М. С. Долинский
Гомельский государственный университет имени Франциска Скорины, Беларусь

РЕШЕНИЕ ЗАДАЧ РЕКУРСИВНОЙ ГЕНЕРАЦИЕЙ ЧИСЕЛ

Аннотация
В статье описана методика изучения темы «Решение задач рекурсивной генерацией чисел» при подготовке школьников к олимпиадам по информатике. Изучение основано на последовательном решении усложняющихся задач. Для каждой задачи приводятся следующие материалы: условие задачи, идея решения с предложением придумать самостоятельно реализацию, решение на языке программирования Pascal. Серьезной технической основой является разработанная под управлением автора инструментальная система дистанционного обучения (http://dl.gsu.by), которая позволяет: предложить ученику условие задачи; отправить решение на проверку; получить от системы вердикт — правильное или неправильное решение; для неправильных решений указывается номер теста, на котором решение не прошло. Ученик может взять тест (входные и выходные данные), на котором не прошло его решение, разобраться, в чем ошибка в его программе, исправить ее и послать решение повторно. Кроме того, для каждой задачи есть ссылка по ней на тему в форуме, где можно задать вопрос по решению этой задачи и/или почитать ответ, если вопросы уже задавались ранее.
Ключевые слова: рекурсия, олимпиады по информатике, инструментальная система дистанционного обучения.
DOI: 10.32517/2221-1993-2021-20-1-46-51

Введение

Подготовка школьников к олимпиадам по информатике и программированию требует наличия у учащихся значительного объема теоретических знаний и практических навыков решения задач. Поэтому в мире компьютерной книжной и журнальной литературы регулярно появляются книги и статьи как по общей теории алгоритмов [3, 5, 9, 10, 13, 15–18], так и на отдельные актуальные темы: динамическое программирование [12, 21, 25], графы [11, 19, 20, 21, 22–24], хеши [26], рекурсия [1, 2, 4, 14].

Автор вносит свою лепту в разработку темы рекурсии [6–8] уникальным подходом, связанным с ранним началом обучения. Как следствие, уже в IV—VI классах появляются ученики, которым приходится объяснять такие темы, как рекурсия. Поэтому приходится модифицировать изложение в сторону более простого и наглядного объяснения и более медленного продвижения по учебному материалу, явно выделяя и обозначая все этапы этого продвижения. Введение в решение задач с помощью рекурсии изложено в работе [6]. Решение задач с помощью рекурсивной генерации таких комбинаторных объектов, как множество всех подмножеств, сочетания, перестановки, перестановки с повторениями, размещения, описано в работе [7]. В работе [8] описано решение рекурсивных задач «по определению».

В настоящей статье предлагается материал по решению задач рекурсивной генерацией чисел, включающий условия задач и примеры их решения.

Проверка решений осуществляется автоматически на сайте DL.GSU.BY. Вдумчивым читателям предлагается после чтения условия задачи попытаться решить ее самостоятельно, а если решить не удалось, аналогичную попытку рекомендуется предпринять перед чтением текста решения каждой задачи, попытавшись вначале воспользоваться приведенными предварительно рекомендациями по алгоритму решения задачи.

Задача 1. Идеальные числа (Гомельская школьная олимпиада 2011 года)

По мнению Андрея, идеальным является число, для которого модуль разности сумм цифр на четных и на нечетных позициях без остатка делится на M. Задано число N, необходимо определить, сколько чисел длины N (без лидирующих нулей) являются идеальными.

