Žodynas

Uždavinio sprendimas gana paprastas, jį sudaro šios dalys, išplaukiančios betarpiškai iš uždavinio sąlygos:

a) Lietuviškų raidžių prioritetų nustatymas: aukščiausią prioritetą žymėsime skaičiumi 1, jį turės raidės A ir Ą, žemiausias – raidės Ž prioritetas. Jis lygus 25;

b) Žodžių rikiavimas. Žodžius galima rikiuoti bet kokiu rikiavimo algoritmu. Pasirinkome rikiavimą sukeitimu. Rikiavimo metu teks lyginti du žodžius.

Žodžių palyginimui teks sudaryti atskirą algoritmą. Imami du gretimi žodžiai ir lyginami paraidžiui, kol randamos nesutampančios raidės arba pasiekiama žodžio pabaiga. Pirmuoju atveju aukščiau bus tas žodis, kurio raidės prioritetas mažesnis, antruoju – tas, kurio ilgis mažesnis.

```
const MAX = 100;     { maksimalus žodžių skaičius }
      ILG = 20;      { ilgiausias žodis }
type  žodis = string [ILG];
      tžodžiai = array [1..MAX] of žodis;
      prioritetai = array [char] of integer;

procedure rasti_prioritetus (var pr: prioritetai);
{ randa raidžių prioritetus }
begin
  pr['A']:=1;   pr['Ą']:=1;
  pr['B']:=2;   pr['C']:=3;   pr['Č']:=4;
  pr['D']:=5;
  pr['E']:=6;   pr['Ę']:=6;   pr['Ė']:=6;
  pr['F']:=7;   pr['G']:=8;   pr['H']:=9;
  pr['I']:=10;  pr['Į']:=10;  pr['Y']:=10;
  pr['J']:=11;  pr['K']:=12;  pr['L']:=13;
  pr['M']:=14;  pr['N']:=15;  pr['O']:=16;
  pr['P']:=17;  pr['R']:=18;  pr['S']:=19;
  pr['Š']:=20;  pr['T']:=21;
  pr['U']:=22;  pr['Ų']:=22;  pr['Ū']:=22;
  pr['V']:=23;  pr['Z']:=24;  pr['Ž']:=25;
end;  { rasti_prioritetus }
```

```
function prieš (žod1, žod2: žodis; pr: prioritetai): boolean;
{ patikrina, ar pirmasis žodis turi eiti prieš antrąjį }
  var i, ilgis: integer;
begin
  { rasime trumpesniojo žodžio ilgį }
  if length(žod1) <= length(žod2)
     then ilgis := length(žod1)
     else ilgis := length(žod2);
  i := 1;
  while (pr[žod1[i]] = pr[žod2[i]]) and (i < ILGIS) do
    i := i + 1;
  prieš := (pr[žod1[i]] < pr[žod2[i]]) or
             (pr[žod1[i]] = pr[žod2[i]]) and
             (length(žod1) < length(žod2));
end; { prieš }

procedure rikiuoti (kiek: integer; { žodžių skaičius }
                    var žodžiai: tžodžiai);
{ žodžiai surikiuojami naudojant Rikiavimą sukeitimu }
  var pr: prioritetai;
      keista: boolean;
      i: integer;
      tarp: žodis;
begin
  { surasime raidžių prioritetus }
  rasti_prioritetus (pr);
  { rikiuosime žodžius }
  keista := true;   { ar du žodžiai sukeisti vietomis }
  while keista do   { pradedama rikiuoti }
    begin
      keista := false;
      for i := 1 to kiek - 1 do
        if prieš (žodžiai[i+1], žodžiai[i], pr)
           { jei (i+1)-as žodis eina prieš i-ąjį }
           then begin     { žodžius keičiame vietomis }
                  keista := true;
                  tarp := žodžiai[i];
                  žodžiai[i] := žodžiai[i + 1];
                  žodžiai[i + 1] := tarp
                end
    end;
end; { rikiuoti }
```