PRESEK
List za mlade matematike, fizike, astronome in računalnikarje

ISSN 0351-6652
Letnik **26** (1998/1999)
Številka 6
Stran 355

Martin Juvan:

IŠČEMO BESEDO

Ključne besede: računalništvo, matematika, kombinatorika, faktorielni zapis števil, nizi znakov, sestavljanje besed.

Elektronska verzija: http://www.presek.si/26/1384-Juvan.pdf

IŠČEMO BESEDO – Rešitev s str. 355

Ljudje števila običajno zapisujemo v desetiškem sestavu. V računalništvu večkrat srečamo tudi osnove dva, osem in šestnajst. Seveda pa obstajajo tudi bolj nenavadni zapisi. Videli bomo, da je rešitev naloge povezana z enim od takih zapisov.

Števila lahko zapišemo tudi v "faktorielnem" zapisu. Vsako naravno število k od 0 do $n! - 1$ lahko enolično zapišemo kot

$$k = c_{n-1} \cdot (n-1)! + c_{n-2} \cdot (n-2)! + \ldots + c_2 \cdot 2! + c_1 \cdot 1!\,,$$

kjer za števke c_i, $i = 1, \ldots, n-1$, velja $c_i \in \{0, 1, \ldots, i-1, i\}$. O obstoju in enoličnosti zapisa se lahko prepričamo z matematično indukcijo.

In kako poiščemo faktorielni zapis števila? Podobno, kot poiščemo zapise v bolj običajnih sestavih. Dvojiški zapis števila poiščemo npr. tako, da število zaporedoma celoštevilsko delimo z 2 in beležimo ostanke. Deljenje končamo, ko število postane enako 0. Ostanki, prebrani od zadnjega proti prvemu, dajo dvojiški zapis izbranega števila. Npr.:

$$\begin{array}{rcccccccc}
37 & = & 18 & \cdot & 2 & + & 1 \\
18 & = & 9 & \cdot & 2 & + & 0 \\
9 & = & 4 & \cdot & 2 & + & 1 \\
4 & = & 2 & \cdot & 2 & + & 0 \\
2 & = & 1 & \cdot & 2 & + & 0 \\
1 & = & 0 & \cdot & 2 & + & 1
\end{array}$$

Torej je $37 = 100101_{(2)}$. Posamezna mesta v dvojiškem zapisu so utežena s potencami števila 2: $1, 2, 4, 8, \ldots$ V faktorielnem zapisu pa so posamezna mesta utežena s fakultetami števil: $1!, 2!, 3!, 4!, \ldots$ Pri iskanju zapisa zato na prvem koraku število delimo z 2, na drugem s 3, na tretjem s 4 itn. Tako dobimo

$$\begin{array}{rcccccc}
37 & = & 18 & \cdot & 2 & + & 1 \\
18 & = & 6 & \cdot & 3 & + & 0 \\
6 & = & 1 & \cdot & 4 & + & 2 \\
1 & = & 0 & \cdot & 5 & + & 1
\end{array}
\qquad
\begin{array}{rcccccc}
k & = & k_2 & \cdot & 2 & + & c_1 \\
k_2 & = & k_3 & \cdot & 3 & + & c_2 \\
 & \vdots & & & & & \\
k_{n-1} & = & k_n & \cdot & n & + & c_{n-1}
\end{array}$$

Če je število k manjše od $n!$, potem je zadnji kvocient k_n enak 0. Faktorielni zapis števila 37 je torej $37 = 1201_{(!)}$.

In kakšno zvezo ima faktorielni zapis z iskanjem besed? Recimo, da imamo na voljo črke `a`, `b`, `o`, `r` in `v`. Iz njih lahko sestavimo $5! = 120$ različnih besed. Iščemo po abecedi 38. besedo. Ker je $4! = 24$, se besede od 1. do 24. začno z `a`, od 25. do 48. se začno z `b`, od 49. do 72. z `o`, od 73. do 96. z `r`, zadnjih 24 besed pa se začne z `v`. Iskana beseda se torej začne z `b`. Vseh 24 besed, ki se začno s črko `a`, bo po abecedi pred njo. Zato moramo le še določiti po abecedi 14. besedo ($38 - 24 = 14$), sestavljeno iz črk `a`, `o`, `r` in `v`. Ponovimo zgornji razmislek. Besed je 24, prve črke pa jih razdelijo v skupine po 6. Iskana beseda je tako v 3. skupini, zato se začne s po abecedi tretjo črko, črko `r`. Na enak način določimo tudi nadaljnje črke in najdemo iskano besedo: `bravo`.

