Robotikos vadovėlis popamokinėms veikloms

Parengė
Artūras Gaulia

LEIDYKLA SPAUSTUVĖ Aukuras
Utena, 2022

Vadovėlis parengtas įgyvendinant 2014–2020 m. Interreg V-A Latvijos ir Lietuvos bendradarbiavimo per sieną programos finansuojamą projektą LLI-542 „IT programavimo ir robotikos kompetencijų plėtra pasienio regionuose Latgalos, Visagino ir Ignalinos mokyklose“ („RoboNet“). Projektą taip pat remia Ignalinos, Zarasų rajonų bei Visagino savivaldybės.

Projektą įgyvendina Latgalės regiono plėtros agentūra ir VšĮ „Euroregiono Ežerų krašto direktorato biuras“.

Visas projekto biudžetas – 369 tūkst. Eur. Iš jų bendrasis Europos regioninės plėtros fondo finansavimas – 314 tūkst. Eur.

Tai – bandomasis projektas, skirtas regionų mokyklose padėti įsteigti robotikos klases, parengti mokytojus, aprūpinti juos reikiama įranga, konstravimo detalėmis. Penkiose Ignalinos ir Zarasų rajonų bei Visagino savivaldybių mokyklose įkurti ir veikia robotikos būreliai, kuriuos lanko ne mažiau kaip 150 vaikų ir jaunuolių. 15 regiono pedagogų parengti vesti būrelius pagal naujausią robotikos metodiką. Vaikus motyvuoti organizuojamos robotikos varžybos tarp Latgalės ir Ežerų krašto regionams atstovaujančiųjų, patys geriausieji turės galimybę jėgas išbandyti tarptautinėse robotikos varžybose vienoje iš Baltijos valstybių sostinių vykstančiame kasmetiniame robotikos renginyje.

Už šio vadovėlio turinį atsako VšĮ „Euroregiono Ežerų krašto direktorato biuras“. Jokiomis aplinkybėmis negali būti laikoma, kad jis atspindi Europos Sąjungos nuomonę.

Bibliografinė informacija pateikiama Lietuvos integralios bibliotekų informacinės sistemos (LIBIS) portale ibiblioteka.lt

ISBN 978-609-455-608-1

Prologas

Šis vadovėlis parengtas VšĮ „Euroregiono Ežerų kraštas direktorato biuro“ užsakymu. Projektas „IT programavimo ir robotikos kompetencijų plėtra pasienio regionuose Latgalos, Visagino ir Ignalinos mokyklose“ dalinai finansuojamas Latvijos ir Lietuvos bendradarbiavimo per sieną 2014–2020 m. programos. Vadovėlis – tai metodinė priemonė, papildanti to paties projekto mokymo kursus „Specializuoti IT programavimo ir robotikos mokymai“.

Leidinys sudarytas iš septynių dalių. Visas šias dalis jungia viena disciplina – robotika. Kadangi vadovėlis skirtas mokykloms, jame pateikiama sutrumpintos ir supaprastintos robotikos sudedamos disciplinos. Kiekvienoje dalyje pateikiami esminiai pagrindai, būtini kuriant robotus. Visas dalis galime suskirstyti į grupes: teorinę, techninę ir praktinę.

Teorinėje dalyje, kuriai priklauso įvadas ir LEGO Education mokymosi metodikos, pateikiamos STEM mokymosi istorija ir metodikos. Šios metodikos paremtos konstrukcionizmo ir konstruktyvumo teorijomis, daugiau nei 30 metų naudojamos įvairiose pasaulio šalių mokyklose ar kitose mokymo įstaigose.

Techninę dalį sudaro trys skyriai: elektronika, programavimas ir 3D modeliavimas. Šiuose skyriuose atrinkta tik dažniausiai naudojama ir moksleiviams suprantama techninė informacija. Elektronikos dalyje apžvelgiami dažniausiai naudojami elektronikos komponentai ir pagrindiniai dėsniai, reikalingi nesudėtingoms schemoms kurti. Programavimo pagrindai pateikti C kalboje, nes ši kalba naudojama Arduino mikrovaldiklių programavime, kurie plačiai paplitę mokyklose dėl savo paprastumo, plataus suderinamumo ir pakankami mažos kainos. 3D modeliavimo skyriuje trumpai paaiškinta 3D spausdinimo technologija ir 3D objektų kūrimas ThinkerCad aplinkoje. Ši programa pasirinkta dėl kelių priežasčių. Pirma, tai viena iš nedaugelio programų, kurios yra nemokamos. Antra, ši programa pritaikyta dirbti mokytojams su klase bei darbo aplinka yra intuityvi ir nesudėtinga moksleiviams.

Praktinė dalis jungia teorija su technine dalimi, susiedama jas mokymo priemonėmis ir pamokų pavyzdžiais. Robotų aprašymo skyriuje pateikiami dažniausiai mokyklose naudojamų robotų aprašymai ir jų techninės charakteristikos bei reikalavimai. Paskutiniame vadovėlio skyriuje pateikiami pamokų pavydžiai kiekvienam robotų tipui.

Robotika – sudėtinga disciplina, tačiau mokymąsi galima paversi smagiu naudojant interaktyvias priemones ir įtraukiančias metodikas. Taip galima pažadinti moksleivių smalsumą ir sukurti savimotyvacijos srautą.

