BENDRASIS PROGRAMAVIMO DOKUMENTAS

EUROPOS SĄJUNGA
Europos socialinis fondas

ŠVIETIMO IR MOKSLO MINISTERIJA

VYTAUTO DIDŽIOJO UNIVERSITETAS
UNIVERSITAS VYTAUTI MAGNI

KURKIME ATEITĮ DRAUGE!

2004-2006 m. Bendrojo programavimo dokumento 2 prioriteto „Žmogiškųjų išteklių plėtra“ 4 priemonė „Mokymosi visą gyvenimą sąlygų plėtra“

Projekto sutarties numeris: **ESF/2004/2.4.0-K01-160/SUT-261**

Projekto pavadinimas: **Inovatyvūs mokymosi metodai ir naujausios technologijos gamtos mokslų bakalaurų rengimui**

FIZ 313 KOMPIUTERINĖ FIZIKA

Laboratorinis darbas

LOGINIŲ REIŠKINIŲ TAIKYMAS

1. **Darbo tikslas:**

1.1. Ištirti saulės elemento energijos konversijos parametrus.

2. **Darbo užduotis:**

2.1. Parašyti sudėtinį loginį reiškinį, kuris nustato, ar nurodyta situacija teisinga. Reiškinį realizuoti veikiančia programa, kuri įvertintų, ar taškas parinktomis koordinatėmis patenka į pažymėtą plotą. Užduoties numerį parenka dėstytojas.

3. Bendroji teorija

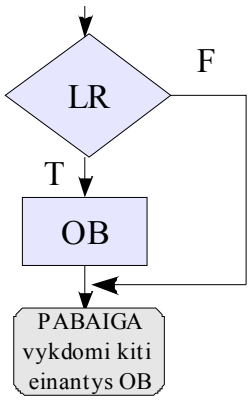
Bet kurioje programavimo kalboje instrukcijos kompiuteriui vykdomos nuosekliai paeiliui. Norint išspręsti uždavinį, reikia instrukcijas sudėlioti teisinga tokia logine tvarka. Bazinės algoritmų struktūros yra trys: operatorių blokas; valdymo perdavimo operatoriai; išsišakojimai; ciklai funkcijos ir paprogramės.

Išsišakojimas – tai programos vykdymo parinkimas iš kelių galimų pagal tam tikras sąlygas. Išsišakojimai įgyvendinami su 4 rūšių sąlygos operatoriais. Sąlygos operatoriai – tai gana didelė grupė konstrukcijų, kurios skirtos programos eigos valdymui. Be šių konstrukcijų neišsiverčia nė viena programa. Aptarsime tris sąlygos operatorių formas nuo paprasčiausios iki pilnosios formos.

IF sąlygos operatorius yra paprasčiausia sąlygos operatoriaus forma:

```
IF (LR) operatorius1
```

Jei loginis reiškinys (LR) teisingas, vykdomas operatorius *operatorius1*; jei LR neteisingas, tai ši sąlygos eilutė tiesiog ignoruojama kaip parodyta 1 paveiksle: kai LR teisingas, atliekamas operatorių blokas OB (jame gali būti ir daugiau kaip vienas operatorius). Atlikęs OB veiksmus programos eiga perduodama tolimesniems operatoriams. Kai LR neteisingas, tada OB tiesiog praleidžiamas.



1 pav. Sąlyginio operatoriaus blokinė schema. Pažymėjimai: LR – loginis reiškinys, F – false (reiškinys neteisingas), T – true (reiškinys teisingas) OB – operatorių blokas.

2 pavyzdyje pateiktas sąlygos operatoriaus vartojimas. Šiame pavyzdyje iš trijų skaičių išrenkamas mažiausias ir jo vertė išvedama į ekraną. Norint iš trijų skaičių išrinkt mažiausią, naudojama standartinė funkcija **min()**, kurios atsakymas (mažiausias skaičius) priskiriamas kintamajam m3. Kadangi nežinome, kuris iš trijų kintamasis mažiausias, toliau einačiuose sąlygos operatoriuose atliekamas patikrinimas. Į ekraną išvada tik ta eilutė, kuri tenkina sąlygą (m? == m3).

```
program fimin
  real :: ma = 5.3, mb = 7.6, mc = 5.3, m3
  m3 = min(ma, mb, mc)
  if (ma == m3) print *, 'Mažiausias ma'
  if (mb == m3) print *, 'Mažiausias mb'
  if (mc == m3) print *, 'Mažiausias mc'
  print  *, 'Mažiausias skaičius lygus = ', m3
end program
```

2 pav. Supaprastintas sąlygos operatoriaus naudojimas.

