

Незнайка и недосортировка. 7-10 классы

Знайка ввёл в Цветочном городе буквенную систему счисления. Это позиционная система счисления с основанием 26, в которой цифрами служат строчные латинские буквы и только они. В ней используется такая таблица цифр и их значений:

цифра	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	<i>h</i>	<i>i</i>	<i>j</i>	<i>k</i>	<i>l</i>	<i>m</i>	<i>n</i>	<i>o</i>	<i>p</i>	<i>q</i>	<i>r</i>	<i>s</i>	<i>t</i>	<i>u</i>	<i>v</i>	<i>w</i>	<i>x</i>	<i>y</i>	<i>z</i>
значение	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Приведем пример записи числа в буквенной системе счисления: $2025_{10} = 2*26^2 + 25*26^1 + 23*26^0 = czx$.

Все стали пользоваться буквенной системой счисления, как положено, кроме Незнайки. Однажды Знайка заглянул в исписанный Незнайкой листок, схватился за голову и пришёл в ужас. В записи чисел помимо обычных латинских букв Незнайка задействовал как цифры буквы с крышками ($\hat{a} \dots \hat{z}$), буквы с волной ($\tilde{a} \dots \tilde{z}$) и буквы с нижними подчёркиваниями ($\underline{a} \dots \underline{z}$). Знайка обратился к Незнайке за разъяснениями. Выяснилось, что Незнайка модифицировал буквенную систему счисления, разрешив переполнять в ней разряды, т. е. записывать в них цифры со значениями, превышающими 25. При этом основание системы счисления он оставил прежним – 26. Незнайка дополнил таблицу цифр и их значений:

цифра	\hat{a}	\hat{b}	\hat{c}	\hat{d}	\hat{e}	\hat{f}	\hat{g}	\hat{h}	\hat{i}	\hat{j}	\hat{k}	\hat{l}	\hat{m}	\hat{n}	\hat{o}	\hat{p}	\hat{q}	\hat{r}	\hat{s}	\hat{t}	\hat{u}	\hat{v}	\hat{w}	\hat{x}	\hat{y}	\hat{z}
значение	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
цифра	\tilde{a}	\tilde{b}	\tilde{c}	\tilde{d}	\tilde{e}	\tilde{f}	\tilde{g}	\tilde{h}	\tilde{i}	\tilde{j}	\tilde{k}	\tilde{l}	\tilde{m}	\tilde{n}	\tilde{o}	\tilde{p}	\tilde{q}	\tilde{r}	\tilde{s}	\tilde{t}	\tilde{u}	\tilde{v}	\tilde{w}	\tilde{x}	\tilde{y}	\tilde{z}
значение	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77
цифра	\underline{a}	\underline{b}	\underline{c}	\underline{d}	\underline{e}	\underline{f}	\underline{g}	\underline{h}	\underline{i}	\underline{j}	\underline{k}	\underline{l}	\underline{m}	\underline{n}	\underline{o}	\underline{p}	\underline{q}	\underline{r}	\underline{s}	\underline{t}	\underline{u}	\underline{v}	\underline{w}	\underline{x}	\underline{y}	\underline{z}
значение	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103

Знайка сразу понял, что модифицированная система позволяет одно и то же число записать по-разному. Например:

$$2025_{10} = 2*26^2 + 25*26^1 + 23*26^0 = czx = 1*26^2 + 51*26^1 + 23*26^0 = b\hat{z}x = 77*26^1 + 23*26^0 = \tilde{z}x = 76*26^1 + 49*26^0 = \tilde{y}\hat{x} = 75*26^1 + 75*26^0 = \tilde{x}\tilde{x} = 74*26^1 + 101*26^0 = \tilde{w}\underline{x} = 1*26^2 + 48*26^1 + 101*26^0 = b\hat{w}\underline{x} = 2*26^2 + 22*26^1 + 101*26^0 = cw\underline{x}.$$

Указаны некоторые, но не все способы записи числа 2025_{10} .

Знайка решил доказать Незнайке его неправоту. Он предложил Незнайке отсортировать по возрастанию последовательность различных чисел, записанных в модифицированной Незнайкой системе счисления. Знайка рассчитывал на то, что недостатки «переполненной» системы счисления станут очевидны сразу, как только понадобится выполнять сравнения чисел, записанных с её помощью. Но не тут-то было. Незнайка и бровью не повёл, и вскоре Знайка получил результат сортировки. Зная «творческий» подход Незнайки и заподозрив неладное, Знайка взгляделся в результат. И точно, сбывшись его подозрения. Последовательность $A[i]$ оказалась недосортированной. В ней встретилась одна пара элементов $A[i_1]$ и $A[i_2]$ ($i_1 \neq i_2$), стоящих не на своих местах. Последовательность оказалась бы возрастающей лишь после обмена местами $A[i_1]$ и $A[i_2]$: установив прежнего числа $A[i_1]$ в позицию i_2 и прежнего числа $A[i_2]$ в позицию i_1 .

