
x2 = int(input())
y2 = int(input())
if x2 > x1:
    for i in range(x2 - x1):
        print("E")
else:
    for i in range(x1 - x2):
        print("W")
if y2 > y1:
    for i in range(y2 - y1):
        print("N")
else:
    for i in range(y1 - y2):
        print("S")
```

Приведем еще один пример решения, в котором для повторения одного символа используется операция умножения строки на число, то есть для того, чтобы повторить символ «E» x_2 - x_1 раз нужно выполнить операцию `"E"*(x2-x1)`. При этом если умножить строку на отрицательное число, то получится пустая строка:

```
x1 = int(input())
y1 = int(input())
x2 = int(input())
y2 = int(input())
print("E"*(x2-x1) + "W"*(x1-x2) + "N"*(y2-y1) + "S"*(y1–y2))
```

Задача 3. Пятница, 13-е

Календарь жителей планеты Плюк состоит из N месяцев, каждый месяц состоит ровно из 30 дней, неделя состоит из 7 дней. Особо несчастливыми на планете Плюк считается 13-е число месяца, если оно выпадает на пятницу.

Известно, что Новый год на планете Плюк начался в k-й по счету день недели (1-й день недели — понедельник, 2-й — вторник, 3-й — среда, … , 7-й — воскресенье). Определите, сколько в этом году на планете Плюк будет особо несчастливых пятниц, 13-е.

Программа получает на вход два натуральных числа, записанных в отдельных строках. Первое число — количество месяцев в календаре планеты Плюк N, не превосходящее 10^9. Второе число — номер дня недели, на который приходится первое число первого месяца нового года, может принимать значения от 1 до 7.

Программа должна вывести единственное целое число — количество несчастливых дней в этом году.

Пример входных и выходных данных

Ввод	Вывод	Примечание
12 1	2	На 13-е число будут приходиться пятницы четверого и одиннадцатого месяцев.

Система оценивания

Решение, правильно работающее только для случаев, когда N не превосходит 100, будет оцениваться в 60 баллов.

Решение

Решение на 60 баллов использует простое моделирование. Сначала посчитаем, на какой день недели приходится 13-е число первого месяца, затем будем считать, на какой день недели приходится 13-е число каждого следующего месяца. Если пронумеровать дни недели от 1 до 6, а воскресенье считать днём недели номер 0, то для получения дня недели 13-го числа следующего месяца нужно к текущему номеру дня недели прибавить 30 и взять остаток от деления на 7. Повторяем это в цикле *N* раз и при получении числа 5 увеличиваем ответ на 1. Пример решения на языке Python:

```
N = int(input())
k = int(input())
ans = 0
k = (k + 12) % 7
for i in range(N):
    if k == 5:
        ans += 1
    k = (k + 30) % 7
print(ans)
```

Такое решение будет получать результат «превышено максимальное время работы» при больших значениях *N,* так как за 1 секунду невозможно выполнить цикл в 10^9 шагов.

Для решения задачи на полный балл заметим, что поскольку числа 7 и 30 — взаимно простые, то в последовательности дней недели, на которые выпадают 13-е числа месяца будет цикл длины 7: пятница, воскресенье, вторник, четверг, суббота, понедельник, среда. Поэтому для нахождения ответа достаточно найти первый месяц года, в котором будет пятница 13-е, и к ответу добавить количество оставшихся месяцев в году, деленное на 7 нацело. Пример решения, набирающего полный балл:

```
N = int(input())
k = int(input())
k = (k + 12) % 7
i = 1
ans = 0
while i <= N and k != 5:
    i += 1
    k = (k + 30) % 7
if i > N:
    print(0)
else:
    print(1 + (N - i) // 7)
```

Задача 4. Автомобильные номера

В Российской Федерации на разных видах транспортных средств устанавливаются разные по формату регистрационные знаки («автомобильные номера»). Вот пример нескольких возможных форматов регистрационных знаков.

№	Пример	Описание формата	Тип транспортного средства
1	`Y019KM`	Буква, три цифры, две буквы	Частные транспортные средства
2	`AB179`	Две буквы, три цифры	Общественный транспорт и такси
3	`OH2645`	Две буквы, четыре цифры	Прицепы
4	`3384CT`	Четыре цифры, две буквы	Мотоциклы

В этой задаче «буквой» может быть любая заглавная буква латинского алфавита.

Напишите программу, которые по регистрационному знаку определяет его тип или определяет, что регистрационный знак некорректен.

Программа получает на вход три строки текста, каждая строка содержит один образец регистрационного знака (возможно, некорректный). Каждый образец содержит от 1 до 10 символов, являющихся цифрами и заглавными латинскими буквами (других символов во входных данных быть не может).

Программа должна вывести для каждого образца число, соответствующее типу транспортного средства, как в приведенной таблице, то есть 1 — для частных транспортных средств, 2 — для общественного транспорта, 3 — для прицепов, 4 — для мотоциклов. Если номерной знак некорректен (не подходит ни к одному из указанных типов), то необходимо вывести число 0. Каждое число необходимо выводить в отдельной строке.

Пример входных и выходных данных

Ввод	Вывод

`Y019KM`
`A9999`
`OH2645`
	`1`
	`0`
	`3`

Система оценивания

Решение, правильно работающее только в тех случаях, когда все регистрационные знаки корректны (относятся к одному из четырех типов), будет оцениваться в 40 баллов.

Решение

Это несложная задача на умение обрабатывать текстовые строки и существует множество способов ее решения. Один из самый простых способов решения такой. Заменим в исходной строке все цифры на символ «0», а все буквы — на символ «A». Для этого создадим новую пустую строку T и будем последовательно перебирать символы исходной строки и проверять, является ли он цифрой. Если символ является цифрой, то добавим в конец строки T символ «0», иначе добавим символ «A». В результате если данная строка была регистрационным знаком первого типа, то строка T будет равна «A000AA», если данная строка была второго типа, то T будет равна «AA000», для третьего типа T будет равна «AA000», для четвертого типа — «0000AA». Это можно проверить при помощи четырех условных инструкций, а во всех остальных случаях нужно вывести число 0. Пример решения на языке Python версии 3:

```
for i in range(3):
    S = input()
    T = ""
    for char in T:
        if "0" <= char <= "9":
            T += "0"
        else:
            T += "A"
    if T == "A000AA":
        print(1)
    elif T == "AA000":
        print(2)
    elif T == "AA0000":
        print(3)
    elif T == "0000AA":
        print(4)
    else:
        print(0)
```

Задача 5. Числа-палиндромы

Палиндромом называется строка, которая одинаково читается как слева направо, так и справа налево. Рассмотрим все натуральные числа, запись которых в десятичной системе счисления является палиндромом (при этом запись не начинается с нуля). Например, числа 121 и 1331 являются палиндромами, а число 123 — нет. По данному числу *N* найдите *N*-е в порядке возрастания число-палиндром.

Программа получает на вход одно натуральное число *N*, не превосходящее 100 000.

Программа должна вывести одно натуральное число — *N*-е в порядке возрастания число-палиндром.

Пример входных и выходных данных

Ввод	**Вывод**
20	111

Система оценивания

Решение, правильно работающее только в тех случаях, когда *N* не превосходит 100, будет оцениваться в 20 баллов.

Решение, правильно работающее только в тех случаях, когда *N* не превосходит 1000, будет оцениваться в 40 баллов.

Решение

На полный балл эта задача довольно сложна, но частничное решение, которое будет набирать 40 баллов, написать довольно несложно. Для этого достаточно подряд перебирать все числа, начиная с 1, и проверять, является ли число палиндромом. Когда будет обнаружено N-е по счету число-палиндром, необходимо вывести ответ.

Для проверки того, является ли число палиндромом, заведем переменную num_reversed, присвоив ей значение 0. Затем рассмотрим текущее число, будем получать его цифры, начиная с конца по одной, для чего необходимо в цикле делить число нацело на 10. Последней цифрой числа при этом будет остаток от деления числа на 10. Если внутри этого

цикла умножать число num_reversed на 10 и добавлять к нему последнюю цифру числа, то в итоге в переменной num_reversed получится значение исходного числа, записанное в обратном порядке. Вот пример цикла, в результате которого в переменную num_reversed будет записана десятичная запись числа curr в обратном порядке:

```
num_reversed = 0
while num > 0:
    num_reversed = num_reversed * 10 + num % 10
    num //= 10
```

Полностью решение можно записать на языке Python следующим образом:

```
N = int(input())
i = 1
while N > 0:
    num = i
    num_reversed = 0
    while num > 0:
        num_reversed = num_reversed * 10 + num % 10
        num //= 10
    if num_reversed == i:
        N -= 1
    i += 1
print(i - 1)
```

Чуть более короткий вариант этого решения, которое для проверки, является ли число палиндромом, преобразует его десятичную запись в строку, разворачивает эту строку в обратном порядке и проверяет, что две получившиеся строки равны:

```
N = int(input())
i = 1
while N > 0:
    if str(i) == str(i)[::-1]:
        N -= 1
    i += 1
print(i - 1)
```

Для решения на полный балл заметим, что существует 9 чисел-палиндромов длины 1 (это числа 1, 2, 3, …, 9), 9 чисел-палиндромов длины 2 (11, 22, 33, …, 99), 90 чисел-палиндромов длины 3 (101, 111, 121, …., 999), 90 чисел-палиндромов длины 4 (1001, 1111, 1221, …, 9999), 900 чисел-палиндромов длины 5, 900 чисел-палиндромов длины 6, 9000 чисел-палиндромов длины 7, 9000 чисел-палиндромов длины 8, 90000 чисел-палиндромов длины 9. Видно, что если исходное число не превосходит 100000, то ответ будет содержать не более 9 цифр.

Поэтому сначала определим длину ответа и порядковый номер данного числа-палиндрома среди всех чисел-палиндромов такой же длины, затем определим и само число-палиндром.

Вот пример полного решения на языке Python:

```
N = int(input())
currlen = 1
currcount = 9
while N > currcount:
    N -= currcount
    currlen += 1
    currcount = 9 * 10 ** ((currlen - 1) // 2)

leftsize = (currlen + 1) // 2
rightsize = currlen - leftsize

leftpart = str(10 ** (leftsize - 1) + N - 1)
rightpart = leftpart[::-1]
if rightsize < leftsize:
    rightpart = rightpart[1:]
print(leftpart + rightpart)
```

Возможен и другой вариант решения, в котором вместо циклов все возможные случаи разбираются при помощи отдельный условий. Поскольку ответ содержит не более 9 цифр, то можно разобрать только 9 случаев.

```
N = int(input())
if N <= 9:
    print(N)
elif N <= 9+9:
    s = str(N-9)
    print(s + s)
elif N <= 9+9+90:
    s = str(N-9-9+10-1)
    print(s + s[-2::-1])
elif N <= 9+9+90+90:
    s = str(N-9-9-90+10-1)
    print(s + s[::-1])
elif N <= 9+9+90+90+900:
    s = str(N-9-9-90-90+100-1)
    print(s + s[-2::-1])
elif N <= 9+9+90+90+900+900:
    s = str(N-9-9-90-90-900+100-1)
    print(s + s[::-1])
elif N <= 9+9+90+90+900+900+9000:
    s = str(N-9-9-90-90-900-900+1000-1)
    print(s + s[-2::-1])
elif N <= 9+9+90+90+900+900+9000+9000:
    s = str(N-9-9-90-90-900-900-9000+1000-1)
    print(s + s[::-1])
else:
    s = str(N-9-9-90-90-900-900-9000-9000+10000-1)
    print(s + s[-2::-1])
```