Задача 1

Выведите лесенку "наоборот": по данному натуральному числу n напечатайте nстрок, в первой из которых записано n раз число n, во второй — n−1 раз число n−1, …, в последней — один раз число 1.

ВХОДНЫЕ ДАННЫЕ

На вход программе подается одно натуральное число n, не превосходящее 100.

ВЫХОДНЫЕ ДАННЫЕ

Выведите n строк, описанных в условии. Числа в одной строке должны быть разделены пробелами.

ПРИМЕРЫ

|  |  |
| --- | --- |
| **ВВОД** | **ВЫВОД** |
| 5 | 5 5 5 5 5  4 4 4 4  3 3 3  2 2  1 |

Начало формы

Задача 2.

Лёгкая

**Трехзначные числа**

Дано натуральное число n. Выведите в порядке возрастания все трехзначные числа, сумма цифр которых равна n.

ВХОДНЫЕ ДАННЫЕ

Вводится натуральное число n.

ВЫХОДНЫЕ ДАННЫЕ

Выведите ответ на задачу.

ПРИМЕР

|  |  |
| --- | --- |
| **ВВОД** | **ВЫВОД** |
| 3 | 102 111 120 201 210 300 |

Начало формы

Задача 3.

Лёгкая

**Четырехзначные палиндромы**

Даны два четырёхзначных числа A и B. Выведите все четырёхзначные числа на отрезке от A до B, запись которых является палиндромом.

ВХОДНЫЕ ДАННЫЕ

Вводятся два целых числа A и B.

ВЫХОДНЫЕ ДАННЫЕ

Выведите ответ на задачу.

ПРИМЕР

|  |  |
| --- | --- |
| **ВВОД** | **ВЫВОД** |
| 1600 2100 | 1661 1771 1881 1991 2002 |

Начало формы

Задача 4

Лёгкая

**Минимальный делитель**

Дано целое число, не меньшее 2. Выведите его наименьший простой делитель.

Начало формы

Задача 5

Лёгкая

**Утренняя пробежка**

В первый день спортсмен пробежал x километров, а затем он каждый день увеличивал пробег на 10% от предыдущего значения. По данному числу y определите номер дня, на который пробег спортсмена составит не менее y километров.

ВХОДНЫЕ ДАННЫЕ

Программа получает на вход действительные числа x и y.

ВЫХОДНЫЕ ДАННЫЕ

Программа должна вывести одно натуральное число.

ПРИМЕРЫ

|  |  |
| --- | --- |
| **ВВОД** | **ВЫВОД** |
| 10 20 | 9 |

Начало формы

Задача 6

Вводная

**Длина последовательности**

Программа получает на вход последовательность целых неотрицательных чисел, каждое число записано в отдельной строке. Последовательность завершается числом 0, при считывании которого программа должна закончить свою работу и вывести количество членов последовательности (не считая завершающего числа 0).

Числа, следующие за числом 0, считывать не нужно.

ВХОДНЫЕ ДАННЫЕ

Вводится последовательность целых чисел, заканчивающаяся числом 0.

ВЫХОДНЫЕ ДАННЫЕ

Выведите ответ на задачу.

ПРИМЕРЫ

|  |  |
| --- | --- |
| **ВВОД** | **ВЫВОД** |
| 1 7 9 0 5 | 3 |

Начало формы

Задача 7

Лёгкая

**Минимальная и максимальная цифры**

Найдите наибольшую и наименьшую цифры заданного натурального числа.

ВЫХОДНЫЕ ДАННЫЕ

Программа получает на вход одно натуральное число N.

ВЫХОДНЫЕ ДАННЫЕ

Программа должна вывести наименьшую и наибольшую цифры этого числа через пробел.

ПРИМЕРЫ

|  |  |
| --- | --- |
| **ВВОД** | **ВЫВОД** |
| 1234 | 1 4 |

Начало формы

 Задача 8

Дорожка замощена плитками в один ряд, плитки пронумерованы числами от 1 до 1000. На плитках с номерами A, B и C (ABC) сидят три кузнечика, которые играют в чехарду по следующим правилам:

1. На одной плитке может находиться только один кузнечик.
2. За один ход один из двух крайних кузнечиков (то есть с плитки A или с плитки C) может перепрыгнуть через среднего кузнечика (плитка B) и встать на плитку, которая находится ровно посередине между двумя оставшимися кузнечиками (то есть между B и C или A и B соответственно). Если между двумя оставшимися кузнечиками находится чётное число плиток, то он может выбрать любую из двух центральных плиток.

Например, если кузнечики первоначально сидели на плитках номер 1, 5, 10, то первым ходом кузнечик с плитки номер 10 может перепрыгнуть на плитку номер 3 (она находится посередине между 1 и 5), или кузнечик с плитки номер 1 может перепрыгнуть на плитку номер 7 или 8 (эти две плитки находятся посередине между плитками 5 и 10).

Даны три числа: A, B, C. Определите, какое наибольшее число ходов может продолжаться игра.

ВХОДНЫЕ ДАННЫЕ

Программа получает на вход три целых числа A, B и C (1⩽A<B<C⩽1000), записанных в отдельных строках.

ВЫХОДНЫЕ ДАННЫЕ

Выведите одно число — наибольшее количество ходов, которое может продолжаться игра.

ПРИМЕРЫ

|  |  |
| --- | --- |
| **ВВОД** | **ВЫВОД** |
| 1 4 6 | 2 |

Начало формы

Решение задачи №1

Пронумеруем строки лесенки от 1 до n сверху вниз. Тогда в i-ой строке нужно напечатать только цифры i и сделать это n−i+1 раз?