IF -THEN sąlygos operatorius Sudėtingesnė konstrukcija IF ... THEN naudojama tuomet, kai operatorių blokas yra didesnis.

```
IF (LR) THEN
   OB
```

Tokia forma naudojama tuomet, kai operatorių bloką sudaro ne viena, o kelios operacijos. Kitame pavyzdyje pateiktos programos veikimas niekuo nesiskiria nuo 3. Tačiau tarp THEN ir END IF galima įrašyti ne vieną operatorių.

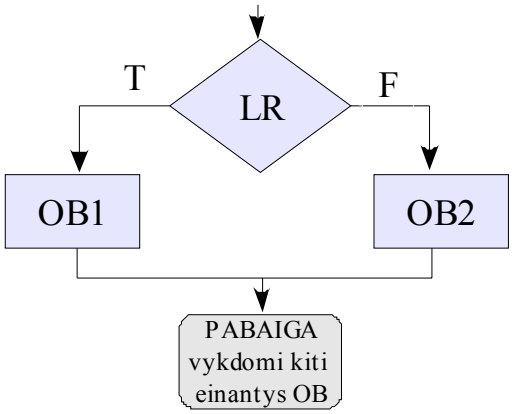
```
program fimin2
 real :: ma = 5.3, mb = 7.6, mc = 5.3, m3
 m3 = min(ma, mb, mc)
 if (ma == m3) then
   print*, 'Mažiausias ma'
   ! galima įrašyti kitus operatorius, pvz.:
   print  *, Mažiausias skaičius lygus = ', m3
 endif
 if (mb == m3) print *, 'Mažiausias mb'
 if (mc == m3) print *,  'Mažiausias mc'
 print  *, 'Mažiausias skaičius lygus = ', m3
end program
```

3 pav. Sąlygos operatoriaus IF .. THEN naudojimas.

IF-THEN-ELSE sąlygos operatorius Pati bendriausia sąlygos operatoriaus forma yra tokia:

```
IF (LR) THEN
  OB-1
ELSE
  OB-2
END IF
```

Jei loginis reiškinys teisingas, tuomet vykdomas operatorių blokas OB1, o jeigu neteisingas – OB2. Žemiau pateiktoje schemoje matyti, kad šiuo atveju turime programos vykdymo išsišakojimą: vykdoma ta atšaka, kuri atitinka loginio reiškinio sąlygą.



4 pav. Pilnosios sąlygos operatoriaus formos IF..THEN .. ELSE.. ENDIF blokinė schema. Pažymėjimai: LR – loginis reiškinys, F – false (reiškinys neteisingas), T – true (reiškinys teisingas) OB – operatorių blokas.

Pilnosios sąlygos operatoriaus formos naudojimas iliustruojamas 5 pavyzdyje.

```
program fimin3
 real :: ma = 5.3, mb = 7.6, mc = 5.3, m3
 m3 = min(ma, mb, mc)
 if (ma == m3) then
   print*, 'Mažiausias ma'
 else
   print*, 'Mažiausias mb arba mc'
 endif
 if (mb == m3) print *, 'Mažiausias mb'
 if (mc == m3) print *,  'Mažiausias mc'
 print  *, 'Mažiausias skaičius lygus = ', m3
end program
```

5 pav. Pilnosios sąlygos operatoriaus formos IF..THEN .. ELSE.. ENDIF.

