Vilniaus
universitetas

Informatikos ir informatinio mąstymo mokomoji veikla

Logika informatikoje

Kuriame Lietuvos ateitį
2014–2020 metų Europos Sąjungos fondų investicijų veiksmų programa

Vilniaus universitetas

Logika informatikoje

Informatikos ir informatinio mąstymo mokomosios veiklos sukurtos įgyvendinant projektą „Aukštųjų mokyklų tinklo optimizavimas ir studijų kokybės gerinimas Šiaulių universitetą prijungiant prie Vilniaus universiteto“ (Nr. 09.3.1-ESFA-V-738-03-0001), finansuojamą iš Europos socialinio fondo lėšų pagal 2014–2020 metų Europos Sąjungos fondų investicijų veiksmų programos 9 prioriteto „Visuomenės švietimas ir žmogiškųjų išteklių potencialo didinimas“ įgyvendinimo priemonę Nr. 09.3.1-ESFA-V-738 „Aukštųjų mokyklų tinklo tobulinimas“.

Metodinė medžiaga „Informatikos, informatinio mąstymo mokomoji veikla. Logika informatikoje“ skirta informatikos ir informacinio mąstymo mokymui. Tikslinė grupė – būsimi pagrindinio ir vidurinio ugdymo informatikos mokytojai. Medžiaga siejasi su informatikos ir matematikos Bendrosiomis programomis, duomenų tyrybos, algoritmų ir programavimo pasiekimų sritimis. Atlikdami veikloje numatytas užduotis būsimi mokytojai išsiaiškins, kas yra logika, teiginių logika, teisingumo lentelės, loginės operacijos, kaip spręsti konkrečius logikos uždavinius. Pateikiamas teorinis temos pagrindimas mokytojui, aptariamos pagrindinės logikos srities sąvokos.

Šios veiklos autoriai: Alvida Lozdienė, Viktoras Dagys

Redagavo: Viktoras Dagys

Iliustravo: Alvida Lozdienė

Logika

Logika – išvertus iš graikų kalbos žodžio *logos (λογική)* – „žodis“ arba „protas“. Transcendentinę logiką I. Kantas aiškino kaip intelekto (kategorijų, pagrindinių teiginių) ir proto (idėjų) apibrėžčių sistemą. Dialektinę logiką G. Hėgelis nagrinėjo dialektiškai jungdamas transcendentines intelekto apibrėžtis su ontologinėmis būties apibrėžtimis. Formaliąją logiką Aristotelis apibūdino kaip mokslą apie mąstymo ir kalbėjimo išvadų formalų taisyklingumą.

Per daugiau nei du tūkstantmečius formalioji logika nuo antikinės ir scholastinės logikos etapų susiformavo į matematinę (dažnai vadinamą simboline) logiką tik XIX a. viduryje. Pagrindiniai matematinės logikos kūrėjai: G. Būlis, A. de Morganas, F. L. G. Frege, E. Schröderis, Ch. Peircas, B. Russellas, J. Łukasiewiczius, A. Tarskis, D. Hilbertas, K. Gödelis, R. Carnapas.

Teiginių logika

Teiginys žymi tam tikrą dalykų padėtį. Mes skiriame **elementarius** teiginius (angl. *atomic proposition*): pavyzdžiui, „Vilnius yra Lietuvos sostinė“ ir **sudėtinius** teiginius. Sudėtiniuose teiginiuose jungtimis „ir“, „arba“ ir kitomis sujungiami elementarūs teiginiai.

Teiginiai pasižymi svarbia savybe: jie gali būti teisingi arba klaidingi. Taigi teiginiai gali turėti dvi reikšmes (teisinga, klaidinga), todėl šiuo atveju kalbama apie dvireikšmę teiginių logiką.

Pagrindinis teiginių logikos klausimas: kaip jungiant teiginius į sudėtinius, gauta teisingumo reikšmė priklauso nuo jungiamųjų teiginių teisingumo reikšmės? Pavyzdžiui, du teiginius „šviečia saulė“ ir „lauke šilta“ galime sujungti įvairiais būdais, naudojant įvairias jungtis „jeigu – tai“, „arba“, „ir“.

Naudojant du teiginius, galima sukurti 16 skirtingų jungčių, kurios dažnai vadinamos operatoriais. Naudodamiesi keliais pagrindiniais operatoriais ir jais jungdami teiginius į sudėtingus teiginius, juos vėl jungdami gausime visus 16 operatorių. Informatikoje operatoriai dažniausiai vadinami loginėmis operacijomis.

Teisingumo lentelės

Teisingumo lentelės (dar vadinamos matricomis) labai palengvina loginių teiginių, turinčių tik dvi reikšmes – „tiesa“ (žymėsime T) ir „netiesa“ (žymėsime (N), – jungimo galimybes. Teisingumo lentelėse vaizduojami ne patys teiginiai, bet vadinamieji **propoziciniai kintamieji**. Propozicinis kintamasis yra būtinas visų taisyklingų teiginių logikos formulių elementas: teiginių logikos formulė gali įgyti teiginio reikšmę tik todėl, kad teiginio reikšmę galima priskirti propoziciniam kintamajam. Teisingumo lentelės viršuje užrašoma propoziciniai kintamieji, kurie paprastai žymimi mažosiomis raidėmis (gali būti su indeksais). Toliau eilutėmis surašomos visos galimos kintamųjų reikšmės.
Panagrinėkime du kintamuosius *p* ir *q*. Už vertikalaus brūkšnio stulpeliais surašykime galimus jungimo būdus. Juos pažymėkime O_1, ..., O_{16}.

p	q	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	O_9	O_{10}	O_{11}	O_{12}	O_{13}	O_{14}	O_{15}	O_{16}
N	N	T	K	T	K	T	K	T	K	T	K	T	K	T	K	T	K
N	T	T	T	K	K	T	T	K	K	T	T	K	K	T	T	K	K
T	N	T	T	T	T	K	K	K	K	T	T	T	T	K	K	K	K
T	T	T	T	T	T	T	T	T	T	K	K	K	K	K	K	K	K

Toliau nagrinėdami loginių elementų ir iš jų sudarytų loginių schemų teisingumo lenteles, *p* ir *q* vadinsime loginiais dėmenimis, operandais arba loginiais kintamaisiais, kurie, kaip ir loginiai teiginiai, taip pat gali įgyti tik dvi reikšmes „tiesa“ (informatikoje dažnai žymima loginiu simboliu „1“) ir „netiesa“ (informatikoje dažnai žymima loginiu simboliu „0“).

Užduotis

Sukurkite teisingumo lentelę su galimais junginiais, kai yra tik vienas kintamasis (2 eilutės) ir, kai yra 3 kintamieji (8 eilutės).
Kiek skirtingų operatorių stulpelių gavote kiekvienu atveju?

Pagrindiniai loginių teiginių junginiai

Panagrinėsime kelis pagrindinius teiginių junginius, kurie naudojami ir filosofijoje, ir informatikoje (programuojant), ir kuriant bei konstruojant schemas, ir kompiuteriu atliekant įvairias užduotis.

Konjunkcija

Konjunkcija (lot. *conjunctio* – sujungimas, ryšys) – tai dviejų ar daugiau paprastų teiginių sujungimas jungtimi „ir“. Pavyzdžiui, „Marytė mokosi groti pianinu ir Jonas skaito knygą“. Skaičiavimo technikoje konjunkcijos operacija dar vadinama logine daugyba.