Didžiausias mokytojo pasiekimas yra sugebėjimas pažadinti mokinio kūrybiškumą ir žingeidumą.

A. Einšteinas

Turinys

I. ĮVADAS

Mokymosi teorijos (konstruktyvizmas ir konstrukcionizmas)

Keičiasi laikai, keičiasi visuomenė, keičiasi ir vaikų švietimas. Nuo industrinės visuomenės sukama link kūrybiškos visuomenės, kur kūrybiškumas yra gyvybiškai svarbus visuomenės gyvavimui ir vystymuisi. XXI amžiuje pagrindiniai žmogiškieji ištekliai yra kūrybinis potencialas ir gebėjimas mokytis. Jeigu praėjusiuose amžiuose pakakdavo vienos profesijos išgyventi suaugus, tai šiame amžiuje jau nepakaks vienos kompetencijos, nes niekas nežino, kokioje visuomenėje augs dabartiniai vaikai, kokios technologijos bus sukurtos, o gal ir toliau pildysis mokslinė fantastika ir tapsime biorobotais. Tik viena aišku, kad kūrybiškumas, kompleksinių problemų sprendimo ir komandinio darbo gebėjimai bus kertiniai. Remiantis šveicarų psichologo Žano Pjažė (Jean Piaget) (1896–1980) moksliniais tyrimais, mes nebegalime žiūrėti į vaiką kaip į tuščią indą, į kurį suaugusieji pila žinias. Pjažė įrodė, kad vaikai nuo pat gimimo aktyviai mokosi iš savo patirties. Mūsų metodiką pristatysime pirmiausia aptardami konstrukcionizmo teoriją, po to mokymosi aplinką ir mokymosi priemones.

Konstrukcionizmo teoriją išplėtojo Seymouras Papertas (Seymour Papert) iš Masačusetso technologijos instituto (MIT). Ji sukurta remiantis šveicarų psichologo Žano Pjažė *pažinimo teorija* (theory of knowledge). S. Papertas dirbo kartu su Ž. Pjažė 1950–1960 metais.

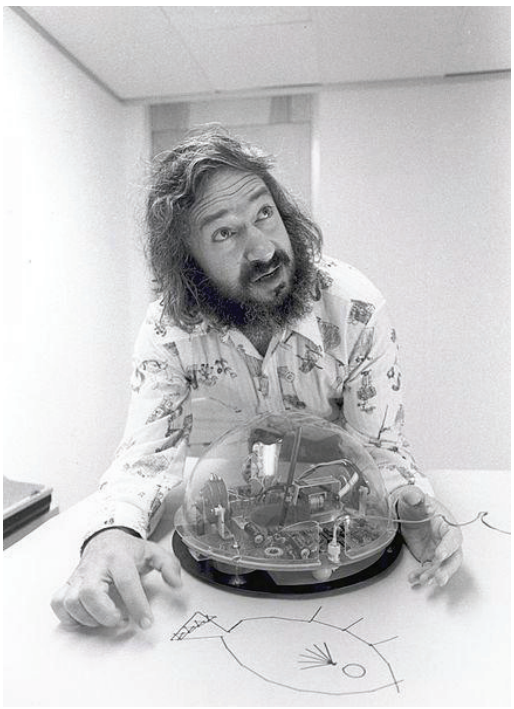
Pažinimo teorija – idėjų visuma, kuri bando paaiškinti žmogaus pažinimo procesą ir jo vystymąsi. Pjažė teorija teigia, kad žinios konstruojamos. Vaikui augant ir mokantis konstruojamos vis stabilesnės ir sudėtingesnės žinių struktūros, kuriomis naudodamasis vaikas interpretuoja pasaulį ir išplečia savo eksperimentų lauko ribas. Dėl šios priežasties jis pavadino savo teoriją *konstruktyvizmu*. Pjažė tikslas buvo suprasti, kaip vaikai konstruoja žinias. Jis nelaikė savęs pedagogu, jis vadino save eksperimentuotoju, o S. Papertas norėjo pritaikyti Pjažė teoriją ugdant vaikus.

Prieš Pjažė vyravo keletas nuostatų, kaip vaikai įgyja žinių. Viena iš jų teigia, kad žinios yra įgimtos, tuomet mokymas yra esamų žinių panaudojimas pagal pateiktas instrukcijas ar užduotus klausimus. Kita, kad žinios kuriamos iš patirties, tuomet mokymas vyksta vaikams įgyjant patirties, parodant, kaip reikia daryti, pateikiant teisingus atsakymus.

Konstrukcionizmo teorija nagrinėja dviejų tipų konstravimą: kai vaikai konstruodami išorinio pasaulio objektus konstruoja ir žinias savo galvose. Vaikas mokosi ir taip

konstruojamos vis sudėtingesnės žinių struktūros, išmokstama vis sudėtingesnių dalykų, įgyjama daugiau žinių. Taip savarankiško mokymosi ir tobulėjimo ciklas ugdo mokinį.