IF-THEN-ELSE-IF sąlygos operatorius Norint įvykdyti 5 pavyzdyje pateiktą užduotį, t.y., rasti, kuris iš trijų skaičių mažiausias, reikia panaudoti dvigubą sąlygos operatorių formą, kai IF .. THEN yra įklijuotas jau esamama IF..THEN (*nested IF-THEN-ELSE-ENDIF*). Blokinę schemą galima atvaizduoti taip:

```
IF (LR1) THEN
  operatoriai
  IF (LR2) THEN
    operatoriai
  ELSE
OB1
    operatoriai
  END IF
  operatoriai
ELSE
  operatoriai
  IF (LR3) THEN
    operatoriai
OB2
  END IF
  operatoriai
ENDIF
```

Šioje schemoje yra trys sąlygos operatoriai, du iš jų yra didžiojo IF operatorių blokuose kaip atskiros operatorių konstrukcijos. Tokių „įklijavimų“ galima daryti kiek tik norima, tačiau tuomet pasidaro sunku suprasti tokių daugialypių blokų loginę prasmę. Naudojant dvigubą formą, mažiausio skaičiaus radimo užduotį galima išspręsti taip kaip parodyta 6 pavyzdyje.

```
program fimin4
    real :: ma = 5.3, mb = 7.6, mc = 5.3, m3
    m3 = min(ma, mb, mc)
    if (ma == m3) then
        print*, 'Mažiausias ma'
    else
        if (mb == m3) then
          print*, 'Mažiausias mb arba mc'
         else
          print*, 'Mažiausias mb arba mc'
        endif
    endif
    print  *, 'Mažiausias skaičius lygus = ', m3
end program
```

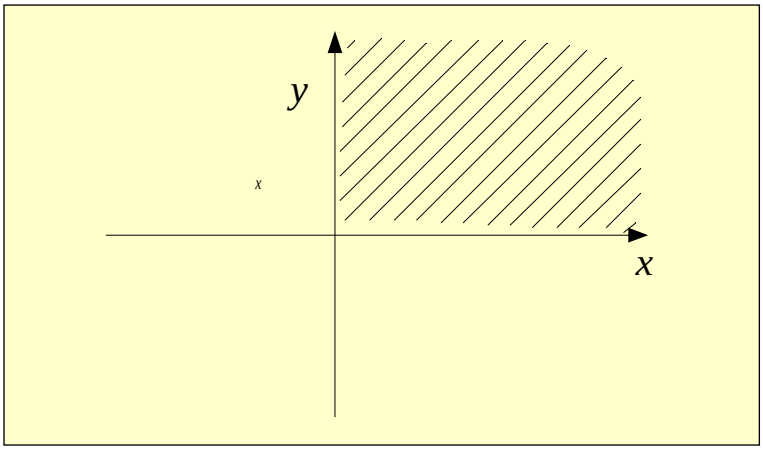
6 pav. Dvigubos formos sąlygos operatoriaus formos IF..THEN .. ELSE.. ENDIF naudojimo iliustracija.

Šiame pavyzdyje matyti, kodėl tvarkingo kodo rašymo taisyklėse yra reikalavimas rašyti programų tekstus „reljefine“ forma. Kad būtų lengviau suprasti ir skaityti programos tekstą, kiekvieną vidinį operatorių bloką priimta paslinkti dešinėn.

4. Tyrimo metodika

Neretai sąlygos operatoriuose naudojami sudėtingi loginiai reiškiniai.

Tarkime, reikia nustatyti, ar taškas patenka į užbrūkšniuotą plotą (7 pav.), kurį riboja koordinačių ašys $x \geq 0$ ir $y \geqslant 0$. Tai reiškia, kad šias dvi sąlygas sujungti, t.y., loginis reiškinys turi būti teisingas (taškas patenka į užbrūkšniuotą plotą), kai teisingi du teiginiai: $x \geqslant 0$ ir $y \geqslant 0$.



7 pav. Parašyti loginį reiškinį, kuris leistų nustatyti, ar taškas (x_0, y_0) patenka į pažymėtą plokštumos ketvirtį.