Konjunkcija žymima reiškiniu $p \cdot q$, $p \wedge q$ ar $p\&q$. Mes naudosime konjunkcijos simbolį $\wedge$.

Konjunkcijos teisingumo lentelė yra tokia:

p	q	$p \wedge q$
N	N	N
N	T	N
T	N	N
T	T	T

Tai reiškia, kad tik vienintelių atveju, kai teisingas *p* ir teisingas *q*, konjunkcijos rezultatas yra tiesa.
Formulės $p \wedge q$ teisingumo lentelė perteikia konjunkcijos operatoriaus reikšmę.

Konjunkcija sujungtos teiginių logikos formulės arba sakiniai vadinami konjunktais. Konjunkcija vadinamas ne tik operatorius, bet ir formulės arba sakiniai, sudaromi panaudojant konjunkcijos operatorių.

Konjunkcijos taisyklė: jei konjunktai teisingi, konjunkcija yra teisinga, o jei bent vienas konjunktas klaidingas, konjunkcija yra klaidinga.

Konjunkcijos lentelė (matrica) parodo, kokią reikšmę turinčiu teiginiu virsta propozicinių kintamųjų konjunkcija, kai propoziciniams kintamiesiems juos interpretuojant priskiriama konkreti teiginio reikšmė.

Matematikoje (Būlio algebroje) konjunkcijos teisingumo lentelė užrašoma suteikiant kintamiesiems (loginiams kintamiesiems arba dėmenims, arba operandams) ir jų junginiams (loginėms operacijoms, loginėms funkcijoms) logines reikšmes „1“ arba „tiesa“ ir „0“ arba „netiesa“.

p	q	$p \wedge q$
0	0	0
0	1	0
1	0	0
1	1	1

Kuriant algoritmus, konjunkcija žymima mažosiomis (and) arba didžiosiomis raidėmis (AND).

Užduotis

Sukurkite trijų loginių teiginių (p, q, k) konjunkcijos ($p \wedge q \wedge k$) teisingumo lentelę

Atsakymas

p	q	k	$p \wedge q \wedge k$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Pavyzdys iš programavimo

Duoti du sveikieji skaičiai a ir b. Jei jie abu lyginiai (patenkinama sąlyga: x mod 2 = 0) reikia rasti jų sumą (s), jei ne – jų sandaugą (m).

```
if (a mod 2 = 0) and (b mod 2 = 0)
        then (s := a + b) else (m := a * b).
```

Arba:

if (a mod 2 < > 1) and (b mod 2 < > 1)

then (s := a + b) else (m := a * b).

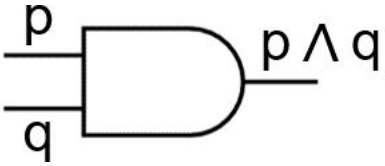
Galima pasinaudoti ir loginiu kintamuoju. Pavyzdžiui, *k* yra loginis kintamasis (k : boolean), o *a* ir *b* – sveikieji skaičiai (a, b : integer), tai:

k : = (a mod 2 = 0) and (b mod 2 = 0);

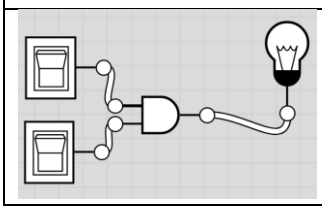
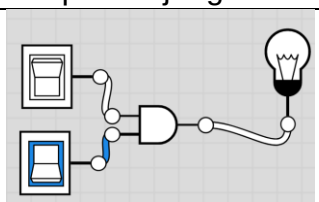
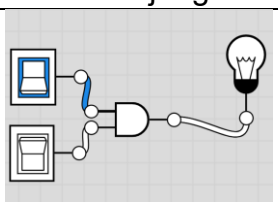
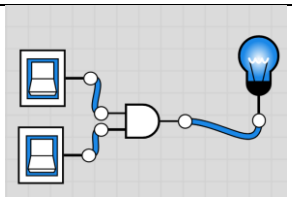
if k then (s := a + b) else (m := a * b).

Konjunkcijos loginis elementas

Elektroninėse schemose Būlio algebros operaciją atlieka loginiai elementai (angl. *logic gate, logic element*).
Konjunkcijos loginis elementas dar vadinamas „AND“ ir dažniausiai vaizduojamas taip:



Konjunkcijos elemento (AND) veikimą galima pavaizduoti taip:

Abu jungikliai išjungti	Įjungtas tik apatinis jungiklis	Įjungtas tik viršutinis jungiklis	Abu jungikliai įjungti
			
Lemputė nedega (rezultatas – „netiesa“, „0“)			Lemputė dega („tiesa“, „1“)

Schemoms kurti naudota aplinka https://logic.ly/

Disjunkcija

Disjunkcijos operatorių žymime teiginių logikos kalbos simboliu „∨“. Dviejų kintamųjų *p* ir *q* disjunkcijos taisyklinga formulė yra *p* ∨ *q* (skaitome: „*p* arba *q*“). Disjunkcija sujungti sakiniai arba teiginių logikos formulės vadinamos disjunktais. Disjunkcijos reikšmę turintį lietuvių kalbos jungtuką galima atrinkti taip pat, kaip buvo atrinkti neigimo ir konjunkcijos atitikmenys.

Disjunkcijos teisingumo lentelė yra tokia:

p	*q*	*p* ∨ *q*
N	N	N
N	T	T
T	N	T
T	T	T

Disjunkcijos taisyklė: jei disjunktai yra klaidingi, disjunkcija yra klaidinga, o jei bent vienas disjunktas teisingas, disjunkcija yra teisinga.

Disjunkcija vadinamas tiek operatorius, tiek formulės bei sakiniai, sudaromi panaudojant disjunkcijos operatorių. Disjunkcijos operatorius yra diadinis.

Jis dar vadinamas operatoriumi „arba“ ir atskyrimo operatoriumi (lot. *disjunctio* – atskyrimas).

Pavyzdžiui, „Šiandien vakare skaitysiu knygą arba eisiu pasivaikščioti“. Jei bent vienas teiginys teisingas, disjunkcijos rezultatas teisingas.

Būlio algebroje disjunkcijos teisingumo lentelė tokia:

p	q	$p \vee q$
0	0	0
0	1	1
1	0	1
1	1	1

Kuriant algoritmus disjunkcija žymima mažosiomis (or) arba didžiosiomis raidėmis (OR).

Užduotis

Sukurkite trijų loginių teiginių (p, q, k) disjunkcijos ($p \vee q \vee k$) teisingumo lentelę.

Atsakymas

p	q	k	$p \vee q \vee k$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Pavyzdys iš programavimo

Jei sveikasis skaičius dalus iš 2 arba iš 5, tai jį reikia pakelti kvadratu, priešingu atveju – kubu.

```
if (a mod 2 = 0) or (a mod 5 = 0)
        then (s := a * a) else (m := a * a * a).
```

Skaičius bus keliamas kvadratu, jei *a* lygus, pavyzdžiui: 4, 20, 15. Skaičius bus keliamas kubu, jei *a* lygus, pavyzdžiui: 7, 21, 203.