Контактная информация

Долинский Михаил Семенович, канд. тех. наук, доцент, доцент кафедры математических проблем управления и информатики, Гомельский государственный университет имени Франциска Скорины, Беларусь; *адрес:* 246000, Республика Беларусь, г. Гомель, ул. Советская, д. 104; *e-mail:* dolinsky@gsu.by

M. S. Dolinsky
Francisk Skorina Gomel State University, Belarus

SOLUTION OF TASKS USING RECURSIVE NUMBERS GENERATION

Abstract
The article describes the methodology to solve tasks of Olympiads in informatics using recursive numbers generation. The study is based on the sequential solution of increasingly complex tasks. For each task, the following materials are given: the condition of the task, the idea of the solution with a proposal to come up with an independent implementation, the solution in the Pascal programming language. Distance learning system DL.GSU.BY is the effective technical base for teaching. It allows offer the student a condition of the task; send the solution for review; get a verdict from the system — a correct or incorrect solution; for incorrect solutions, the number of the test on which the solution did not pass is indicated. A student can take a test (input and output data), on which his solution did not pass, figure out what the error is in his program, correct and send the solution again. In addition, for each task there is a link on it to the topic in the forum, where you can ask a question on solving this problem and / or read the answer if the questions have already been asked before.
Keywords: recursion, Olympiads in informatics, distance learning tools.

Формат ввода:
Два целых числа, разделенных пробелом: *N M*.

Формат вывода:
Одно число — количество идеальных чисел длины *N*.

Ограничения:
$1 \le N \le 8$,
$1 \le M \le 20$.

Пример ввода	*Пример вывода*
3 3	300

Идея решения.
Идея состоит в том, чтобы рекурсивно конструировать числа из цифр и проверять выполнение свойства.

Полный текст решения.

```
var
 N,M,ans : longint;

procedure Rec(N,S:longint);
 var
  y : longint;
 begin
  if (N=0) and ((Abs(S) mod M)=0) then inc(ans)
    else
  if N=0 then exit
    else
  if N=1 then for y:=1 to 9 do Rec(N-1,S+y)
    else
  if N>1 then for y:=0 to 9 do
   if Odd(N)
    then Rec(N-1,S+y)
    else Rec(N-1,S-y);
end;

begin
 readln(N,M);
 ans:=0;
 Rec(N,0);
 writeln(Ans);
end.
```

Задача 2. Интересные числа (Гомельская городская олимпиада 2012 года)

Пусть задано *N*-значное число, где *N* — четное. Разделим его на две равные части по *N*/2 цифр. Левая часть числа называется интересной, если она нацело делится на 3 и сумма цифр в ней не менее *M*. Правая часть числа называется интересной, если она нацело делится на 9 и сумма цифр в ней не менее *M*. Заданное число называется интересным, если обе половины его интересные. Например, при $N = 2$ и $M = 5$ интересные числа — 69, 99, неинтересные — 33, 30, 24 и т. д.

Ваша задача — подсчитать общее количество интересных *N*-значных чисел без лидирующих нулей.

Формат ввода:
Два целых числа, разделенных пробелом: *N M*.

Здесь: *N* — число знаков в числе, *M* — критерий интересности.

Ограничения:
Все числа — целые;
$2 \le N \le 14$;
N — четное;
$1 \le M \le 100$.

Формат вывода:
Одно число *Ans* — количество интересных *N*-значных чисел.

Пример ввода	*Пример вывода*
2 5	2

Идея решения.
Рекурсивно считаем количество левых интересных половин *AnsL* и количество правых интересных половин *AnsR*. Ответ на задачу — произведение этих чисел.

В рекурсии для левых половин генерируем числа без значащих нулей, для правых половин — с ними.

Поскольку произведение может превышать longint, объявляем *AnsL* и *AnsR* как int64.

Полный текст решения.

```
var
 N,M : longint;
 AnsL,AnsR : int64;

procedure RecR(N,S:longint);
 var
  i : longint;
 begin
  if N=0
   then begin
    if (S>=M) and (S mod 9 = 0)
     then inc(AnsR);
      exit;
   end;
  for i:=0 to 9 do RecR(N-1,S+i);
 end;

procedure RecL(N,S:longint);
 var
  i : longint;
 begin
  if N=0
   then begin
    if (S>=M) and (S mod 3 = 0)
     then inc(AnsL);
    exit;
   end;
  if N=1
   then for i:=1 to 9 do RecL(N-1,S+i)
   else for i:=0 to 9 do RecL(N-1,S+i);
end;

begin
  readln(N,M);
  AnsL:=0; RecL(N div 2,0);
  AnsR:=0; RecR(N div 2,0);
  writeln(AnsL*AnsR);
end.
```