Šią kombinuotą sąlygą galima įgyvendinti dviem būdas: naudojant dvigubą IF-THEN konstrukciją arba naudojant sudėtinį loginį reiškinį. Abu sprendimai pateikti žemiau.

```
program IFplotas1_1
    ! pradinių duomenų įvestis (praleista)
    if (x0 > =0) then
        if (y0 >= 0) then
          print*, x0, y0, 'taškas patenka į plotą'
        else
          print*, x0, y0, 'nepatenka į plotą y0<0'
        endif
    else
        print*, x0, y0, 'nepatenka į plotą x0<0'
    endif
end program
```

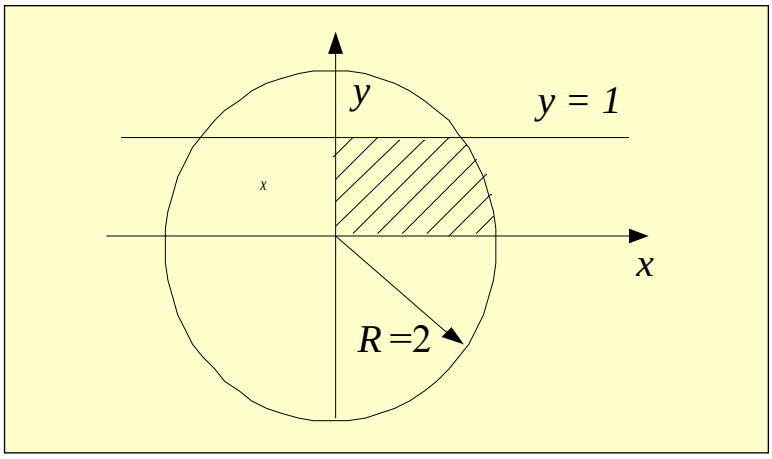
8 pav. 7 užduoties sprendimas dviguba IF-THEN konstrukcija.

Antrame sprendimo būde (8 pav.) panaudotas loginis reiškinys su operatoriumi (.and.). Prie IF stovintis sudėtinis loginis reiškinys teisingas tik tuomet, kai teisingas ir pirmas, ir antras loginis operandas, t.y., ir $x \geq 0$, ir $y \geq 0$. Kai šios abi sąlygos tenkinamos, spausdinamas pranešimas, kad taškas pateko į užbrūkšniuotą plotą; kitu atveju (ELSE) – taškas nepatenka į plotą.

```
program IFplotas1_2
    ! pradinių duomenų įvestis (praleista)
    if ((x0 >= 0) .and.  (y0 >= 0)) then
        print*, x0, y0, 'taškas patenka į plotą'
    else
        print*, x0, y0, 'nepatenka į plotą'
    endif
end program
```

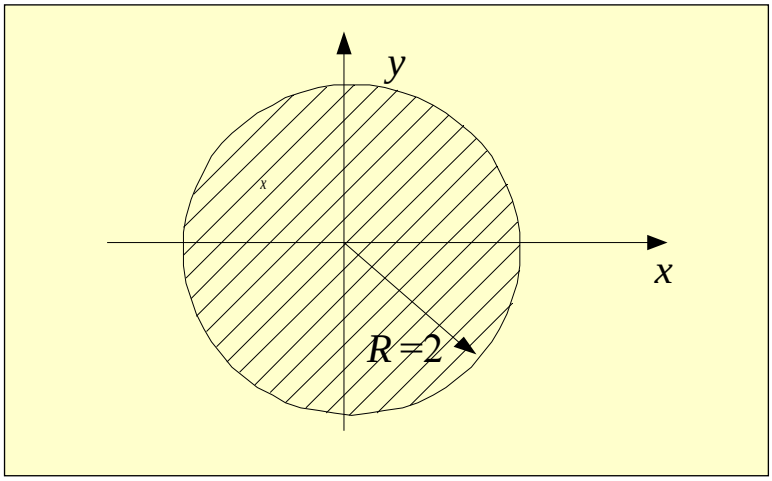
9 pav. 7 užduoties sprendimas su sudėtiniu loginiu reiškiniu.

Dabar padarykime sudėtingesnę užduotį: parašykime loginį reiškinį, kuris patikrintų, ar duotasis plotas patenka į keturiomis kreivėmis apribotą plotą.



10 pav. Parašyti loginį reiškinį, kuris patikrintų, ar taškas (x_0, y_0) patenka į pažymėtą plotą.