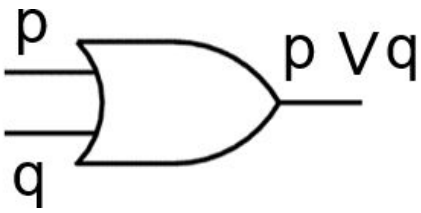
Galima pasinaudoti ir loginiu kintamuoju. Pavyzdžiui *k* yra loginis kintamasis (k : boolean), o *a* – sveikasis skaičius (a : integer):

k : = (a mod 2 = 0) or (a mod 5 = 0);

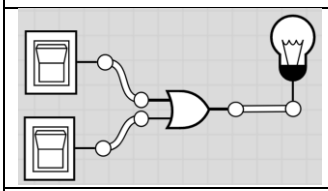
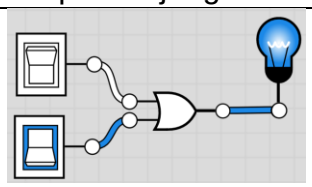
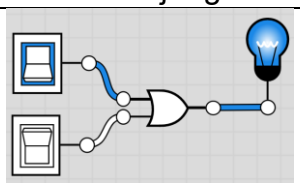
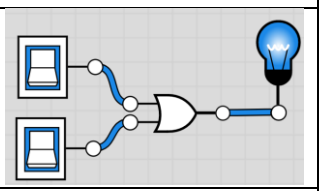
if k then (s := a * a) else (m := a * a * a).

Disjunkcijos loginis elementas

Disjunkcijos loginis elementas dar vadinamas „OR“ ir dažniausiai vaizduojamas taip:



Disjunkcijos elemento (OR) veikimą galima pavaizduoti taip:

Abu jungikliai išjungti	Įjungtas tik apatinis jungiklis	Įjungtas tik viršutinis jungiklis	Abu jungikliai įjungti
			
Lemputė nedega („netiesa“, „0“)	Lemputė dega (rezultatas – „tiesa“, „1“)		

Schemoms kurti naudota aplinka https://logic.ly/

Žodis „arba“ nėra vienareikšmis. Natūralios kalbos sakiniuose galima aptikti du skirtingus disjunkcijos variantus. Jie atitinka silpnąją disjunkciją ir griežtąją disjunkciją.

Mūsų aptartoji disjunkcija vadinama silpnąja disjunkcija, nes rezultatas taip pat teisingas, jei ir abu teiginiai teisingi.

Griežtoji disjunkcija

Griežtoji disjunkcija (angl. *exclusive or*) yra jungtuko „arba…, arba…“ loginė reikšmė.

Pateikiame sakinio su griežtąja disjunkcija ir jam lygiavertaus sudėtinio sakinio pavyzdį: „Arba šiuo metu yra diena, arba naktis“.

Griežtosios disjunkcijos operatorių žymime teiginių logikos kalbos simboliu „⊕“. Dar yra naudojamas simbolis ⊻, o kartais ≠.

Griežtosios disjunkcijos teisingumo lentelė yra tokia:

p	*q*	$p \oplus q$
N	N	N

N	T	T
T	N	T
T	T	N

Griežtosios disjunkcijos operatorius dar vadinamas ekvivalentiškumo neigimo operatoriumi.

Būlio algebroje griežtosios disjunkcijos teisingumo lentelė tokia:

p	*q*	$p \oplus q$
0	0	0
0	1	1
1	0	1
1	1	0

Kuriant algoritmus griežtoji disjunkcija žymima mažosiomis (xor) arba didžiosiomis raidėmis (XOR).

Pavyzdys iš programavimo

Jei sveikasis skaičius *a* dalus tik iš 2 arba tik iš 5, tai jį reikia pakelti kvadratu, priešingu atveju – kubu.

if (a mod 2 = 0) xor (a mod 5 = 0)

then (s := a * a) else (m := a * a * a).

Skaičius bus keliamas kvadratu, jei *a* lygus, pavyzdžiui: 4, 25, 15. Skaičius bus keliamas kubu, jei *a* lygus, pavyzdžiui: 7, 20, 203.

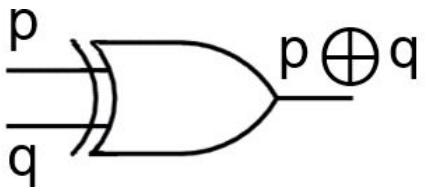
Galima pasinaudoti ir loginiu kintamuoju. Pavyzdžiui, *k* yra loginis kintamasis (k : boolean), o *a* – sveikasis skaičius (a : integer):

k : = (a mod 2 = 0) xor (a mod 5 = 0);

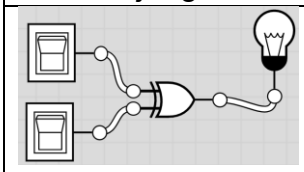
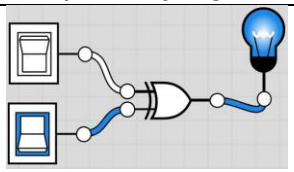
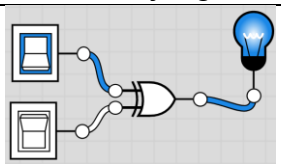
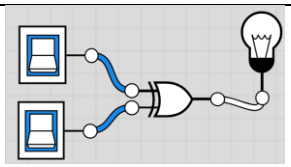
if k then (s := a * a) else (m := a * a * a).

Griežtosios disjunkcijos loginis elementas

Griežtosios disjunkcijos loginis elementas vadinamas „XOR“ ir dažniausiai vaizduojamas taip:



Griežtosios disjunkcijos elemento (XOR) veikimą galima pavaizduoti taip:

Abu jungikliai išjungti	Įjungtas tik apatinis jungiklis	Įjungtas tik viršutinis jungiklis	Abu jungikliai įjungti
			
Lemputė nedega (rezultatas – „netiesa“, 0)	Lemputė dega („tiesa“, 1)		Lemputė nedega (rezultatas – „netiesa“, 0)

Schemoms naudota aplinka https://logic.ly/.

Užduotis

Sukurkite nagrinėtų loginių elementų veikimo schemas ir jas išbandykite pagal pateiktus pavyzdžius. Naudokite tą pačią aplinką: https://logic.ly/.

Neigimas (inversija)

Neigimo operatoriaus (operacijos, jungties) taisyklė: neigimas pakeičia teisingą teiginį klaidingu ir atvirkščiai – klaidingą teisingu.

Neigimui, kaip ir kitoms loginėms jungtims, yra naudojami įvairūs simboliai: ¬, ˜, !.

Neigimo (inversijos) teisingumo lentelė

p	¬*q*
FALSE	TRUE
TRUE	FALSE

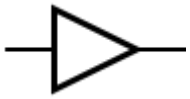
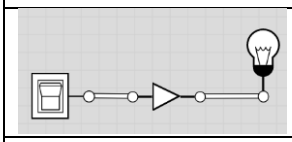
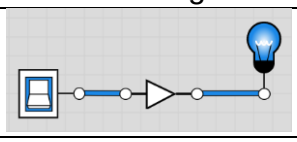
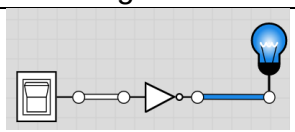
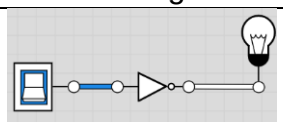
Būlio algebroje neigimo (inversijos) teisingumo lentelė tokia:

p	¬*q*
0	1
1	0

Kuriant algoritmus neigimas žymimas mažosiomis (not) arba didžiosiomis raidėmis (NOT).