Задача 3. Пароль (Гомельская городская олимпиада 2010 года)

Сережа забыл пароль к своей электронной почте. Помнит он только то, что сам пароль состоял из L цифр. Также он помнит, что сумма элементов, стоящих на позициях $1+3\times i$ (подразумеваются позиции 1, 4, 7, 10, ...), равна сумме элементов, стоящих на позициях $2+3\times j$ (подразумеваются позиции 2, 5, 8, ...), и сумме элементов, стоящих на позициях $3\times d$ (подразумеваются позиции 3, 6, 9, ...).

Ваша задача — помочь определить Сереже, какое количество паролей длины L соответствует данным требованиям.

Формат ввода:
Одно число — L.

Формат вывода:
Одно число — ответ на задачу.

Ограничения:
$3 \le L \le 14$.

Пример ввода	*Пример вывода*
3	10

Идея решения.

Если имеем всего L позиций, значит, у нас три группы цифр, в каждой из которых не более чем $L3 := 1 + (L \text{ div } 3)$ цифр. Поскольку $L \le 14$, мы можем предпросчитать массив $K[0..5, 0..45]$ (цифр в группе может быть не более пяти, и сумма может быть не более 45 (5×9)).

$K[i,j]$ — количество способов, которыми можно составить сумму j, использовав ровно i цифр.

Тогда ответ — сумма по всем возможным суммам произведений количеств способов составить такую сумму в первой, второй и третьей группах (соответственно содержащих $L1$, $L2$ и $L0$ чисел).

Массив $K[i, j]$ просчитывается рекурсивно, генерацией всех возможных комбинаций цифр на не более чем $L3$ позициях.

Полный текст решения.

```
var
 L,i,L1,L2,L0,L3 : longint;
 Ans : int64;

 K : array[0..6,0..45] of int64;

procedure Rec(N,S:longint);
 var
  i : longint;
 begin
  if N>L3 then exit;
  inc(K[N,S]);
  for i:=0 to 9 do Rec(N+1,s+i);
end;

procedure Split(L:longint; var
L1,L2,L0:longint);
 begin
  L0:=L div 3;
  L2:=1 + (L-1) div 3;
  L1:=L-L0-L2;
end;

begin
 readln(L);
 L3:=1 + (L div 3);
 for i:=0 to 6 do
  for j:=0 to 45 do K[i,j]:=0;
 Rec(0,0);
 Split(L,L1,L2,L0);
 Ans:=0;
 for i:=0 to 45 do
  inc(Ans,K[L1,i]*K[L2,i]*K[L0,i]);
 writeln(Ans);
end.
```

Задача 4. Сумма делителей (7-я интернет-олимпиада 2011 года, Санкт-Петербург)

Пусть x — натуральное число. Назовем y его делителем, если $1 \le y \le x$ и остаток от деления x на y равен нулю.

Задано число x. Найдите сумму его делителей.

Формат ввода:
Первая строка входного файла содержит целое число x ($1 \le x \le 10^{18}$). Все простые делители числа x не превосходят тысячу.

Формат вывода:
В выходной файл выведите ответ на задачу.

Пример ввода	*Пример вывода*
12	28
239	240

Идея решения.

Строим все простые делители числа x (можно непосредственно, поскольку по условию все они не больше 1000). Пусть kp — их количество, а сами они содержатся в массиве $p[i]$, где i меняется от 1 до kp.

Далее рекурсивно строим (и прибавляем к ответу *ans*) все возможные делители числа x, сконструированные как произведения вида:

$$p[1]\hat{}i1*p[2]\hat{}i2*...p[kp]\hat{}ikp,$$

где $i1$, $i2$, ..., ikp — числа от 0 до максимально возможного значения, оставляющего произведение делителем числа x.

Рассмотрим подробнее реализацию.