Apie *konstrukcionizmą* S. Papertas pradėjo galvoti 1960-aisiais. Jis stebėjo moksleivių grupę, kūrusią muilo skulptūras per dailės pamokas, visi mokiniai buvo labai įsitraukę į šį procesą. Tuomet jam ir kilo klausimas, kodėl matematikos pamokos nebuvo tokios įdomios ir įtraukiančios, kaip šios meno pamokos. Daugumoje matematikos pamokų moksleiviams pateikiama užduočių sprendimo technika arba matematinis įrodymo modelis. Tuomet vaikams pateikiama užduotis, kurią jie turi išspręsti pagal pateiktus sprendimo kelius. Tokio tipo mokymas vadinamas instrukciniu. O daugelyje meno pamokų vaikai dalyvauja kaip kūrėjai, jie kuria ką nors prasmingo sau, nors visi moksleiviai klasėje naudoja tas pačias priemones, dirba toje pačioje aplinkoje ir kuria tą patį objektą. Vaizduotė ir kūrybiškumas sukuria unikalų produktą, kuris parodo jo, kūrėjo, indėlį. Tai nereiškia, kad instrukcinis mokymas yra blogai, tiesiog tai yra kaip vaistai, jo reikia tam tikro kiekio ir tam tikru laiku, tik tada tai būna naudinga.



Papertui stebint ir mąstant apie muilo skulptūrų klasės sėkmę gimė idėja apie *konstrukcionistinio* matematikos mokymo modelio sukūrimą. 1970 metais kartu su savo kolegomis iš MIT jis sukūrė kompiuterinę programavimo kalbą *LOGO*, kuri suteikia galimybę vaikams mokytis matematikos, kompiuteriu kuriant paveikslėlius, animacijas, muziką ir žaidimus. Vėliau, 1980-aisiais, Paperto komanda sukuria programavimo *„LEGO TC Logo“* aplinką, kuri sujungia prieš tai sukurtą *LOGO* programavimo kalbą ir *LEGO* konstravimo elementus. *„LEGO TC Logo“* suteikia vaikams galimybę kurti, statyti ir konstruoti įvairias struktūras ir mechanizmus iš LEGO kaladėlių ir elektromechaninių komponentų, ir po to jas programuoti LOGO programavimo kalba. Vaikų sukurtos konstrukcijos programuojamos, kad jos galėtų važinėti arba vaikščioti, junginėti šviesas ar reaguoti į įvairius aplinkos pokyčius. Tokie mechanizmai gali būti jungiami ar sąveikauti tarpusavyje nuo paprastų iki sudėtingiausių konstrukcijų. Kartu su *„LEGO TC Logo“* vaikai įtraukiami į trijų tipų *konstrukcionizmą*:

1. Struktūrų konstravimas iš LEGO elementų;
2. Kompiuterinių programų kūrimas, algoritmų konstravimas;
3. Žinių / žinojimo ir suvokimo konstravimas iš patirties.

Tinkamos konstravimo priemonės yra viena iš svarbių *konstrukcionistinio* mokymo dalių, tačiau tai dar ne viskas. Lygiai taip pat yra svarbu tinkama mokymosi aplinka arba socialinis kontekstas, kur yra mokomasi. Gera mokymosi aplinka turi atitikti tris pagrindinius reikalavimus: *pasirinkimo, įvairovės ir draugiškumo.*

Pagal *konstrukcionizmo* teoriją mokymasis yra efektyvus, kai moksleiviai kuria sau asmeniškai prasmingus dalykus – tuos, kurie jiems tikrai rūpi. Tačiau vienas asmuo negali diktuoti, kas yra prasminga kitam asmeniui. Tai ir yra *pasirinkimas*. Kuo didesnė pasirinkimo laisvė, tuo didesnė asmeninio įsitraukimo ir nuosavo indėlio į atliekamą užduotį tikimybė, tuo pačiu stiprėja sąsaja su mokymo turiniu ir jau įgytų žinių sujungimu. Būtent tai ir turėjo omeny Pjažė sakydamas *„žinių asimiliacija“*. Asmeniškai prasmingas besimokančiojo įsitraukimas suteikia prasmę įgytoms žinioms, jos ilgai išlieka.

Įvairovė yra irgi labai svarbi mokymosi aplinkos savybė dėl dviejų dalykų: įgūdžių ir stiliaus įvairovės. Turtingoje mokymo priemonių mokymosi aplinkoje gali mokytis skirtingą patirtį turintys ir skirtingo amžiaus vaikai, nuo pradedančiojo iki eksperto. Tai sudaro sąlygas vienoje klasėje mokyti skirtingo amžiaus ir lygio grupes. Tokios aplinkos vienas iš privalumų – vaikai su mažesne patirtimi turi galimybę mokytis iš didesnę patirtį turinčių ar vyresnių vaikų. O vaikai su didesne patirtimi tobulina savo įgūdžius ir žinias padėdami kitiems. Stilių įvairovė – kai nėra teisingo ar neteisingo kelio, kiekvienas būdas ar sprendimas tinkami, jeigu tai prasminga kūrėjui.

SEQ Figure * ARABIC 3 pav. LEGO EV3 ir STEM

Gera mokymosi aplinka turėtų būti *draugiška*, maloni ir patraukli besimokančiajam. Draugiška mokymosi aplinka tuomet tampa ne tik mokymosi vieta, tačiau ir bendravimo erdve, kurioje renkasi bendraminčiai dėl bendrų tikslų. Taip džiaugsmai ir kartais net nusivylimai tampa neatsiejama *konstrukcionistinio* mokymo dalimi, kai mokymosi patirtimi dalijamasi su bendraminčiais, kuriems patinka tie patys ar panašūs dalykai.