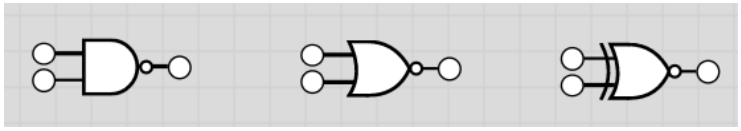
Neigimo (inversijos) loginis elementas (inverteris)

Paprasčiausias ir dažnai naudojamas loginis elementas yra elementas „NE“ (arba inverteris, angl. *inverter*, NOT). Jis turi vieną įėjimą ir vieną išėjimą. Jei įėjime yra loginio vieneto įtampa, išėjime – loginio nulio ir atvirkščiai.

<table>
<tr><td colspan="2">Įėjimo loginis elementas</td><td colspan="2">Įėjimo su inversija loginis elementas</td></tr>
<tr><td colspan="2">  </td><td colspan="2">  </td></tr>
<tr><td>Įėjo nulinis signalas, išėjo nulinis signalas</td><td>Įėjo vienetinis signalas, išėjo vienetinis signalas</td><td>Įėjo nulinis signalas, išėjo vienetinis signalas</td><td>Įėjo vienetinis signalas, išėjo nulinis signalas</td></tr>
<tr><td>  </td><td>  </td><td>  </td><td>  </td></tr>
<tr><td colspan="2">Signalo vertė nesikeičia</td><td colspan="2">Signalas virsta priešingu pradiniam</td></tr>
</table>

Užduotis

Patyrinėkite loginių schemų kūrimo aplinką https://logic.ly/demo/. Jei iš loginių elementų išeinančius signalus norime pakeisti priešingais, tai inverteris žymimas analogiškai vieno elemento inverteriui (mažas skrituliukas). Patikrinkite, kaip pasikeičia išeinantis signalas, lyginant su konjunkcija, disjunkcija, griežtąja disjunkcija.



Logikos dėsniai

Teiginių logikos dėsniai – tai sudėtiniai teiginiai, kurie gali turėti tik teisingumo reikšmę „T“. Užrašant logines jungtis dažniausiai naudojami skliaustai.

Matematinė logika įrodė, kad logikos dėsnių yra be galo daug, tačiau čia aptarsime tik svarbiausius dėsnius.

Logikos dėsnis	**Logikos dėsnio pavadinimas**
(p ∧ q) ≡ (q ∧ p)	Konjunkcijos komutatyvumo
(p ∨ q) ≡ (q ∨ p)	Disjunkcijos komutatyvumo
((p ∧ q) ∧ k) ≡ (p ∧ (q ∧ k))	Konjunkcijos asociatyvumo
((p ∨ q) ∨ k) ≡ (p ∨ (q ∨ k))	Disjunkcijos asociatyvumo
¬(¬p) ≡ p	Dvigubo neigimo
¬(p ∧ q) ≡ (¬q ∨ ¬p)	De Morgano
¬(p ∨ q) ≡ (¬q ∧ ¬p)	De Morgano
(p ∧ (q ∨ k)) ≡ ((p ∧ q) ∨ (p ∧ k))	Distributyvumo

$(p \vee (q \wedge k)) \equiv ((p \vee q) \wedge (p \vee k))$	Distributyvumo

Šie dėsniai labai praverčia, kai norima supaprastinti sudėtinius teiginių sakinius ar logines schemas.

Užduotis

De Morgano dėsnis teigia, kad $\neg(p \wedge q) \equiv (\neg q \vee \neg p)$. Įrodykite.

Sprendimas

Sudarome teisingumo lenteles:

p	q	$p \wedge q$	$\neg(p \wedge q)$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

p	q	$\neg q$	$\neg p$	$(\neg q \vee \neg p)$.
0	0	1	1	1
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

Teisingumo lentelės sutampa ir tai reiškia, kad De Morgano dėsnis teisingas.

Įrodykite kitą De Morgano dėsnį: $\neg(p \vee q) \equiv (\neg q \wedge \neg p)$

Užduotis

Panagrinėkite konjunkcijos, disjunkcijos ir griežtosios disjunkcijos (XOR) loginius elementus ir jų veikimą kitoje aplinkoje: https://logic.modulo-info.ch/

Pavyzdžiui, inversiją atliekantis elementas vaizduojamas  , o jo veikimą paaiškina tokia schema:

Šioje schemoje iš kairės yra įeinančio signalo elementas, o iš dešinės – išeinančio. Šiuo atveju įeina aukštas signalas (žymime „1“), o inverteris pakeičia jį žemu signalu (žymime „0“).

Jei įeina žemas signalas, tai išeina aukštas: 

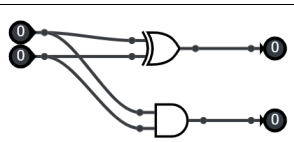
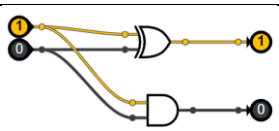
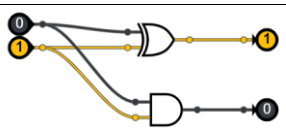
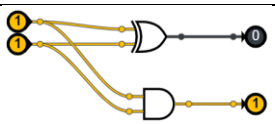
.
Šioje aplinkoje jungiklis pakeistas įprastu įėjimo (angl. *input*) elementu, o lemputė – išėjimo (angl. *output*) elementu.

Pažintis su kompiuterio atliekama sudėties operacija

Kompiuteriai, vykdydami programas, atlieka įvairius veiksmus. Paprasta schema paaiškina, kaip kompiuteris – automatas – gali atlikti aritmetines operacijas, pavyzdžiui, paprastą sudėtį. Kompiuteriuose sudėtis yra kitų aritmetinių funkcijų – atimties, daugybos ir dalybos –

pagrindas. Mikroprocesoriaus aritmetinis-loginis įtaisas atlieka dvejetainę sudėtį. Paprastą loginę schemą, galinčią sudėti du vienaženklius dvejetainius skaičius, galima nesunkiai sukurti naudojant tik du loginius konjunkcijos (AND) ir griežtosios disjunkcijos (XOR) elementus. Tokia schema vadinama pusiniu sumatoriumi (angl. *half-adder*).

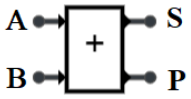
Dvejetainė suma

0 + 0 = 0	1 + 0 = 1	0 + 1 = 1	1 + 1 = 10
			

Viršutinis išėjimo signalas vaizduoja dvejetainį vienetą tik tuomet, kai tik vienas iš įeinančių signalų turi „1“ reikšmę, jis vadinamas suma (žymima „S“). Tai užtikrina XOR elementas.

Apatinis išėjimo signalas vaizduoja dvejetainį vienetą tik tuomet, kai tik abu įeinantys signalai turi „1“ reikšmę, jis vadinamas pernaša „P“ (angl. *carry*). Pernaša perkelia „1“ į aukštesnę skiltį. Tai užtikrina loginis elementas AND. Šiuo atveju sudėjus du dvejetainius vienetus gaunama 10_2.