Процедура *BiuldP* строит массив $p[i]$ всех простых делителей числа x, kp — их получившееся количество.

```
procedure BuildP;
 var
  i,j : longint;
 begin
  kp:=0;
  for i:=2 to 1000 do
```

```
   if (x mod i)=0
    then begin
     j:=1;
     while (j<=kp) and ((i mod p[j])<>0) do inc(j);
     if j>kp
      then begin inc(kp); p[kp]:=i;end;
    end;
end;
```

Сначала мы проверяем все числа от 2 до 1000 включительно — являются ли они делителями числа x ($(x \bmod i) = 0$). Для каждого такого числа затем проверяем, является ли оно простым: делится ли это число i хоть на одно из ранее запомненных в массиве p простых чисел. Если ни на одно не делится, значит, оно тоже простое, и мы заносим его в массив p.

Рекурсивная процедура конструирования всех возможных чисел, составленных произведением простых p[i] в произвольных степенях от 0 до максимально возможной, выглядит так:

```
procedure Rec(r,n:qword);
 var
  i : longint;
 begin
  inc(ans,r);
  for i:=n to kp do
   if (x mod (r*p[i]))=0
    then Rec(r*p[i],i);
end;
```

Здесь:

r — текущее произведение (начинаем с 1);

n — номер делителя, с которого продолжается увеличение произведения (начинаем с 1).

```
Для всех i от n до kp
  Если x делится на r*p[i]
    То вызываем рекурсию с параметрами r*p[i] и i
```

Полный текст решения.

```
var
 p : array[1..1000] of longint;
 x,ans : qword;
 kp : longint;

procedure Rec(r,n:qword);
 var
  i : longint;
 begin
  inc(ans,r);
  for i:=n to kp do
   if (x mod (r*p[i]))=0
    then Rec(r*p[i],i);
end;

procedure BuildP;
 var
  i,j : longint;
 begin
  kp:=0;
  for i:=2 to 1000 do
   if (x mod i)=0
    then begin
     j:=1;
     while (j<=kp) and ((i mod p[j])<>0) do inc(j);
     if j>kp
      then begin inc(kp); p[kp]:=i;end;
    end;
end;

begin
 assign(input,'divsum.in'); reset(input);
 assign(output,'divsum.out'); rewrite(output);
 readln(x);
 BuildP;
 ans:=0;
 Rec(1,1);
 writeln(ans);
 close(input); close(output);
end.
```

Задача 5. Сумма делителей — 2 (4-я интернет-олимпиада 2007 года)

Пусть x — натуральное число. Назовем y его делителем, если $1 \le y \le x$ и остаток от деления x на y равен нулю.

Задано число x. Найдите сумму его делителей, делящихся на каждое из простых чисел, на которое делится x.

Формат ввода:

Первая строка входного файла содержит целое число x ($1 \le x \le 10^{18}$). Все простые делители числа x не превосходят 1000.

Формат вывода:

В выходной файл выведите ответ на задачу.

Пример ввода	*Пример вывода*
12	18
239	239

Основное отличие от предыдущей задачи вызвано отличием в условии: «найдите сумму делителей, делящихся на каждое из простых чисел, на которое делится x».

Идея решения.

Строим все простые делители числа x (можно непосредственно, поскольку по условию все они не больше 1000). Пусть kp — их количество, а сами они содержатся в массиве $p[i]$, где i меняется от 1 до kp.

Далее рекурсивно строим (и прибавляем к ответу *ans*) все возможные делители числа x, сконструированные как произведения вида:

$$p[1]\hat{}i1*p[2]\hat{}i2*...p[kp]\hat{}ikp,$$

где $i1$, $i2$, ..., ikp — числа от 1 (в предыдущей задаче было от 0) до максимально возможного значения, оставляющего произведение делителем числа x.

Рассмотрим подробнее реализацию.