Robotika švietime

Vienas iš švietimo technologijų novatorių buvo Seymouras Papertas, konstrukcionizmo teorijos vienas iš kūrėjų. Nors konstrukcionizmas daugiausia naudojamas gamtos mokslų, informacinių technologijų ir matematikos pamokose, tačiau pastaruoju metu tai sėkmingai plinta ir į kitus dalykus. Viena iš šiuo metu labiausiai išvystytų sričių yra robotika neformaliame ugdyme ir integruota į STEM pamokas.

Robotikos mokymo priemonės yra skirstomos į du tipus: atvira platforma („white-box“), kurioje konstruojama ar kuriama pati technologija ir uždara platforma („black-box“), kur technologija jau sukurta ir jos negali keisti. „Black-box“ robotus galima tik programuoti, kad jie atliktų tam tikrus veiksmus arba užduotis, nesiaiškinant jų konstrukcijos ir veikimo

principų. „White-box“ robotus galima ne tik programuoti, bet ir keisti, modifikuoti, tobulinti, taip susipažinti su jų konstrukcija ir veikimo principais. Pavyzdžiui, „Black-box“ robotas – „Pico – Crickets kits“, o „White-box“ – „Lego Mindstorms EV3“, kuriame robotą kuria ir programuoja patys vaikai. Čia yra pateikiama ne tik programavimo aplinka, bet ir konstravimo įrankiai, ir priemonės. Vienas iš „White-box“ edukacinių robotų trūkumų – pasiekus tam tikrą lygį arba įgijus bazinių žinių, žengiant toliau sudėtingumo lygis kyla tiek programine, tiek konstrukcine prasme. Antra, pradedant mokymą nuo roboto konstravimo, reikalauja daugiau laiko pasiruošimui ir užduočiai atlikti. Be to, vaikų susidomėjimas roboto konstravimu ir jo valdymu gali būti didesnis nei keliami pamokos tikslai. Kita vertus, šis susidomėjimas gali būti ir naudingas įtraukiant į pamokas papildomus uždavinius, susijusius su atitinkamomis konstrukcijomis ar jų elementais. Tačiau, jeigu kalbama apie robotiką, konstravimas turi būti neatsiejama robotų programavimo dalis, bent jau pradiniame etape, kol vaikai susipažįsta su robotų konstrukciniais ir veikimo principais ar sąvokomis.

Edukacinė robotika gali būti dviejų formų. Pirmoji forma – robotika yra mokymo tikslas ar dalykas (dirbtinis intelektas, robotų mechatronika ir pan.). Antroji – robotika yra mokymo priemonė koncepcijoms suvokti ar vaizduoti (matematika, fizika ir pan.). Vienas iš unikalių edukacinės robotikos privalumų – vaikai kartu ugdo inžinerinius, problemų sprendimo, kūrybiškumo ir bendravimo įgūdžius. Interaktyvus robotų konstravimo ir programavimo procesas leidžia moksleiviams pamatyti ir paliesti, eksperimentuoti ar patobulinti savo darbo rezultatą. Žinios tampa apčiuopiamos, įgauna prasmę, taikomąją prasmę. Papertas teigė, kad baimė suklysti yra viena iš mokymosi baimių. Kurdami robotus, vaikai gali padaryti klaidų, tačiau, svarbiausia, jie turi išmokti surasti klaidas ir išmokti jas ištaisyti. Tai skatina vaikus mąstyti kritiškai, bandyti dar ir dar kartą, kol sukurs veikiantį robotą.

Edukacinėje robotikoje vaikai kuria ir programuoja robotus, kurie turi jutiklius ir gali reaguoti į aplinką, atlikti veiksmus. Tai leidžia ne tik sukurti įvairiausių robotų, bet ir panaudoti juos praktiškai, pavyzdžiui, eksperimentams atlikti. Viena iš svarbių edukacinių robotų savybių – su tomis pačiomis priemonėmis galima sukurti ir paprastų mechanizmų, ir sudėtingiausių autonominių robotų. Taigi vaikai su tomis pačiomis priemonėmis gali efektyviai mokytis ir mokykloje, ir tęsdami tolimesnius mokslus universitetuose, kolegijose ar kitose švietimo įstaigose. Todėl šiems robotams yra taikomas angliškas posakis *„having low floor, high ceiling and wide walls“* – žemas slenkstis, aukštos lubos ir plačios sienos.

Galima mokyti vaikus nuo 8–9 metų amžiaus iki universiteto. Čia yra didžiulis tyrinėjamų sričių pasirinkimas.

Vienas iš pirmųjų edukacinių robotų buvo sukurtas 1980-aisiais metais. Tai buvo robotas vėžliukas, kurį buvo galima programuoti LOGO programavimo kalba. Šiuo metu švietimo sistemoje egzistuoja ne viena edukacinių robotų sistema. Tačiau labiausiai paplitusi ir visame pasaulyje yra „LEGO Education Midstorm EV3“, kurią galima programuoti ir grafinėje aplinkoje („EV3–G“), ir tekstinėje aplinkoje („RobotC“). Verta atkreipti dėmesį, kad „RobotC“ yra viena iš nedaugelio programavimo kalbų, kuri turi tekstinę ir pusiau grafinę, pusiau tekstinę programavimo kalbą. „LEGO Education“ ne tik sukūrė robotą ir programavimo aplinką, tačiau ir gan aktyviai kūrė STEM srities mokymo turinį. Mokymas su „LEGO Education Mindstorm EV3“ robotais suteikia galimybę visiems įsitraukti į ugdymo procesą, nors darbas vyksta komandose, visi vaikai patys konstruoja ir kuria. Toks mokymo metodas ir yra vadinamas *kostrukcionistiniu.* Tai suteikia galimybę mokyti STEM dalykų, tokių kaip matematika ar fizika, kitaip, prasmingai ir su džiaugsmu.