Ši schema vaizduojama vienu elementu („juodaja dėže“) taip:

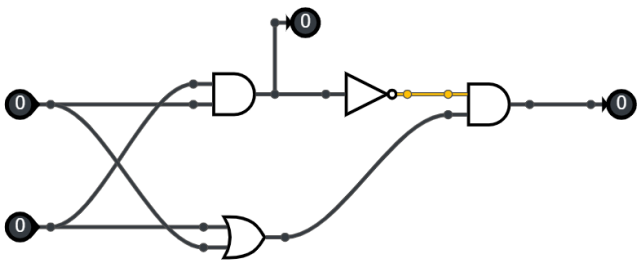


Jos teisingumo lentelė:

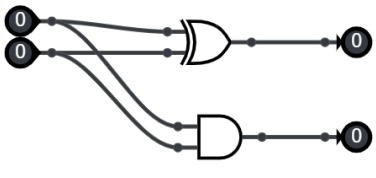
A (įvedimas)	B (įvedimas)	P (išvedimas)	S (išvedimas)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Užduotis

Atverkite https://logic.modulo-info.ch/ aplinką ir sukurkite tokią loginę schemą:

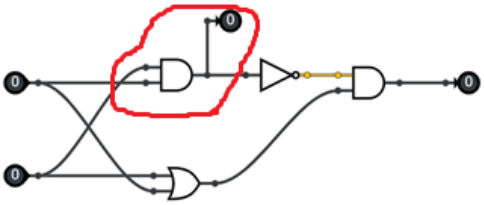


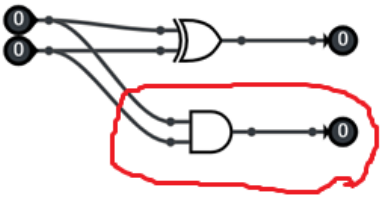
Įrodykite, kad ši schema yra ekvivalenti jau nagrinėtai pusinio sumatoriaus schemai



Sprendimas

Paveiksle raudonai apibrėžtos identiškos dalys – konjunkcija, kai skaičiuojama pernaša:





Pavaizduokime likusių dalių, kurios skaičiuoja sumą, teisingumo lenteles. Jau nagrinėtu atveju teisingumo lentelė sutampa su griežtosios disjunkcijos lentele. Pažingsniui gaukime naujos schemos, skirtos sumai skaičiuoti, dalies teisingumo lentelę.

Pirmiausia užrašykime loginį reiškinį: įėjimo signalų A ir B konjunkcija (A ∧ B), po to jos rezultatas patenka į inverterį (neigimo operacija, **¬(**A ∧ B)). Schemos apatinėje dalyje iš pradžių vyksta A ir B disjunkcija (A ∨ B). Galiausiai skaičiuojame viršutinės dalies ir disjunkcijos rezultato konjunkciją ((**¬(**A ∧ B)) ∧ (A ∨ B)):

A	B	A ∧ B	**¬(**A ∧ B)	A ∨ B	**(¬(**A ∧ B)) ∧ (A ∨ B)	A ⊕ B
0	0	0	1	0	0	0
0	1	0	1	1	1	1
1	0	0	1	1	1	1
1	1	1	0	1	0	0

Teisingumo lentelės sutampa, todėl galime daryti išvadą, kad sudėtinis loginis sakinys

(¬(A ∧ B)) ∧ (A ∨ B) yra ekvivalentus griežtajai disjunkcijai A ⊕ B.

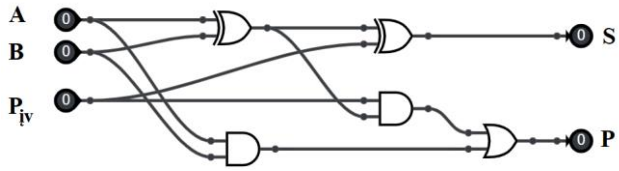
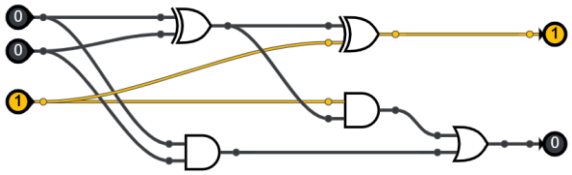
Būlio algebroje visiškai pakanka trijų loginių operacijų (neigimo, konjunkcijos ir disjunkcijos) įvairiems sudėtiniams sakiniams sudaryti ir kurti logines schemas.

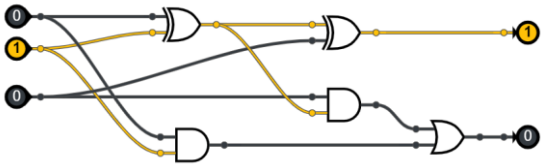
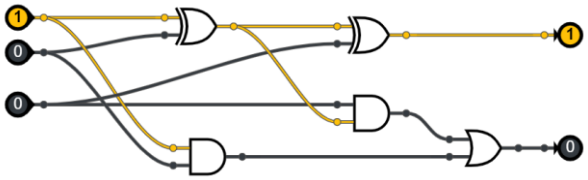
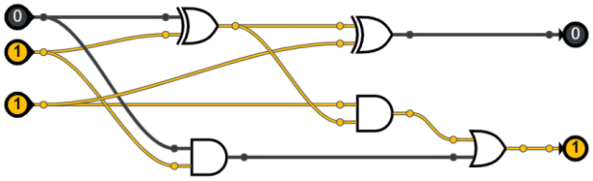
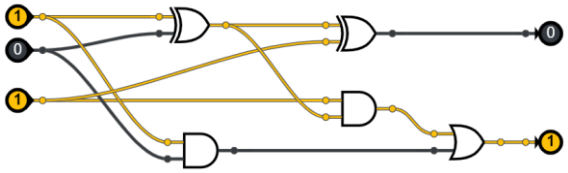
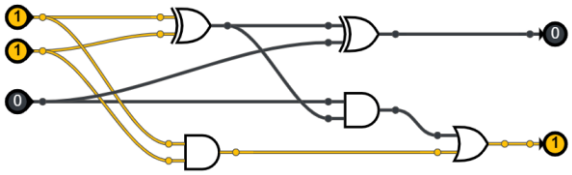
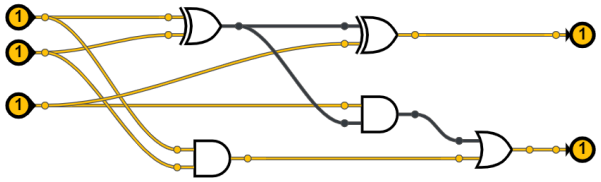
Sumatorius

Schema, leidžianti sudėti tik du vienaženklius dvejetainius skaičius, yra labai riboto naudojimo.

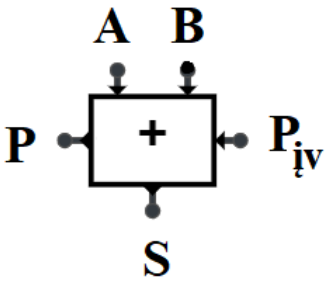
Norint sudėti daugiaženklius dvejetainius skaičius, reikia išspręsti žemesnėje skiltyje esančio bito perkėlimo problemą.