Помогите Незнайке составить программу, находящую в недосортированной по возрастанию последовательности пару элементов, после обмена которых местами последовательность становится возрастающей. Программа считывает десятичное натуральное ненулевое число N , а затем последовательность из N записей чисел в модифицированной Незнайкой системе счисления. Во вводе буква с крышкой задаётся двумя символами, например, \hat{a} означает \hat{a} . Буква с волной также задаётся двумя символами, например, \tilde{b} означает \tilde{b} . Буква с подчёркиванием также задаётся двумя символами, например, \underline{c} означает \underline{c} . Заранее известно, что последовательность является недосортированной по возрастанию (в описанном выше смысле). Программа находит номера двух элементов, которые следует переставить, чтобы последовательность стала возрастающей, и выводит эти номера. Первым выводится меньший номер, вторым – больший. Нумерация элементов последовательности начинается с единицы.

Пример недосортированной по возрастанию последовательности из пяти элементов: `jw_ b d a kx~`. После обмена местами первого и четвертого элементов она становится отсортированной по возрастанию: `a b d jw_ kx~`

Формат ввода: В первой строке вводится натуральное число N в десятичной записи: $2 \leq N \leq 1000$. В последующих N строках содержатся записи различных неотрицательных чисел $A[i]$ ($i = 1..N, A[i] \neq A[j], i \neq j$) в системе счисления, модифицированной Незнайкой. В каждой записи используются только строчные латинские буквы, крышка `^`, волна `~`, подчёркивание `_`. Длина каждой записи $A[i]$ не более чем 25. В каждой строке содержится только запись одного числа.

Формат вывода: В единственной строке выводятся искомые номера i_1 и i_2 ($i_1 < i_2$), записанные в десятичной системе и разделённые одиночным пробелом.

Ввод примера №1:

```
5
jw_
b
d
a
kx~
```

Вывод примера №1:

```
1 4
```

Ввод примера №2:

```
4
mx
m~x~
m^x^
m_x_
```

Вывод примера №2:

```
2 3
```

Ввод примера №3:

```
2
bcdrfghijklmnopqrstuvwxyz
bcdrfghijklmnopqrstuvwxyz
```

Вывод примера №3:

```
1 2
```

Решение

В решении стоит использовать массив из 25 элементов со значениями от 0 до 106 как внутреннее представление чисел, записанных 26 разрядами рассматриваемой системы. 106 выбрано верхней границей диапазона, так как потребуется запас для осуществления нормализации числа. Для удобства сравнения старшие разряды стоит хранить в начальных элементах массива. В старших разрядах могут быть незначащие нули. Программа будет в цикле считывать очередной элемент последовательности, осуществлять его нормализацию (т. е. переводить во внутреннее представление в виде массива со значениями разрядов от 0 до 25). Так как заранее известно, что последовательность будет недосортированной по возрастанию, то либо будет только одна пара соседних элементов, стоящих не по порядку ($A[i] > A[i + 1]$), либо только две таких пары ($A[i] > A[i + 1], A[j] > A[j + 1], i < j$). В первом случае ответом будут числа i и $i + 1$, во втором – числа i и $j + 1$. Программе достаточно одного прохода по последовательности, осуществляемого во время её ввода, в ходе которого она будет сравнивать нормализованный массив текущего элемента последовательности с нормализованным массивом предыдущего элемента последовательности и запоминать номера пар, в которых числа не по порядку. После обнаружения второй неупорядоченной пары можно не дочитывать последовательность до конца, так как ответ уже известен.

При нормализации считанного числа следует начинать с младшего разряда, оставлять в нём остаток от деления нацело значения записанной в разряде цифры на 26, а частное прибавлять

к следующему по старшинству разряду. Такие действия проделываются со всеми прочитанными разрядами. Если их меньше 25, то оставшиеся старшие разряды заполняются нулями.

Код возможного решения

```
program NEDOSORT710(input, output);
type    number = array [1..25] of byte;
var  N, I, NUM1, NUM2 : word;
      FLAG : boolean;
      ACURR, APREV : number;
procedure readNumber(var A : number);
var CH : char; I, J : word;
begin for I := 1 to 25 do A[I] := 0;
      read(CH);
      I := 0;
      J := 0;
      while (I < 25) do begin
        case CH of
          'a'..'z' : begin J := J + 1; A[J] := ord(CH) - ord('a') + 26 end;
          '^' : A[J] := A[J] + 26;
          '~' : A[J] := A[J] + 52;
          '_' : A[J] := A[J] + 78;
        end;
        if not (eoln or eof) then begin
          read(CH);
          I := I + 1
        end else I := 25
      end;
      for I := J downto 1 do A[25 - J + I] := A[I];
      for I := 1 to 25 - J do A[I] := 0;
      for I := 25 downto 2 do begin
        A[I - 1] := A[I - 1] + A[I] div 26;
        A[I] := A[I] mod 26
      end;
end;

begin
  readln(N);
  NUM1 := 0;
  NUM2 := 0;
  FLAG := FALSE;
  readNumber(ACURR);
  readln;
  I := 1;
  repeat
    APREV := ACURR;
    readNumber(ACURR);
    if (CompareByte(APREV, ACURR, 25) > 0) then begin
      if (NUM1 = 0) then NUM1 := I
      else begin
        NUM2 := I;
        FLAG := TRUE
      end;
    end;
    I := I + 1;
  if (I < N) then readln
```

```
until (I = N) or FLAG;  
if FLAG then write(NUM1, ' ', NUM2 + 1)  
else write(NUM1, ' ', NUM1 + 1)  
end.
```