

Начальное значение суммы *s* считаем равным 0. В цикле просматриваем значения всех чисел, обозначатощих возраст участников группы. Если возраст меньше 11, то значение суммы s не изменяется, и мы переходим к просмотру следующего числа. Если возраст меньше 19, значение суммы s увеличиваем на 50 (py6.), в противном случае увеличиваем *s* на 100 (py6.).

for i := 1 to п do begiп read(d);

if *d I I* then continue

if *d i 9* then Inc(s, 50) else Inc(s, 60);

# end;

write(s),’



В десятичной записи ліобого палиндрома равноудаленные от концов цифры совпадаіот. Из ii цифр заданного набора можно будет составить палиндром только в том случае, когда каждая цифра от 0 до 9 входит в этот набор *чёт ное* количество раз (возможно, ни разу), или же *ровно одна* цифра входит нечётное число раз, а все остальные — чётное число. Например, последнему условиіо удовлетворяют числа 1112002 и 2200550, из которых можно составить палиндромы 2011102 и 2050502. В записи этих палиндромов цифра, входящая *нечётное* число раз, находится ровно посередине. Исключение составляіот числа, которые содержат чётное число нулей и ровно одну ненулевую цифру. К ним относится, в частности, число 10000, из которого составить палиндром невозможно. Реализация этого алгоритма требует аккуратной проверки всех этих условий, а также умения работать с отдельными цифрами-символами исходного

набора. Сложность этого алгоритма — О(п) операций.

(Аптор задачи и решений /tnn6ey *М.И. )*

ПEPBOE РЕШЕНИЕ. Воспользуемся жадным алгоритмом. Из данного числа х вычтем нопболоtиий возможный факториал п! и к полученной разности применим те же рассуждения. Другими словами, из полученной разности снова вычтем наибольший факториал, и так далее.

Количество вычитаемых факториалов и будет искомым ответом в задаче. Реализация этого алгоритма несложная. Корректность жадного алгоритма следует из неравенства

1 1! + 2-2! + . . . + (u — 1) (п — 1)! < п! .

# Действительно, если для заданного числа т выполняется неравенство ii! х < (ii + 1)!, то представление х в виде суммы наименьшего количества факториалов обязательно содержит слагаемое п!. В противном случае, как следует из приведённого неравенства, сумма «остальных» факториалов будет меньше х. (Если факториал k! входит с коэффициентом с, > k, то слагаемое с, k! = (k + 1)! + (с, — k — 1)a! можно заменить суммой меньшего числа слагаемых.)

BTOPOE РЕШЕНиЕ основано на применении *факіпоритньной сисіпемьt счисления.* В этой системе счисления основанием служит последовательность натуральных чисел: 1!, 2!, ... , *h!.* В факториальной системе натуральное число х представляется единственным образом в виде

Х Су 1! ’+ C2

2! + . . . + с, п!, где 0 с, L.

# Из этого представления следует, что ноиженьtпее количество факториалов, при сложении которых получается заданное х, равно сумме «цифр» с, в записи числа х в факториальной системе счисления, то есть равно

*S* — Cj + C2 +- • . + Сц .

Осталось научиться переводить число в факториальнуіо систему. Оказывается, для этого не обязательно вьшислять факториалы натуральных чисел. Действительно, каждое из чисел L! делится на все натуральные числа от 1 до L, поэтому слагаемые вида с, L! делятся на 2 при

лтобом k й 2. Значит, число с, равно остатку от деления *х* на 2. Разделив нацело *х* на 2, то есть

отбрасывая слагаемое с-,

1!, применим те же рассуждения к числу *х* div 2. Теперь число cc равно

остатку от деления х div 2 на 3. Этот процесс продолжаем до тех пор, пока частное от деления на очередное число fi не станет уже равным нулто. Ниже приведен фрагмент кода на языке Delphi, реализующего этот алгоритм.

s := 0; k := 2;

while (п > 0) do begin s := s + х mod k;

х := *х* div k;

# inc(k); end; write(s);

(Логонда coc=i на студен=іами КФУ, ав=іop зауаии и решений *Киндер М.И.)*

РЕШЕНИЕ ПОДЗАДАЧИ 1. Для «небольших» значений *h* и nt (2 й h й 3, 1 й m 100) все требуемые наборы можно найти несложным перебором. Такое решение проходит тесты на 30 баллов.

РЕШЕНИЕ ПОДЗАДАЧИ 2. Поскольку требуется подсчитать только количество требуемых наборов чисел, можно применить комбинаторные рассуждения. Пусть /п — наименьшее общее красное всех чисел нашего набора, и пусть m — р 'p 22 .. *ра' —* разложение nt на простые множители.

# Тогда условито задачи удовлетворятот все наборы чисел п, =

o[i,2]

*or* "‘"' › 1 *i* й k, у которых

неотрицательные показатели o[i,s] степеней простых чисел *р,* не превосходят о, и для которых выполняются условия

max(cl[1,s], п[2,s], . . . , ‹i[L,s]) = ‹i, ,

для всех 1 й s й *г.* Подсчитаем число наборов по каждому показателто о отдельно. Количество наборов, которые удовлетворятот условиям 0 й о, й п для всех i от 1 до *h,* равно

# В самом деле, число о, может принимать о + 1 вначений это все неотрицательные числа от 0 до о, и по правилу произведения количество таких наборов равно (п + 1)a. Осталось исклточить из этого множества наборы, для которых max(n , о , ... , п,) < о. Для таких наборов 0 й о, й о — 1 для всех i от 1 до *h,* поэтому их количество равно п\*. Значит, количество требуемых наборов равно (п + 1)a — о\*. Теперь по правилу произведения число наборов по всех показателям о, равно:

*L(т, k) —— [(а, + )k gk]*

Например, для m = 10 = 2' 5' и L = 2 получаем

*L(1* О, 2) = [(1+ })2 }2] . 1 + 1)2 }2] ()

упорядоченных наборов из двух чисел, у которых HOW равен 10. Алгоритмическая сложность такого решения — *О(Fact(т) + k).Taкoe* решение проходит тесты ещё на 35 баллов.

РЕШЕНИЕ ПОДЗАДАЧИ 3. Для чисел k из диапазона [1; 10' ] для быстрого вычисления (л + 1)' и о\* используем бинарный алгоритм возведения в степень. Алгоритмическая сложность такого решения — *О(Fact(т) +* log£).

*Председатель жюри М.И. Киидер*