Tam naudojama sumatoriaus schema, kuri sudeda du bitus, gauna sumą (S), jei reikia, atlieka perkėlimą (P) į aukštesnę skiltį, priima ankstesnės operacijos perkėlimą ($P_{įv}$) ir gauna teisingą atsakymą.

0 + 0 + 0 = 0	0 + 0 + 1 = 01
	

0 + 1 + 0 = 01	1 + 0 + 0 = 01
	
0 + 1 + 1 = 10	1 + 0 + 1 = 10
	
1 + 1 + 0 = 10	1 + 1 + 1 = 11
	

Sumatoriaus schema dar vaizduojama tokiu vienu elementu („juodaja dėže“):



Sumatoriaus teisingumo lentelė

A (įvedimas)	B (įvedimas)	Pįv (pernašos bitas)	P (išvedimas)	S (išvedimas)
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
1	0	0	0	1

A (įvedimas)	B (įvedimas)	Pįv (pernašos bitas)	P (išvedimas)	S (išvedimas)
0	1	1	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Dvejetainių skaičių sudėčiai reikia sujungti tiek sumatorių, kiek skilčių turi sudedami skaičiai. Pavyzdžiui, 32 bitų procesoriui skaičiams sudėti reikia 32 sumatorių.
Keturių skilčių sudėtis (naudojami 4 sumatoriai) atrodo taip:

P_4	$\leftarrow P_3$	$\leftarrow P_2$	$\leftarrow P_1$	$\leftarrow P_0$	$\leftarrow P$	pernaša į aukštesnę skiltį
	A_3	A_2	A_1	A_0	A	pirmas dėmuo
	B_3	B_2	B_1	B_0	B	antras dėmuo
	S_3	S_2	S_1	S_0	S	suma

Trigeris

Trigeris yra atminties elementas, turintis dvi stabilias būsenas. Trigeris paprastai sudarytas iš dviejų tranzistorių, vienas kitą veikiančių grįžtamojo ryšiu pagrindu. Trigerio būseną gali pakeisti poveikis iš išorės (impulsas). Būsena išlieka stabili, kol tiekiamas maitinimas.

Trigeris naudojamas duomenims ir būsenoms įsiminti procesoriaus registruose ir kitose schemose.

Iš trigerių sudaromi skaitikliai, registrai ir kitos skaitmeninės elektroninės schemos. Jose trigerių būna šimtai ar net tūkstančiai, todėl šios schemos vadinamos integrinėmis. Trigeriai gaminami ir kaip atskiros integrinės schemos, kartais po du ar daugiau viename kristale.

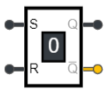
Trigerių gaminama įvairių. Panagrinėkime paprasčiausią RS trigerį.

Integrinis asinchroninis RS trigeris

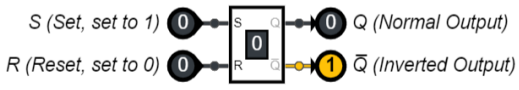
Integriniai RS trigeriai sudaromi naudojant standartinius dviejų įėjimų loginius disjunkcijos su inversja  elementus (jie dažnai vadinami „ARBA-NE“ elementais) arba konjunkcijos su inversija  elementus (dažnai jie vadinami „IR-NE“ elementais). Vienas įėjimas naudojamas grįžtamajam ryšiui, o į kitą perduodamas valdymo signalas.

Trigeris turi du atskirus įėjimus. Vienas iš jų vadinamas nustatymo (įrašymo) įėjimu (angl. *set*), kitas – atkūrimo (ištrynimo) įėjimu (angl. *reset*). Pagal anglų k. įėjimų pavadinimų pirmąsias raides S ir R šis trigeris vadinamas RS trigeriu.

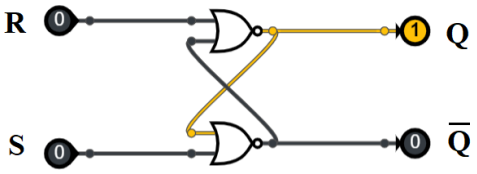
Trigeris turi du išėjimus: tiesioginį ir inversinį. Tiesioginis išėjimas dažniausiai žymimas simboliu Q (angl. *normal output*), o inversinis simboliu $\overline{Q}$ (angl. *inverted output*). Tiesioginis išėjimas Q yra tas, kuriame gaunamas įėjimo S signalas.

„Juodaja dėže“ pavaizduotas trigeris atrodo taip: 

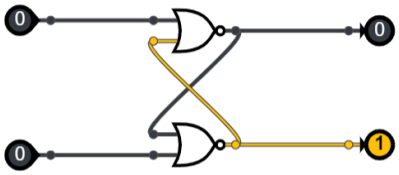
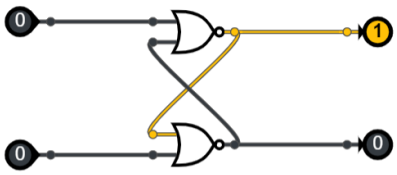
, o įjungtas:



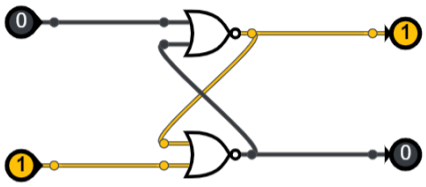
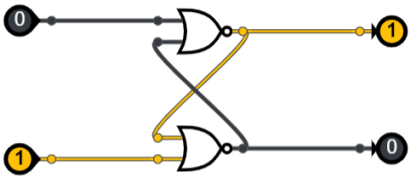
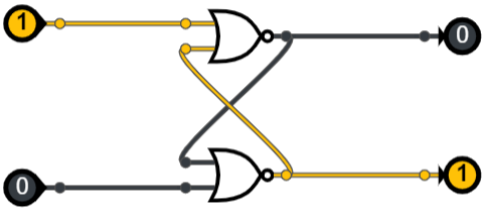
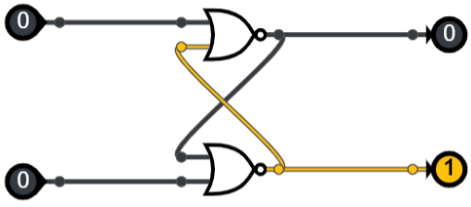
Trigerio loginė schema (naudojant https://logic.modulo-info.ch/ aplinką):



Į trigerį perdavus neutralųjį signalų rinkinį R = 0 ir S = 0, trigeris lieka vienoje iš stabilių būsenų, nes abu įėjimo signalai yra pasyvūs.

Abiejų būsenų tikimybė vienoda	
	

Perdavus signalų rinkinį R = 0 ir S = 1, trigeris persijungia į vieneto būseną (Q = 1) ir ją įsimena

	
Perdavus R = 1 ir S = 0, trigeris persijungia į nulio būseną ir ją įsimena	
	

Signalų rinkinys R = 1 ir S = 1 draudžiamas, nes gali perjungti trigerį į nestabilią būseną.

Trigeris, sudarytas iš dviejų konjunkcijos su inversija (IR-NE) elementų, veikia kitaip.

Daugiau apie trigerius galima susipažinti Antano Savicko leidinyje „Aviacinės elektronikos pagrindai“ (http://dspace.vgtu.lt/bitstream/1/1401/1/1348-S_Savickas_Aviacines_WEB.pdf).