##Python

n = int(input())   
  
for i in range(n, 0, -1):   
   print("{} ".format(i)\*i)

##Pascal

var

  n, i, j: longint;

begin

  readln(n);

  for i := 1 to n do

  begin

    for j := 1 to n - i + 1 do

      write(n - i + 1, ' ');

    writeln;

  end;

end.

Решение задачи №2

Переберем все трехзначные числа, для каждого числа подсчитаем сумму цифр:

n = int(input())  
for i in range(100, 1000):  
    x = i // 100  
    y = i // 10 % 10  
    z = i % 10  
    if x + y + z == n:  
        print(i)

Решение задачи №3

Переберем все числа от A до B включительно, для каждого числа сравним его первую цифру с последней, а вторую с предпоследней. Если обе пары равны, то печатаем число.

A = int(input())  
B = int(input())  
for i in range(A, B + 1):  
    x = i // 1000  
    y = i // 100 % 10  
    z = i // 10 % 10  
    t = i % 10  
    if x == t and y == z:  
        print(i)

Программу можно написать короче, если не разделять числа на цифры, а рассмотреть число как строку и сравнить его с самим собой в перевернутом виде:

A = int(input())  
B = int(input())  
for i in range(A, B + 1):  
    s = str(i)  
    if s == s[::-1]:  
        print(i)

Решение задачи №4

n = int(input())  
d = 2  
while n % d != 0:  
    d += 1  
print(d)

Решение задачи №5

Напишем ровно то, что описывается в условии:

x = float(input())  
y = float(input())  
days = 1  
while x < y:  
    x += 0.1 \* x  
    days += 1  
print(days)

Решение задачи№6

РЕШЕНИЕ НА ЯЗЫКЕ PYTHON:

ans = 0  
while int(input()) != 0:  
    ans += 1  
print(ans)

Решение задачи №7

Заведем две переменные для хранения текущей максимальной и минимальной цифры числа. Затем, пока число не равно нулю, будем отрезать от него последнюю цифру и обновлять значения переменных, в которых хранятся минимальная и максимальная цифры.

РЕШЕНИЕ НА ЯЗЫКЕ C

    int n;  
    scanf("%d", &n);  
    int min\_d = 9;  
    int max\_d = 0;  
    while (n > 0)  
    {  
        if (n % 10 < min\_d)  
            min\_d = n % 10;  
        if (n % 10 > max\_d)  
            max\_d = n % 10;  
        n /= 10;  
    }  
    printf("%d %d\n", min\_d, max\_d);

РЕШЕНИЕ НА ЯЗЫКЕ ПАСКАЛЬ

var  
  n, max\_d, min\_d: longint;  
begin  
  readln(n);  
  min\_d := 9;  
  max\_d := 0;  
  while n > 0 do   
  begin  
    if n mod 10 > max\_d then  
      max\_d := n mod 10;  
    if n mod 10 < min\_d then  
      min\_d := n mod 10;    
    n := n div 10;    
  end;  
  writeln(min\_d, ' ', max\_d);  
end.

 РЕШЕНИЕ НА ЯЗЫКЕ PYTHON

n = int(input())  
min\_d = 9  
max\_d = 0  
while n > 0:  
   if n % 10 > max\_d:  
       max\_d = n % 10  
   if n % 10 < min\_d:  
       min\_d = n % 10      
   n //= 10  
print(min\_d, max\_d)

Решение задачи №8

Эта задача решается при помощи так называемого "жадного" алгоритма: кузнечики должны прыгать так, чтобы расстояние между ними оставалось максимально возможным. На первом шаге расстояния между соседними кузнечиками равны B—A и C—B.  Соответственно, если B−A>C—B, то должен прыгать третий кузнечик, иначе должен прыгать первый кузнечик.

Таким образом, на первом шаге нужно выбрать наибольший из двух отрезков длины B−A и C—B. Запишем длину этого отрезка в переменную d. На каждом шаге третий кузнечик прыгает в середину отрезка длины d, поэтому значение сокращаетсявдваразаПриэтомеслиdсокращаетсявдвараза.Приэтом,еслиdчетноетоонопростоделитсянааеслинечетноетоотрезокделитсянадвечастидлиныкоторыхотличаютсянаинеобходимовыбратьбольшуюизэтихчастейТоестьнужноподелить—четное,тоонопростоделитсяна2,аеслинечетное,тоотрезокделитсянадвечасти,длиныкоторыхотличаютсяна1,инеобходимовыбратьбольшуюизэтихчастей.Тоестьнужноподелитьd$ на 2 с округлением вверх. Для деления на 2 с округлением вниз в языке Python используется оператор //, чтобы поделить на 2 с округлением вверх, можно воспользоваться таким присваиванием: d = (d + 1) // 2. Будем делить d на 2 с округлением вверх, подсчитывая количество выполненных делений. Цикл закончится при d = 1, то есть, когда кузнечики после очередного прыжка окажутся на соседних клетках.

КОД НА ЯЗЫКЕ PYTHON:

a = int(input())  
b = int(input())  
c = int(input())  
d = max(b - a, c - b)  
ans = 0  
while d > 1:  
    ans += 1  
    d = (d + 1) // 2  
print(ans)

Конец формы

Конец формы

Конец формы

Конец формы

Конец формы

Конец формы

Конец формы

Конец формы