VIENUOLIKTAS SKYRIUS

Rekursija

Rekursija – tai tam tikras kartojimas. Kartojimas būna natūralus, sutinkamas gamtoje (palikuonys atkartoja tėvus, t. y. perima jų genetines ypatybes), ir dirbtinis (matematinės progresijos, du priešais stovintys veidrodžiai). Toks svarbus procesas vienaip ar kitaip turėjo būti realizuotas ir programavime.

Rekursija apibrėžiama kaip funkcijos (procedūros) kreipimasis į save tiesiogiai arba netiesiogiai, per kitą funkciją (procedūrą) – šiuo atveju funkcija (procedūra) A iškviečia B, o B – A.

Reikia pastebėti, kad ne visose programavimo kalbose (pvz., ankstyvosiose „Fortran“ versijose) rekursija realizuojama tiesiogiai pagal pateiktą apibrėžimą. Kai kuriose kalbose rekursyvūs procesai realizuojami tik iteraciniais metodais (ciklais). Bendru atveju rekursiją visada galima pakeisti ciklu.

Rekursijos sąvoka naudojama matematikoje, bet, skirtingai nuo programavimo, ten šis procesas gali būti begalinis. Naudodami rekursiją programoje, turime užtikrinti jos baigmę: nurodyti sąlygą, kada nebereikia rekursyviai kreiptis į funkciją (procedūrą). Dažniausiai taip būna, kai rekursyvios funkcijos (procedūros) parametras (-ai) įgyja tam tikrą reikšmę (-es), nes būtent parametrais perduodami duomenys. Todėl būtina parametrų reikšmių kaita ribinės reikšmės linkme. Dažnai ji būna apibrėžta rekurentiniu sąryšiu ir privalu jo laikytis.

Išnagrinėkime klasikinį pavyzdį – natūraliojo skaičiaus faktorialo radimą. Faktorialą galima apibrėžti taip:

$$n! = \begin{cases} 1 & ,n = 1; \\ n \cdot (n-1)! & ,n > 1. \end{cases}$$

Suprantama, kad funkcijos parametras n nurodo, kurio skaičiaus faktorialas ieškomas. Rekurentinėje formulėje jis priklauso nuo vienetu mažesnio skaičiaus dar neapskaičiuoti faktorialo. Tai ir yra rekursija. Rekursiniame kreipinyje reikia nurodinėti mažesnę parametro reikšmę tol, kol ji nepasidarys lygi 1. Vyksta rekursinis nusileidimas. Vieneto faktorialas turi konkrėią reikšmę – 1. Ji įstatoma į rekurentinę formulę vietoj 1!, gautas rezultatas – į rekurentinę formulę su $n = 3$ ir t. t., kol n neįgys savo pradinės reikšmės. Taip vyksta rekursinis grįžimas.

Apibrėžiant rekursiją svarbu sudaryti teisingą rekurentinę formulę, t. y. nustatyti, nuo kokio dydžio reikšmės (didesnės ar mažesnės už ieškomą) priklauso ieškomas dydis. Tada belieka keisti rekursinės funkcijos (procedūros) parametrų reikšmes ta linkme. Dažniausia ieškomas dydis priklauso nuo mažesnės reikšmės dydžio.

Uždaviniai

1. *Parašykite programą*, kuri ekrane spausdintų kiekvieną klaviatūra renkamo teksto žodį (žodis sudarytas iš ne daugiau kaip 20 simbolių sekos, tarp jų nėra tarpo simbolių) ir iš karto po jo spausdintų tą žodį atbulai (atbulo žodžio spausdinimas vyksta teksto įvedimo metu!). Žodžiai vienas nuo kito skiriami vienu tarpu, teksto skaitymas iš klaviatūros baigiamas paspaudus įvesties klavišą.

<u>*Reikalavimas*</u>. Uždaviniui išspręsti programoje turi būti panaudota rekursinė procedūra. Negalima naudoti struktūrinių ir dinaminių duomenų tipų.