Kadangi elemento IR-NE išėjime yra įtampa (vienetas), jei nors į vieną įėjimą perduotas nulis (o ne vienetas, kaip ARBA-NE elementuose), šio trigerio R ir S įėjimai bus inversiniai. Į vieneto būseną (Q = 1) trigeris persijungia perdavus S = 0, o į nulio būseną – perdavus R =

0. Signalų rinkinys R = 1 ir S = 1 yra neutralus, o draudžiamasis rinkinys R = 0 ir S = 0 perjungia trigerį į nestabilią būseną.

Logikos naudojimas skaičiuoklėse

Skaičiuoklės yra labai galingos programos, kurios leidžia su duomenimis atlikti įvairius veiksmus.

Skaičiuoklės turi ir loginių funkcijų, kurios atlieka logines operacijas su duomenis, ir loginių duomenų.

Panagrinėkime keletą paprastų „Excel“ skaičiuoklės pavyzdžių: konjunkcijos, disjunkcijos ir griežtosios disjunkcijos teisingumo lentelių sudarymą.

Užrašome galimas loginių kintamųjų reikšmes ir pasinaudojame funkcijomis. Paveiksle pavaizduota, kaip lietuviškai užrašomos funkcijos. (Angliškai rašant vietoje kabliataškių vartojami kableliai.)

3	a	b	AND	OR	XOR
4	0	0	=AND(B4;C4)	=OR(B4;C4)	=XOR(B4;C4)
5	0	1	=AND(B5;C5)	=OR(B5;C5)	=XOR(B5;C5)
6	1	0	=AND(B6;C6)	=OR(B6;C6)	=XOR(B6;C6)
7	1	1	=AND(B7;C7)	=OR(B7;C7)	=XOR(B7;C7)

Galima kintamuosius rašyti skaitmenimis, galima žodžiais:

3	a	b	AND	OR	XOR
4	0	0	FALSE	FALSE	FALSE
5	0	1	FALSE	TRUE	TRUE
6	1	0	FALSE	TRUE	TRUE
7	1	1	TRUE	TRUE	FALSE

a	b	AND	OR	XOR
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	FALSE	TRUE	TRUE
FALSE	FALSE	FALSE	FALSE	FALSE

Loginės funkcijos dažnai naudojamos sąlygos (šakojimo) uždaviniuose.
Pavyzdžiui, duotos trys trikampio kraštinės. Reikia nustatyti, ar trikampis yra lygiakraštis, įvairiakraštis, lygiašonis:

	B	C	D	E	F
1	**b**	**c**	**Trikampis**	**Trikampis**	**Trikampis**
2	4	4	lygiakraštis	ne įvairiakraštis	ne statusis
3	8	9	ne lygiakraštis	įvairiakraštis	ne statusis
4	3	3	lygiakraštis	ne įvairiakraštis	ne statusis
5	4	5	ne lygiakraštis	įvairiakraštis	statusis
6	8	10	ne lygiakraštis	įvairiakraštis	statusis

Pirmuoju atveju naudojame konjunkciją, nes visos trys kraštinės turi būti tarpusavyje lygios. Antruoju atveju taip pat naudojame konjunkciją, nes visos trys kraštinės turi būti tarpusavyje skirtingos. Šie atvejai surašyti atskiruose stulpeliuose, nes dar gali būti ir lygiašonis trikampis. Trečiuoju atveju bet kuri kraštinė gali būti įžambinė, todėl naudojama disjunkcija ir išrašomi visi sąryšiai tarp kraštinių, patenkinantys Pitagoro formulę.

	A	B	C	D	E	F
1	a	b	c	Trikampis	Trikampis	Trikampis
2	4	4	4	=IF(AND((A2=B2);(B2=C2));"lygiakraštis";"ne lygiakraštis")	=IF(AND((A2<>B2);(B2<>C2);(A2<>C2));"įvairiakraštis";"ne įvairiakraštis")	=IF(OR((A2*A2+B2*B2=C2*C2);(A2*A2+C2*CB2=BC2*B2);(C2*C2+B2*B2=A2*A2));"statusis";"ne statusis")
3	5	8	9	=IF(AND((A3=B3);(B3=C3));"lygiakraštis";"ne lygiakraštis")	=IF(AND((A3<>B3);(B3<>C3);(A3<>C3));"įvairiakraštis";"ne įvairiakraštis")	=IF(OR((A3*A3+B3*B3=C3*C3);(A3*A3+C3*CB3=BC3*B3);(C3*C3+B3*B3=A3*A3));"statusis";"ne statusis")
4	3	3	3	=IF(AND((A4=B4);(B4=C4));"lygiakraštis";"ne lygiakraštis")	=IF(AND((A4<>B4);(B4<>C4);(A4<>C4));"įvairiakraštis";"ne įvairiakraštis")	=IF(OR((A4*A4+B4*B4=C4*C4);(A4*A4+C4*CB4=BC4*B4);(C4*C4+B4*B4=A4*A4));"statusis";"ne statusis")
5	3	4	5	=IF(AND((A5=B5);(B5=C5));"lygiakraštis";"ne lygiakraštis")	=IF(AND((A5<>B5);(B5<>C5);(A5<>C5));"įvairiakraštis";"ne įvairiakraštis")	=IF(OR((A5*A5+B5*B5=C5*C5);(A5*A5+C5*CB5=BC5*B5);(C5*C5+B5*B5=A5*A5));"statusis";"ne statusis")
6	6	8	10	=IF(AND((A6=B6);(B6=C6));"lygiakraštis";"ne lygiakraštis")	=IF(AND((A6<>B6);(B6<>C6);(A6<>C6));"įvairiakraštis";"ne įvairiakraštis")	=IF(OR((A6*A6+B6*B6=C6*C6);(A6*A6+C6*CB6=BC6*B6);(C6*C6+B6*B6=A6*A6));"statusis";"ne statusis")

Užduotys

1. Papildykite lentelę dar vienu atveju, kai dvi bet kurios kraštinės yra lygios, ir sukurkite dar vieną papildomą formulę, kuri nustatytų ir parašytų „lygiašonis“ arba „nelygiašonis“.
2. Sukurkite lentelę, kurioje būtų pavaizduojami trijų atkarpų įvairūs ilgiai (10–12 atvejų). Parašykite gretimame langelyje formulę, kuri kiekvienu atveju užrašytų, ar iš šių atkarpų galima sudaryti trikampį.

Užduotis „Uogienė“

Petras, Jonas ir Onutė pasiliko namuose vieni. Grįžusi mama pastebėjo, kad vaikai suvalgė uogienę. Į jos klausimą, kas tai padarė, vaikai atsakė:

Petras: „Aš nevalgiau. Marytė taip pat nevalgė“.
Jonas: „Onutė tikrai nevalgė. Petras suvalgė“.
Onutė: „Jonas meluoja. Tai jis suvalgė“.

Parenkite vaikų teiginių teisingumo lentelę ir atsakykite, kas suvalgė uogienę.

Žinoma, kad du vaikai mamai nemelavo, o vienas – pamelavo vieną kartą. Geltona spalva pažymėtus langelius užpildykite tinkamomis formulėmis ir funkcijomis. Padarykite teisingą išvadą.