Dvi sąlygos jau yra aiškios: $x_0 \geq 0$ ir $y_0 \geq 0$. Trečiąją sąlygą apibrėžia tiesė $y = 1$: visiems taškams, patenkantiems į plotą, y_0 turi būti $y_0 \geq 1$. Iš dešinės plotą riboja apskritimo lankas. Apskritimo, kurios centras koordinačių pradžioje (0,0), lygtis yra $x_0^2 + y_0^2 = R^2$. Tada į skritulio vidinę dalį patenka visi taškai, kurių koordinatės tenkina sąlygą $x_0^2 + y_0^2 \leq R^2$. Ir atvirkščiai: skritulio išorėje yra visi taškai, kurie tenkina sąlygą $x_0^2 + y_0^2 > R^2$.



11 pav. Į skritulio vidinę dalį patenka visi taškai, kurių koordinatės tenkina sąlygą $x_0^2 + y_0^2 \le R^2$. Ir atvirkščiai: skritulio išorėje yra visi taškai, kurie tenkina sąlygą $x_0^2 + y_0^2 > R^2$.

Tuo būdu taškas yra užbrūkšniuotame plote tik tuomet, kai vienu metu tenkinamos 4 sąlygos: $x_0 \geq 0, y_0 \geq 0, y_0 \leq 1,\ x_0^2 + y_0^2 \leq R^2$.

Užduoties sprendimas gali būti užrašytas taip kaip pateikta 12 pavyzdyje.

```
program IFplotas2_1
    ! pradinių duomenų įvestis (praleista)
    if ((x0 >= 0) .and.  (y0 >= 0) .and. (y0 <= 1)  &
        x0**2 + y0**2 <= R**2) then
        print*, x0, y0, 'taškas patenka į plotą'
    else
        print*, x0, y0, 'nepatenka į plotą'
    endif
end program
```

12 pav. 11 užduoties sprendimas su sudėtiniu loginiu reiškiniu.

10 užduoties sprendimas naudojant IF-THEN konstrukcijas galimas, tačiau būtų gana gremėzdiškas, todėl tokiais atvejais geriau naudoti sudėtinį loginį reiškinį.

5. Tyrimo eiga

1. Laboratorinis darbas atliekamas kompiuterių klasėje.

2. Išanalizuokite brėžinį, kurį parenka dėstytojas, ir parašykite sudėtinį loginį reiškinį, kuris nustato, kada nurodyta situacija teisinga. Pateiktuose brėžiniuose koordinačių mastelį parinkti pagal tai, jog koordinačių ašies ilgis yra 6.

3. Parašykite programą, kuri įvertintų, ar taškas parinktomis koordinatėmis patenka į pažymėtą plotą.

4. Pradiniai duomenys – taško koordinatės (x, y) - turi būti įvedami klaviatūra, patikros, ar taškas su nurodytomis koordinatėmis patenka į pažymėtą plotą, rezultatas turi būti išvedamas į ekraną.

5. Rašant programą laikytis tvarkingo rašymo taisyklių, t.y., IF-THEN konstrukcijų blokai turi matytis pagal „reljefinį“ kodo užrašymą.

6. Atsiskaitomų darbų projektai turi būti saugomi tvarkingoje asmeninių failų struktūroje. Atsiskaitymo metu mokėti paaiškinti su fortran programos kompliavimu ir vykdymu susijusius klausimus FPS 4.1 sistemoje.

7. Atsiskaitymas vyksta prie kompiuterio klasėje, būtina mokėti paaiškinti pagrindinius programos struktūros elementus ir jos veikimą.

6. Kontroliniai kiausimai

1. Loginiai operatoriai, loginiai reiškiniai, jų skaičiavimas.
2. Sudėtiniai loginiai reiškiniai, jų skaičiavimas.
3. Programos algoritmas, programos vykdymo eigos valdymas.
4. Išsišakojimų struktūros. Pateikti pavyzdį, iliustruojantį jų reikalingumą fizikinio uždavinio kompiuterinio sprendimo valdymą.
5. Pateiktosios programos struktūra, veikimo principas.