Pavyzdys:

```
Aš šA rašau uašar programą. .ąmargorp :) ):
```

2. Tarkime, kad įsigijote kompiuterį, nevykdantį natūraliųjų skaičių dalybos operacijų.

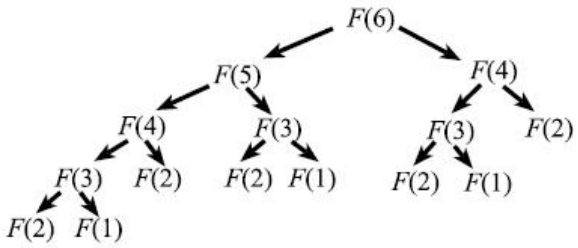
Parašykite rekursinę procedūrą dviejų natūraliųjų skaičių dalybos dalmeniui ir liekanai rasti, t. y. atliekančią operacijas `div` ir `mod`.

3. *Parašykite rekursinę funkciją pow*(x, n), randančią x^n pagal formulę

$$x^n = \begin{cases} 1 & ,n = 0, \\ 1/x^{|n|} & ,n < 0, \\ x \cdot x^{n-1} & ,n > 0. \end{cases}$$

Čia x – realusis skaičius, n – natūralusis skaičius.

4. Fibonacci skaičius galima apskaičiuoti rekursyviai. Rekursyvius kreipinius galima pavaizduoti dvejetainiu medžiu:



Galima pastebėti, kad kai kurie Fibonacci skaičiai randami kelis kartus, kas yra neefektyvu.

Parašykite funkciją, kurioje kiekvienas Fibonacci skaičius būtų rekursyviai apskaičiuojamas tik vieną kartą, o jo reikšmė tolimesniems skaičiavimams būtų laikoma masyve.

5. Faile (rinkmenoje, byloje) parašytas reiškinys. Reiškinį gali sudaryti arba skaitmuo, arba skliaustuose du sujungti ženklu reiškiniai:

<reiškinys> ::= <skaitmuo> | (<reiškinys><ženklas><reiškinys>)

Ženklas gali būti arba sudėties, arba atimties, arba daugybos:

<ženklas> ::= + | - | *

Akivaizdu, kad

<skaitmuo> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Parašykite programą, skaičiuojančią šio reiškinio reikšmę.

Pradinių duomenų ir rezultatų pavyzdys

Pradiniai duomenys (reiškinys)	*Rezultatai*
5	5
((2-4)*6)	12

6. Suformuluotas Hanojaus bokštų uždavinys n diskų skaičiui:

Duoti trys stiebai ir n skirtingo dydžio diskų. Iš pradžių visi šie diskai sumauti ant pirmojo stiebo: apačioje pats didžiausias diskas, ant jo – mažesnis ir t. t. Viršuje užmautas pats mažiausias iš diskų.

Reikia perkelti visus diskus nuo pirmojo stiebo ant paskutiniojo laikantis tokių taisyklių:

- vienu ėjimu galima kelti tik vieną diską;
- diską galima užmauti tik ant tuščio stiebo arba uždėti ant didesnio už jį disko.

Reikia parašyti programą, spausdinančią ėjimus, kuriuos reikia atlikti, kad diskus perkeltume nuo pirmojo stiebo ant trečiojo stiebo, laikydamiesi minėtųjų taisyklių.

Šiam uždaviniui spręsti parašyta programa su rekursine procedūra:

```
program HanojausBokštai;
  var a,b,c: char;
      n: integer;
  procedure perkelk (n: integer; a, b, c: char);
  begin
    if n >= 1
      then
        begin
          perkelk(n-1, a, c, b);
          writeln(a,'-->',c,' ');
          perkelk(n-1, b, a, c);
```

```
      end;
  end;
begin
  write('Įveskite žiedų skaičių:'); readln (n);
  perkelk (n, '1', '2' , '3');
  readln
end.
```

Parašykite nerekursinį Hanojaus uždavinio sprendimą (programą).

Šio skyriaus 1–6 uždavinių sprendimai turi būti pateikti JPM interneto svetainėje (http://ims.mii.lt/jpm/) iki 2009 m. vasario 8 d. 24 val.

Failo vardas turi būti *?????11.** (čia * – *doc*, *txt*, *rtf arba odt*). Vietoj klaustukų įrašykite pirmąsias penkias savo pavardės raides (be diakritinių ženklų). Jei kartais būtų pavardžių, trumpesnių, negu penkios raidės, trūkstami simboliai keičiami pabraukimo brūkšniais „_“.

Elektroninio pašto adresas JPM antrosios dalies klausytojams: <jpm.2kursas@gmail.com>.