	A	B	C	D	E	F	G	H	I
1	Kas suvalgė uogienę								
2	Petras valgė	Jonas valgė	Onutė valgė	Petro teiginiai		Jono teiginiai		Onutės teiginiai	
3				Aš nevalgiau	Onutė taip pat nevalgė	Onutė tikrai nevalgė	Petras suvalgė	Jonas meluoja	Tai jis (Jonas) suvalgė
4	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE
5	FALSE	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE
6	FALSE	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE
7	FALSE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE
8	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE	FALSE
9	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE
10	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE
11	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
12									
13				Išvada					
14									

Tik 6-oje eilutėje yra viena netiesa, todėl uogienę suvalgė Jonas.
Galima tris pirmuosius stulpelius užpildyti nuliais ir vienetais.

Atsakymas

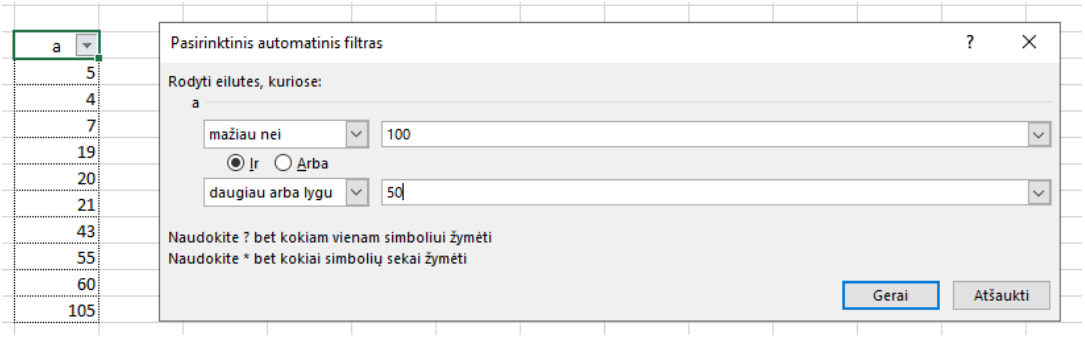
Kas suvalgė uogienę

Petras valgė	Jonas valgė	Onutė valgė	Petro teiginiai		Jono teiginiai		Onutės teiginiai	
			Aš nevalgiau	Onutė taip pat nevalgė	Onutė tikrai nevalgė	Petras suvalgė	Jonas meluoja	Tai jis (Jonas) suvalgė
FALSE	FALSE	FALSE	=NOT(A4)	=NOT(C4)	=NOT(C4)	=A4	=NOT(AND(F4;G4))	=B4
FALSE	FALSE	TRUE	=NOT(A5)	=NOT(C5)	=NOT(C5)	=A5	=NOT(AND(F5;G5))	=B5
FALSE	TRUE	FALSE	=NOT(A6)	=NOT(C6)	=NOT(C6)	=A6	=NOT(AND(F6;G6))	=B6
FALSE	TRUE	TRUE	=NOT(A7)	=NOT(C7)	=NOT(C7)	=A7	=NOT(AND(F7;G7))	=B7
TRUE	FALSE	FALSE	=NOT(A8)	=NOT(C8)	=NOT(C8)	=A8	=NOT(AND(F8;G8))	=B8
TRUE	FALSE	TRUE	=NOT(A9)	=NOT(C9)	=NOT(C9)	=A9	=NOT(AND(F9;G9))	=B9
TRUE	TRUE	FALSE	=NOT(A10)	=NOT(C10)	=NOT(C10)	=A10	=NOT(AND(F10;G10))	=B10
TRUE	TRUE	TRUE	=NOT(A11)	=NOT(C11)	=NOT(C11)	=A11	=NOT(AND(F11;G11))	=B11

Išvada	

Duomenų filtravimas skaičiuoklėje

Skaičiuoklėje labai patogūs automatiniai filtrai, kurie greitai atrenka reikalingus duomenis. Yra galimybė panaudoti teiginius jungiant juos konjunkcijos (IR) ar disjunkcijos (ARBA) jungtimis.



Užduotis

Panagrinėkite įvairius atvejus, kurie galimi naudojant skaičiuoklės automatinį filtrą. Kuriuos duomenis tvarkant galima pasinaudoti konkrečiomis siūlomomis galimybėmis?

Registrų centras yra paskelbęs Lietuvos gyventojų skaičių pagal savivaldybes 2022 m. sausio 1 d.
(https://www.registrucentras.lt/bylos/dokumentai/gr/2022_01_05_Gyventoju_skaicius_pagal_savivaldybes.xls)

Eil. Nr.	Savivaldybės pavadinimas	Bendras gyventojų skaičius
1	Akmenės r. sav.	20597
2	Alytaus m. sav.	53920
3	Alytaus r. sav.	28170
4	Anykščių r. sav.	24619
5	Birštono sav.	4425
6	Biržų r. sav.	25141
7	Druskininkų sav.	21282
8	Elektrėnų sav.	25903
9	Ignalinos r. sav.	15495
10	Jonavos r. sav.	43564
11	Joniškio r. sav.	22234
12	Jurbarko r. sav.	27145
13	Kaišiadorių r. sav.	29746
14	Kalvarijos sav.	10737
15	Kauno m. sav.	313503
16	Kauno r. sav.	105032
17	Kazlų Rūdos sav.	11621
18	Kelmės r. sav.	27513
19	Klaipėdos m. sav.	165710
20	Klaipėdos r. sav.	67232
21	Kretingos r. sav.	39922
22	Kupiškio r. sav.	17425
23	Kėdainių r. sav.	49360
24	Lazdijų r. sav.	19579
25	Marijampolės sav.	57937

Atsisiųskite šį skaičiuoklės failą, ištrinkite nereikalingą informaciją ir pasinaudodami filtro loginėmis funkcijomis, suraskite, kiek Lietuvoje 2022 m. sausio 1 d. buvo savivaldybių, kuriose gyveno:

a) mažiau nei 10000 gyventojų,
b) daugiau arba lygiai 10000 gyventojų, bet mažiau nei 40000,
c) daugiau arba lygiai 40000 gyventojų, bet mažiau nei 100000,
d) daugiau arba lygiai 100000 gyventojų.

Savivaldybių yra 60, paveiksle pateikiamas tik fragmentas.

Literatūra

1. Anzenbacher, A. Filosofijos įvadas. – Vilnius, „Katalikų pasaulis“, 1992. – 344 p.
2. Bubelis, R., Jakimenko, V., Logika. I dalis. – Vilnius, Mykolo Romerio universitetas, 2012. – 224 p.
3. Russell, S., Norvig, P. Artificial Intelligence. A modern Approach, Pearson, 2016. – 1132 p.
4. Savickas, A. Aviacinės elektronikos pagrindai. – Vilnius, Technika, 2012. – 112 p. http://dspace.vgtu.lt/bitstream/1/1401/1/1348-S_Savickas_Aviacines_WEB.pdf (žiūrėta 2022-10-26).
5. Dagienė, V., Jevsikova, T. Aiškinamasis kompiuterijos terminų žodynas. – Vilnius, Vilniaus universitetas, 2016. – 484 p.
6. Pellet, J.-P., Logic-Circuit-Simulator, https://logic.modulo-info.ch/ (žiūrėta 2022-10-26).
7. Loginiai elementai ir loginės schemos. Logicly, https://logic.ly/ (žiūrėta 2022-10-26).