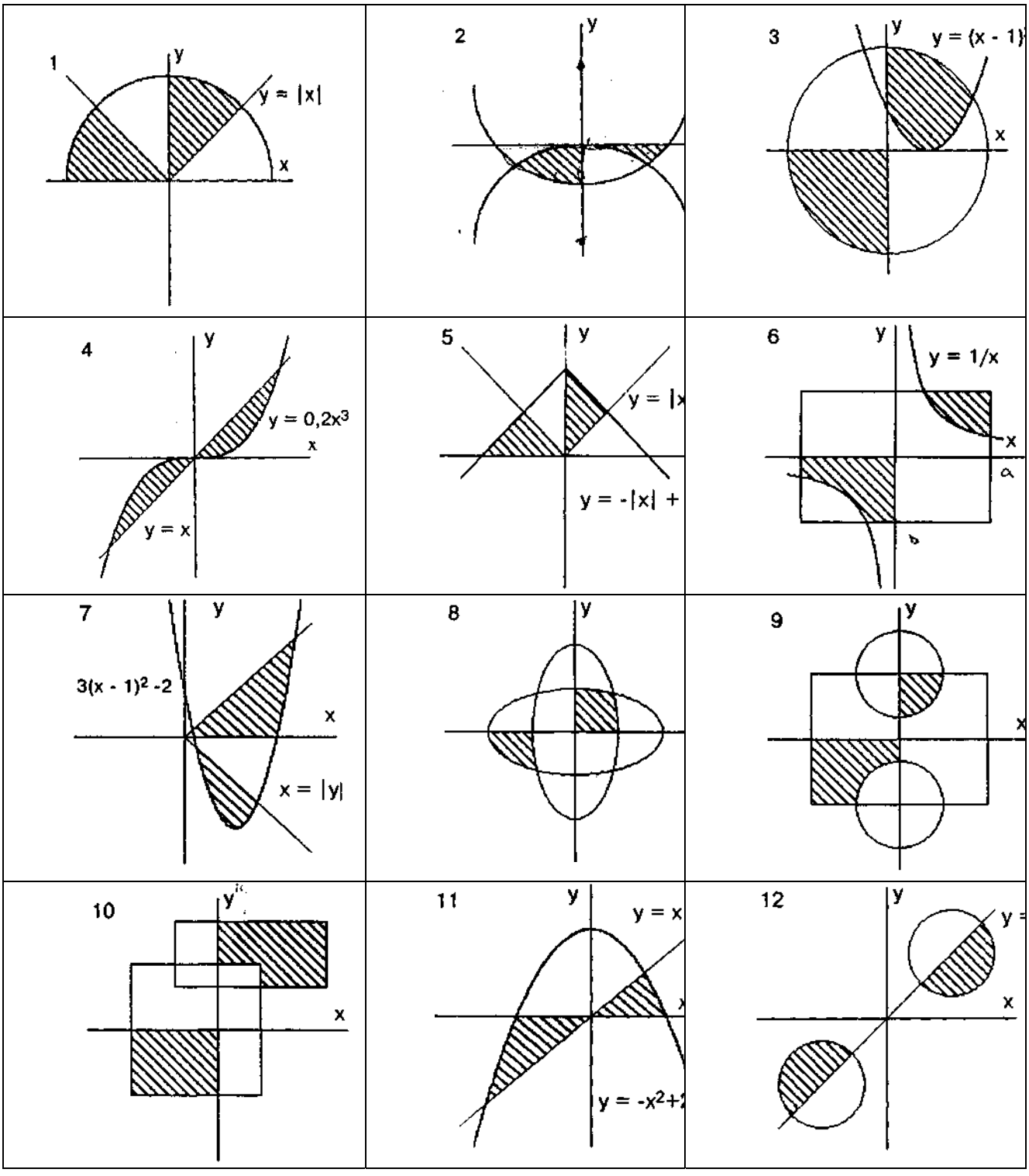
7. Literatūra

1. A. Kanapickas. Paskaitų konspektas. 3 skyrius.

2. MS FPS 4.1 pagalbos sistema: *Microsoft Developer Studio User's Guide -> Chapter 5 -> The IF construct.*

3. R. Belevičius, R. Kutas. FORTRANAS. Vadovėlis. V.: 1998. 241 p., 5 skyrius.

1 priedas.


1
y
y = |x|
x
2
y
3
y
y = (x - 1)
x
4
y
y = 0,2x³
x
y = x
5
y
y = |x
y = -|x| +
6
y
y = 1/x
x
a
b
7
y
3(x - 1)² -2
x
x = |y|
8
y
9
y
x
10
y
x
11
y
y = x
y = -x²+
12
y
y =
x