Šiuo metu edukaciniai robotai sparčiai plinta visame pasaulyje ne tik kaip neformalaus ugdymo priemonė, tačiau tai sėkmingai integruojama ir į formalųjį ugdymą. Viena iš artimiausių perspektyvų yra STEM mokymas, kuriame robotika yra neatsiejama dalis ir gali būti naudojama mokyti įvairių disciplinų: technologijų, matematikos, informatikos, fizikos, chemijos, biologijos ir pan. Nors skiriasi ir mokymosi tikslai, ir sudėtingumo lygis, besimokančiųjų amžius yra labai įvairus – nuo ikimokyklinio amžiaus iki aukštosios mokyklos.

Vienas iš sėkmingų robotikos mokymo metodų yra projektinis mokymas. Projektinis mokymas orientuotas į ilgesnės trukmės veiklas, dažniausiai būna tarpdisciplininės, orientuotos į moksleivį ir su realaus pasaulio iššūkiais ir patirtimi. Moksleiviai sprendžia realaus pasaulio užduotis ar iššūkius, mokomas dalykas įgauna prasmės besimokančiajam, nes tuomet aišku, kam to reikia ir kur tai galima panaudoti ateityje. Projektinis mokymas ugdo ir kitus gebėjimus: valdyti procesus, dirbti komandoje, priimti sprendimus, spręsti kompleksines problemas. Vienas iš svarbesnių projektinio mokymo aspektų – siekiant projektų tikslų daugeliu atveju nepakanka vienos kompetencijos ar vieno tipo gebėjimų. Tai skatina ne tik ugdyti skirtingus gebėjimus vienu metu, bet ir dirbant komandoje su skirtingomis kompetencijomis siekti vieno tikslo. Projektai turi būti autentiški ir su prasmingu kontekstu, ir kuo didesnė sprendimų įvairovė, tuo didesnė tikimybė, kad visi mokiniai įsitrauks į mokymosi procesą. Sprendimų įvairovė priklauso ne tik nuo pateiktos užduoties, bet ir nuo priemonių, kokiomis mokiniai naudosis.

Pastaruosius dvidešimt metų vieni plačiausiai paplitusių edukacinių robotų mokyklose yra „LEGO Education Mindstorm EV3“ robotai. Patikimumas, paprastumas, daugybė sprendimų variantų, žinomas prekinis ženklas padarė didžiulį edukacinės robotikos šuolį. Tenka pripažinti, kad be „LEGO Education“ priemonių robotika vaikams būtų iš mokslinės fantastikos ar inžinerinės veiklos srities, vaikai susipažintų su robotais tik studijuodami ar profesinėje veikloje, jau po mokyklos. Tačiau pradedant nuo Paperto ir jo komandos sukurtos konstrukcionizmo teorijos, po to LOGO programavimo kalbos ir galiausiai edukacinių LEGO robotų, tai tapo prieinama kiekvienai mokyklai visame pasaulyje ir technologine, ir kainos prasme.

Galima išskirti 5 robotikos integravimo į švietimą kelius. Kiekvienas iš jų turi savo tikslus ir reikalauja mokytojo kompetencijos.

Robotikos disciplina – robotika mokoma kaip atskiras dalykas neformaliajame ugdyme. Šioje srityje labiausiai išsivystė ir išpopuliarėjo FLL varžybos. FLL – FIRST LEGO LEAGUE, FIRST (for inspiration and recognition in science and technology), LEGO – LEGO prekės ženklas ir LEAGUE – sporto lyga. Tai robotikos varžybos, skirtos 9–16 metų vaikams, kurios vyksta jau 17 metų visame pasaulyje ir dalyvauja daugiau nei 20 000 komandų. FIRST – tai JAV organizacija, kuri užsiima techninės kūrybos neformaliuoju vaikų švietimu. Ši organizacija kartu su „LEGO Education“ taip pat rengia robotikos varžybas ir pradinių klasių moksleiviams Jr. FLL (Junior FLL), ir vyresniųjų klasių moksleiviams, ir studentams FTC (FIRST technical challenge – FIRST techninės varžybos) ir FRC (FIRST robotics challenge – FIRST robotų varžybos).

STEM mokymas – robotika naudojama kaip mokymo priemonė aiškinti ar demonstruoti gamtos mokslų sąvokas, tokias kaip jėga, sukimo momentas, galia, trintis; mokyti programavimo, matematikos ir pan.

Teminis mokymas – robotų konstravimas, naudojamas mechanizmams ir sistemoms modeliuoti, pavyzdžiui, oro uosto, gamyklos, pramogų parko ir pan. Tai turėtų palengvinti ir papildyti tradicinį mokymą.