7. Naujųjų Metų proga mokiniai nusprendė papuošti ilgą mokyklos koridorių savadarbėmis girliandomis. Ant koridoriaus lubų kabo šviestuvai. Bet kokius du šviestuvus galima sujungti girlianda. Reikia sujungti šviestuvus poromis taip, kad prie kiekvieno šviestuvo būtų prikabinta nors viena girlianda, o visų girliandų bendras ilgis būtų kuo mažiausias.

Parašykite programą, randančią mažiausią visų girliandų bendrą ilgį.

Pradiniai duomenys įrašyti faile *duom.txt*. Pirmoje failo eilutėje įrašytas natūralusis skaičius n ($2 \le n \le 1000$) – šviestuvų skaičius. Kitoje eilutėje įrašyti atskirti tarpais n neneigiamų skaičių, neviršijančių 10000, – kiekvieno šviestuvo koordinatė.

Rezultatas – mažiausias visų girliandų bendras ilgis – įrašomas į failą *rez.txt.*

Pradinių duomenų ir rezultatų pavyzdys

Pradiniai duomenys	*Rezultatai*	*Paaiškinimas*
5 4 10 0 12 2	6	Sujungti šviestuvai, kurių koordinatės 0 ir 2 (girliandos ilgis 2), 2 ir 4 (girliandos ilgis 2), 10 ir 12 (girliandos ilgis 2). Bendras girliandų ilgis – 6.

8. Senelis Sodinukas augina rožes. Jo gėlyną sudaro n lysvelių, kuriuose auga po m rožių krūmų. Deja, senelis pamiršo patrešti savo rožes nuo augalų lygų ir kai kurie krūmai užsikrėtė jomis. Jas teko išrauti. Tai padarius, visas gėlynas gali skilti į mažesnius įvairaus dydžio bei formos rožių krūmynus.

Parašykite: **a**) *sprendimo idėją* ir

b) *programą*, randančią kiek dabar yra atskyrų rožių krūmynų.

Pradiniai duomenys įrašyti faile *duom.txt*. Pirmoje failo eilutėje atskirti tarpu įrašyti du natūralieji skaičiai n ir m ($1 \le n \le m \le 100$). Likusiose n eilučių įrašyta po m neatskirtų tarpais 0 arba 1: 0 rodo, kad šioje vietoje rožių krūmo nėra, 1 – yra.

Rezultatas – rožių krūmynų skaičius – įrašomas į failą *rez.txt.*

Pradinių duomenų ir rezultatų pavyzdys

Pradiniai duomenys	*Rezultatai*	*Paaiškinimas*
3 12 110110110111 100010010111 111011010011	4	11 11 11 111 1 1 1 111 111 11 1 11 Matosi, kad yra 4 rožių krūmynai.

Kartu su 7-o ir 8-o uždavinio sprendimu turi būti pateikta *programa, generuojanti* ***atsitiktinį*** *sąlygą atitinkantį pradinių duomenų rinkinį (atsitiktinį testą).*

Vertinant šių uždavinių sprendimus programos bus tikrinamos (testuojamos) su sąlygose pateiktais pradiniais duomenimis. *Tik tos programos,* kurios su šiais duomenimis duos teisingus rezultatus, bus tikrinamos su specialiai parengtais kontroliniais testais ir vertinamos balais.

Programai surinkus *bent pusę balų* (5), bus vertinama *7-o uždavinio* sprendimo *programavimo stilius (kultūra)* bei *8-o uždavinio* sprendimo *idėjos aprašas*, kuris turi būti parašytas kaip *komentaras sprendimo faile prieš programą*.

Šio skyriaus 7 ir 8 uždavinių sprendimai turi būti pateikti JPM interneto svetainėje (http://ims.mii.lt/jpm/) iki 2009 m. vasario 19 d. 24 val.

Sprendimų failų vardai turi būti *?????117.pas* ir *?????118.pas*, generuojančių programų (*generatorių*) – *?????117gen.pas* ir *?????118gen.pas*. Vietoj klaustukų įrašykite pirmąsias penkias savo pavardės raides (be diakritinių ženklų). Jei kartais būtų pavardžių, trumpesnių, negu penkios raidės, trūkstami simboliai keičiami pabraukimo brūkšniais „_“.

Elektroninio pašto adresas JPM antrosios dalies klausytojams: jpm.2kursas@gmail.com.