Процедура *BiuldP* строит массив $p[i]$ всех простых делителей числа x, kp — их получившееся количество, M — произведение всех найденных простых делителей числа x.

```
procedure BuildP;
 var
  i,j : longint;
 begin
  kp:=0; M:=1;
  for i:=2 to 1000 do
   if (x mod i)=0
    then begin
     j:=1;
     while (j<=kp) and ((i mod p[j])<>0) do inc(j);
     if j>kp
      then begin inc(kp); p[kp]:=i; M:=M*I; end;
    end;
end;
```

Сначала мы проверяем все числа от 2 до 1000 включительно: являются ли они делителями числа x ((x mod i) = 0). Для каждого такого числа затем проверяем, является ли оно простым: делится ли это число i хоть на одно из ранее запомненных в массиве p простых чисел. Если ни на одно не делится, значит, оно тоже простое, и мы заносим его в массив p. Одновременно считаем M — произведение всех найденных простых делителей числа x.

Рекурсивная процедура конструирования всех возможных чисел, составленных произведением простых p[i] в произвольных степенях от 1 до максимально возможной, выглядит так:

```
procedure Rec(r,n:qword);
 var
  i : longint;
 begin
  inc(ans,r);
  for i:=n to kp do
   if (x mod (r*p[i]))=0
    then Rec(r*p[i],i);
end;
```

Здесь:

r — текущее произведение (начинаем с M);

n — номер делителя, с которого продолжается увеличение произведения (начинаем с 1).

```
Для всех I от n до kp
  Если x делится на r*p[i]
    То вызываем рекурсию с параметрами r*p[i] и i
```

Полный текст решения.

```
var
 p : array[1..1000] of longint;
 x,ans,M : qword;
 kp,i : longint;

procedure Rec(r,n:qword);
 var
  i : longint;
 begin
  inc(ans,r);
  for i:=n to kp do
   if (x mod (r*p[i]))=0
    then Rec(r*p[i],i);
end;

procedure BuildP;
 var
  i,j : longint;
 begin
  kp:=0; M:=1;
  for i:=2 to 1000 do
   if (x mod i)=0
    then begin
     j:=1;
     while (j<=kp) and ((i mod p[j])<>0) do inc(j);
     if j>kp
      then begin inc(kp); p[kp]:=i; M:=M*I; end;
    end;
end;

begin
 assign(input,'divsum2.in'); reset(input);
 assign(output,'divsum2.out'); rewrite(output);
 readln(x);
 BuildP;
 ans:=0;
 Rec(M,1);
 writeln(ans);
 close(input); close(output);
end.
```

Заключение

В данной статье представлена методика изучения темы «Решение задач рекурсивной генерацией чисел» с учащимися V—VIII классов, предлагающая, по мнению автора, наиболее простой способ постепенного осознания механизма рекурсии и способа решения задач с ее помощью. Методика включает в себя последовательность задач в порядке возрастания сложности, снабженных, где необходимо, предварительными общими пояснениями и последующими полными решениями предлагаемых задач. Серьезной технической основой является разработанная под управлением автора инструментальная система дистанционного обучения (http://dl.gsu.by), которая позволяет максимально автоматизировать процесс обучения.

Список использованных источников

1. *Баррон Д.* Рекурсивные методы в программировании. М.: Мир,1974. 79 с.

2. *Бердж В.* Методы рекурсивного программирования. М.: Машиностроение, 1983. 256 с.

3. *Вирт Н.* Алгоритмы и структуры данных. М.: ДМК, 2016. 272 с.

4. *Головешкин В. А., Ульянов В. В.* Теория рекурсии для программистов. М.: Физматлит, 2006. 296 с.

5. *Дасгупта С., Пападимитриу Х., Вазирани У.* Алгоритмы. М.: МЦНМО, 2019. 320 с.

6. *Долинский М. С.* Введение в решение задач с помощью рекурсивных процедур и функций // Информатика в школе. 2016. № 9. С. 49–56.

7. *Долинский М. С.* Генерация комбинаторных объектов с помощью рекурсивных процедур и функций // Информатика в школе. 2019. № 4. С. 59–63.

8. *Долинский М. С.* Решение рекурсивных задач по определению // Информатика в школе. 2020. № 2. С. 60–66.