Robotika formaliajame ugdyme naudojama kaip priemonė sprendžiant konkretų uždavinį formaliojo ugdymo programoje. Pavyzdžiui, per matematikos pamokas, mokantis apskritimo ilgį. Sukonstravus robotą, reikia parašyti programą, pagal kurią robotas važiuoja tam tikru maršrutu. Tam, kad būtų galima atlikti užduotį, reikia išmokti, kaip apskaičiuoti apskritimo ilgį, tada galima apskaičiuoti, kokį atstumą ratas nuvažiuos per vieną apsisukimą. Šis pavyzdys parodo, kaip taikomas problemų sprendimo metodas ir ugdomi dalykiniai gebėjimai.

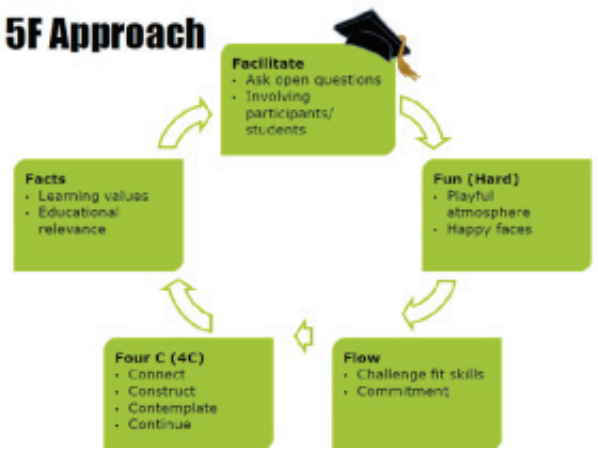
Laisva kūryba (angl. freestyle) – robotų konstravimas ir programavimas yra naudojami kaip laisvos kūrybos priemonė vaikų saviraiškai ir kūrybiškumui skatinti ir ugdyti. Tai gali būti naudojama ir kaip priemonė idėjoms ar problemoms vizualizuoti. Suaugusiųjų švietimo srityje tai gana dažnai naudojama kaip komandos formavimo mokymo priemonė.

Robotika švietime – tai ne tik džiaugsmas vaikams, bet ir iššūkis pedagogams. Tai naujos technologijos, nauji mokymosi metodai, žaisminga ir kūrybiška atmosfera, tai nėra klasikinis mokymas, net mokytojo vaidmuo yra kitoks. Kitoks ir priemonių administravimas, ir saugojimas, papildomos mokomosios medžiagos ir aplinkos paruošimas. Todėl mokytojų mokymas edukacinėje robotikoje yra viena iš svarbiausių sričių, nuo kurios tiesiogiai priklauso *konstrukcionistinio* robotikos mokymo integracija tiek į formalųjį, tiek ir į neformalųjį ugdymą.

II. „LEGO Education“ mokymosi metodika

Šioje dalyje išsamiai aprašomos LEGO Education metodikos, paremtos konstruktyvizmo ir konstrukcionizmo teorijomis. Mokymo metodika viena iš svarbiausių mokymo sudedamųjų dalių, nes padeda pažadinti mokinių smalsumą ir įtraukti juos į mokymosi procesą.

XXI amžiuje mokymosi metodika turi keistis kartu su technologiniu progresu ir vaikų poreikiais nuo *instrukcinės* į *konstrukcionistinę,* ar mišrią metodiką. Vaikai komandoje mokosi aktyviai ir kūrybiškai, greitai besikeičiančioje aplinkoje eksperimentuodami per problemų sprendimo metodiką improvizuoja ir tyrinėja, modeliuoja realaus gyvenimo pavyzdžius, suteikdami jiems asmeninę prasmę. Pedagogo vaidmuo irgi keičiasi, jis nebėra vienintelis žinių šaltinis. Pedagogas – komandos lyderis, moderatorius ir efektyvios mokymosi aplinkos kūrėjas. Toks yra „LEGO Education“ požiūris į mokymą, sukurtas kartu su S. Papertu ir jo komanda. Tai – *konstrukcionistinio* mokymo metodika ir priemonės jai diegti. 2013 metais buvo sukurtas naujas „LEGO Education Mindstorm EV3“ robotas, jau trečiosios kartos mokomasis LEGO robotas.



Ši metodinė priemonė yra parengta konstrukcionizmo ir „LEGO Education“ metodikos pagrindu. Ji sujungia laiko patikrintas teorijas, labiausiai paplitusias metodikas ir Lietuvoje susiformavusią patirtį. „LEGO Education“ metodika sudaryta iš dviejų sudedamųjų dalių: „5F“ mokymo metodikos ir „4C“ mokymosi proceso.

„5F“ mokymo metodika

„5F“ metodika – tai pagrindas, kuriant sėkmingo mokymosi aplinką ir įtraukiantį mokymo procesą su „LEGO Education“ mokymosi priemonėmis. Ta pati metodika tinka ne tik robotikos mokymui, bet ir kitoms sritims. Neseniai „LEGO Education“ sukūrė kalbų mokymosi priemones, kur irgi naudojama ta pati „5F“ metodika. Nuo pat pradžių „5F“ metodika buvo kuriama kaip atmintinė mokytojui, bet ne kaip atskira mokymo metodika. „5F“ metodiką sudaro penki elementai: faktai, fasilitavimas, smagumas, srautas ir „4C“. Visi penki elementai yra tarpusavyje susiję, todėl jie visi privalo būti mokymosi procese. Pavyzdžiui, faktai naudojami „4C“ procese, smagumas tiesiogiai daro įtaką srautui ir t. t. „5F“ metodika buvo sukurta analizuojant ir atrenkant penkis svarbiausius mokymo elementus, kurie daro didžiausią įtaką ir yra naudingiausi mokymosi procese. Nors jie ir

skirtingi, tačiau visi penki yra lygiaverčiai vienas kitam ir vienodai svarbūs mokymosi proceso kokybei.