Če ste pozorno spremljali gornji razmislek, ste gotovo opazili, da smo pravzaprav opisali še en način za iskanje faktorielnega zapisa. Ker smo besede oštevilčili od 1 do $n!$, s faktorielnim zapisom pa opišemo števila od 0 do $n! - 1$, moramo pogledati zapis števila 37: $37 = 1201_{(!)}$. Števke štejemo od 0 dalje, skupine (oz. začetne črke) pa smo šteli od 1 naprej, zato vsako števko še povečamo za 1. Dobimo zaporedje 2, 3, 1, 2. Izmed črk `a`, `b`, `o`, `r`, `v` izberemo 2., jo odstranimo, med preostalimi vzamemo 3., jo zopet odstranimo, nato 1., pa 2., na koncu pa dodamo še edino preostalo črko. Seveda dobimo besedo `bravo`.

Ponovimo, kako poiščemo k-to besedo. Najprej poiščemo faktorielni zapis števila $k-1$. (Če imamo n črk, bo zapis imel $n-1$ števk; ne smemo pozabiti na vodilne ničle, če so potrebne, da zagotovimo pravo dolžino.) Vsako števko povečamo za 1, nato pa iz zaporedja črk izbiramo črke, kot narekujejo števke. Vsako izbrano črko sproti odstranimo iz zaporedja. Črko, ki ostane, dodamo na konec besede. Opisanega postopka ni težko sprogramirati. V nadaljevanju je zapisan v turbo pascalu kot funkcija `PoisciBesedo`.

```
function PoisciBesedo(crke: string; k: longint): string;
{ Poišče po abecedi k-to besedo, sestavljeno natanko iz črk iz crke. }
{ Črke v nizu crke morajo biti urejene po abecedi.                  }
var
  n: integer;
  stevka: array [1..255] of byte;                 { faktorielni zapis }
  beseda: string;                                      { iskana beseda }
  i: integer;
begin
  n := length(crke);                        { Število razpoložljivih črk. }
```

```
  { Poiščemo zapis števila k - 1 v 'faktorielnem' zapisu. }
  k := k - 1;
  for i:=1 to n - 1 do begin                { Zapis ima n - 1 števk. }
    stevka[n - i] := k mod (i + 1); {Najprej dobimo zadnjo števko.}
    k := k div (i + 1);
  end;

  if k > 0 then begin                       { Število k je preveliko. }
    writeln('Toliko razlicnih besed ne obstaja.');
    PoisciBesedo := '';
    exit;
  end;

  { Zgradimo iskano besedo. }
  beseda := '';
  for i:=1 to n - 1 do begin
    { Pazimo, da števke povečamo za ena. }
    beseda := beseda + crke[stevka[i] + 1];
    delete(crke, stevka[i] + 1, 1);
  end;
  beseda := beseda + crke;                  { Dodamo še preostalo črko. }

  PoisciBesedo := beseda;
end; {PoisciBesedo}
```

Črke, ki jih imamo na voljo za gradnjo besed, smo predstavili z nizom znakov (tip `string`). Taka izbira nam olajša brisanje že porabljenih črk. Uporabimo lahko kar v turbo pascal vgrajeni podprogram `delete` (ta ima tri parametre: niz, iz katerega brišemo znake, indeks prvega zbrisanega znaka in število zbrisanih znakov). Funkcija `PoisciBesedo` zahteva, da so črke v parametru `crke` urejene po abecedi, sicer ne deluje pravilno. Pokličemo jo lahko kar s konstantnim nizom in številom, saj za oba parametra uporabljamo prenos po vrednosti. Tako na primer klic `PoisciBesedo('aborv', 38)` vrne niz `bravo`.

Martin Juvan

KRIŽANKA Z GESLOM – Rešitev s str. 267

EKSPONENTNA ENAČBA.

Darjan Trupi