9. *Златопольский Д. М.* 1400 задач по программированию. М.: ДМК, 2019. 192 с.
10. *Луридас П.* Алгоритмы для начинающих. Теория и практика для разработчика. М.: ЭКСМО, 2018. 608 с.
11. *Рафгарден Т.* Совершенный алгоритм. Графовые алгоритмы и структуры данных. СПб.: Питер, 2020. 256 с.
12. *Рафгарден Т.* Совершенный алгоритм. Жадные алгоритмы и динамическое программирование. СПб.: Питер, 2020. 256 с.
13. *Рафгарден Т.* Совершенный алгоритм. Основы. СПб.: Питер, 2019. 256 с.
14. *Рубио-Санчес М.* Введение в рекурсивное программирование. М.: ДМК Пресс, 2019. 436 с.
15. *Скиена С.* Алгоритмы. Руководство по разработке. СПб.: BHV, 2011. 720 с.
16. *Солтис М.* Введение в анализ алгоритмов. М.: ДМК, 2019. 278 с.
17. *Стивенс Р.* Алгоритмы. Теория и практическое применение. М.: ЭКСМО, 2016. 544 с.
18. *Шень А.* Программирование: теоремы и задачи. М.: МЦНМО, 2017. 320 с.
19. *Do P. T., Pham B. T., Than V. C.* Latest algorithms on particular graph classes // Olympiad in Informatics. 2020. Vol. 14. P. 21–35.
20. *Castro R., Lehmann N., Pérez J., Subercaseaux B.* Wavelet trees for competitive programming // Olympiad in Informatics. 2016. Vol. 10. P. 19–37.
21. *Forisek M.* Towards a better way to teach dynamic programming // Olympiad in Informatics. 2015. Vol. 9. P. 45–55.
22. *Manev K.* Tasks on graphs // Olympiad in Informatics. 2008. Vol. 2. P. 90–104.
23. *Manev K., Nikolov N., Markov M.* Reconstruction of trees using metric properties // Olympiad in Informatics. 2011. Vol. 5. P. 82–91.
24. *Erdősné Németh A.* Teaching graphs for contestants in lower-secondary-school-age // Olympiad in Informatics. 2017. Vol. 11. P. 41–53.
25. *Erdősné Németh A., Zsakó L.* The place of the dynamic programming concept in the progression of contestants' thinking // Olympiad in Informatics. 2016. Vol. 10. P. 61–72.
26. *Pachocki J., Radoszewskij J.* Where to use and how not to use polynomial string hashing // Olympiad in Informatics. 2013. Vol. 7. P. 90–100.

ПРАВИЛА ДЛЯ АВТОРОВ

Уважаемые коллеги!

Статьи для публикации в журналах «Информатика и образование» и «Информатика в школе» должны отправляться в редакцию **только через электронную форму на сайте ИНФО (раздел «Авторам → Отправка статьи»)**:

http://infojournal.ru/authors/send-article/

Обращаем ваше внимание, что для отправки статьи необходимо предварительно зарегистрироваться на сайте ИНФО (или авторизоваться — для зарегистрированных пользователей).

С требованиями к оформлению представляемых для публикации материалов можно ознакомиться на сайте ИНФО в разделе «Авторам»:

http://infojournal.ru/authors/

Обратите внимание: требования к оформлению файла рукописи — **разные** для журналов «Информатика и образование» и «Информатика в школе». При подготовке файла рукописи ориентируйтесь на требования для того журнала, в который вы представляете статью. Если вы представляете рукопись в оба журнала (для публикации в одном из изданий — на усмотрение редакции), при ее оформлении следует руководствоваться требованиями к оформлению рукописи в журнал «Информатика и образование».

Дополнительную информацию можно получить в разделе **«Авторам → Часто задаваемые вопросы»**:

http://infojournal.ru/authors/faq/

а также в редакции ИНФО:

E-mail: readinfo@infojournal.ru

Телефон: (495) 140-19-86