Faktai (Facts)

Faktai naudojami kaip priemonė sudominti vaikus. Faktai – tai koncepcijos, terminai, sąvokos ar aprašymai, paimti iš mokymo medžiagos ar susiję su ja. Formaliai, faktai – tai terminai ir koncepcijos iš mokymosi programos. Faktai suteikia galimybę vaikams diskutuoti pasirinkta tema, reikšti savo nuomonę ir ją argumentuoti.

Tradicinėje mokymosi metodikoje su faktais vaikas susipažįsta iš knygų, pratybų, mokytojo. Juos moksleivis turi išmokti, įsiminti. Tačiau mokymosi esmė yra ne tik sausas faktų žinojimas, tačiau ir mokėjimas juos pritaikyti realiame gyvenime. Pagal Jungtinėse Amerikos Valstijose atliktą apklausą, 4 iš 10 amerikiečių nemėgsta matematikos todėl, kad, išskyrus pinigų skaičiavimą, nemato tame prasmės realiame gyvenime. Todėl faktai ugdyme labai svarbūs susiejant ir įprasminant mokymo teorijas ir koncepcijas. Kai žinios įgyjamos tik mokantis, įgyjamos žinios ir gebėjimai nesusieti su faktais, ir moksleivis šia tema gali tik teoriškai postuluoti. Kita vertus, jeigu mokymasis susietas su faktais, mokiniai patys eksperimentuoja konstruodami robotus, tada jie ne tik suvokia mokomą temą, bet ir turi savo nuomonę, kurią gali argumentuoti diskusijų metu. Kaip vieną iš pavyzdžių galima pateikti robotikos pamoką, kurioje mokoma apie judesio jutiklius. Kaip faktai yra pateikiama keletas pavyzdžių, kur naudojami šie jutikliai ir kokia iš to nauda (automatinės prekybos centrų ar modernių įstaigų durys padeda efektyviai taupyti šilumos energiją; naktinio apšvietimo lempos namų laiptinėse sumažina elektros energijos suvartojimą ir kt.).

Fasilitavimas – dinaminis koordinavimas (Facilitation)

Fasilitavimas apibūdina mokytojo vaidmenį mokymo procese, jo požiūrį į mokymą. Fasilitavimas turi daug paralelių su koučingu (ugdomasis vadovavimas), nes abu jie nukreipti į pagalbą asmenybei įgyti savo įžvalgų. Už viso to slypi teorija, kuri teigia, kad savarankiškai įgytos įžvalgos ir suformuotos išvados turi gilesnę įtaką asmenybės gebėjimui argumentuoti ir taikyti realiame gyvenime.

Vienas iš fasilitatoriaus vaidmenų – palaikyti mokymosi procesą: padėti planuoti ir siekti užsibrėžtų tikslų, palaikyti grupinį darbą, netrukdant mokinių tarpusavio diskusijos, palaikyti diskusiją kolektyve. Fasilitatoriaus uždavinys – sudaryti ir pateikti bazinį ar pradinį instrukcijų paketą, laiku ir vietoje užduoti tinkamus klausimus, kurie įkvėptų reflektuoti ir mokytis, ir kad tai būtų mokinių atradimai ir jų sprendimo variantai.

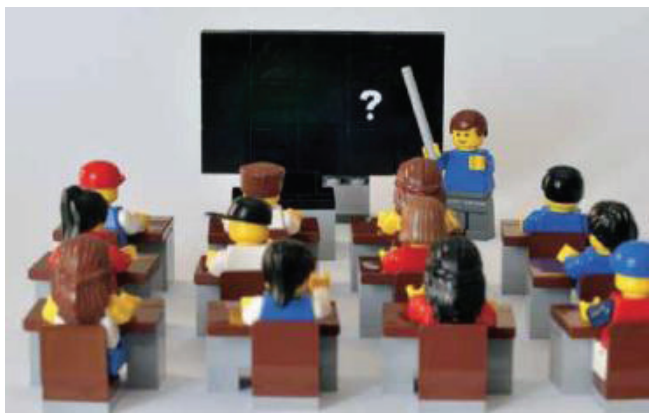
Mokymasis tampa savotišku nuotykiu, kartu su gidu įgyta patirtis pakeliui į sprendimus ir asmeninius atradimus yra emociškai jaudinantis asmeninis pasiekimas, kuris visiškai skiriasi nuo teoriškai įgytų žinių. Kuo didesni pasiekimai, tuo didesnis emocinis

pasitenkinimas, tuo didesnis įsitraukimas į mokymosi procesą. Mokytojo kaip fasilitatoriaus, veiklos koordinatoriaus vaidmuo yra kitoks nei klasikinio mokytojo ar instruktoriaus, nes fasilitatorius suteikia galimybę per savo patirtį ir atradimus įgyti žinių. Taip mokinys tampa aktyviu sprendimų ieškotoju, įgyjamos ne tik žinios ir patirtis pagal mokymo programą, bet ugdomi ir asmeniniai problemų sprendimo, kritinio mąstymo ir komunikaciniai gebėjimai. Mokytojas- fasilitatorius pamokoje mokinius skatina būti labai aktyvius, savarankiškai formuoti klausimus ir spręsti kylančias problemas.

Sąlygos atsirasti sėkmingam fasilitavimo procesui:

Aiškiai suformuluoti ir apibrėžti pamokos rėmai ir tikslai. Tam, kad būtų pasiekta srauto būsena mokymosi metu, turi būti aiškiai nustatytos ribos (užduočiai atlikti skirtas laikas ar pamokų kiekis, aiškiai suformuluoti tikslai ir pan.). Tačiau turi būti galimybių keisti, tobulinti ir klysti. Mokytojas-fasilitatorius turi taip organizuoti mokymosi procesą, kad mokymosi eiga liktų numatytuose rėmuose ir būtų pasiekta pamokos tikslų.

Valdomas mokymosi procesas. Jeigu A dalyje fasilitatoriaus pareiga suformuoti tikslus, kurie vaikus įtrauktų į mokymosi srautą, tai proceso valdymo metu svarbu išlaikyti srauto būseną tol, kol bus pasiekta nustatytų tikslų. Svarbu, kad fasilitatorius stebėtų procesą ir pagal poreikį koordinuotų, pateikdamas papildomų instrukcijų ir nuorodų, užduodamas klausimus, komentuodamas taip, kad vaikai dirbtų ir savarankiškai, ir mokymosi procesas būtų nenuobodus ir įtraukiantis.



SEQ Figure * ARABIC 9 pav. LEGO klasė

Nuolatinis grįžtamasis ryšys. Grįžtamas ryšys yra labai svarbus, kad mokinys nesijaustų paliktas vienas. Mokiniai visada turėtų gauti patarimą ar užuominą. Fasilitatoriaus refleksija yra šiek tiek kitokia nei klasikiniame mokyme. Fasilitatoriaus užduotis yra įdėmiai išklausyti problemą ar klausimą ir atsakyti, kaip jis suprato iškeltą problemą ar klausimą. Dažniausiai pakanka išgirsti klausimą ar problemą tarsi veidrodžio atspindy. Įmanomas ir tiesioginis atsakymas į klausimą, tačiau tokiu atveju geriau užduoti klausimus, vedančius link atsakymo, kad mokiniai jaustųsi esantys situacijos šeimininkai ir jie būtų problemos sprendėjai.

Atviri, fasilituojantys klausimai. Dialogas ir reflektavimas turi vykti viso mokymosi metu. Tai padeda kontroliuoti mokymosi kokybę ir laiku pastebėti nukrypimus. Uždavinėjant klausimus svarbu, kad tai būtų atviri klausimai, nes tuomet sprendimų ir atsakymų yra ne vienas. O atsakymai reikalauja argumentų, paaiškinimų. Svarbu, kad fasilitatorius suvoktų, jog mokymosi procesas yra skirtas vaikams, o išmoksti geriausiai, kai patiri, realizuoji ir pats ar su komanda įgyji žinių.

Konstruktyvi veikla ir komunikacija. Mokymosi procese mokytojo-fasilitatoriaus santykis su vaiku turi būti paremtas tarpusavio pagarba. Mokymasis fasilitavimo metodu – tai mokymo procesas, kelionė į pažinimą, nuotykis, kur ir mokytojas, ir mokiniai sėdi tame pačiame vežime. Mokiniai ir fasilitatorius yra lygūs, ir abi pusės turi gerbti kiekvieno darbą, idėjas ir nuomones. Tai nereiškia, kad problemos turi būti ignoruojamos, klaidos nutylėtos, fasilitatorius garbinamas, nes jis mokytojas. Tai reiškia, kad mokymasis turi būti konstruktyvus, problemos ir klaidos turi virsti pažinimo galimybėmis ar kūrybiškais sprendimais.

Smagumas / džiaugsmas (Fun)

Šiuolaikinės psichologijos požiūriu žmogaus laimė dažniausiai apibūdinama jo pamatiniu pasitenkinimu gyvenimu ir patiriamo džiaugsmo kiekiu. Pamatinis pasitenkinimas priklauso nuo to, kaip žmogus suvokia savo gyvenimo prasmę bei pats save realizuoja. Tai ir yra vienas pagrindinių džiaugsmo šaltinių. Žaidimas ir džiaugsmas dažniausiai apibūdinami kaip įtraukianti smagi ir linksma veikla.

Tačiau pramoga nėra vienintelis žaidimo tikslas. Žmogus žaidžia, nes nori džiaugtis gyvenimu, patirti emocinę gerovę, asmeniškai ir socialiai tobulėti. Žaidžiant irgi galima mokytis bei įgyti gebėjimų (žaisdami futbolą išmokstame taisykles ir įgyjame naujų gebėjimų, žaisdami šachmatais laviname loginį mąstymą ir kt.). Mokymasis žaidžiant buvo išsamiai ištirtas ir aprašytas S. Paperto. Pagal konstrukcionizmo teoriją žaidimas padaro mokymąsi kitokį, smagų, kūrybišką ir savarankišką.

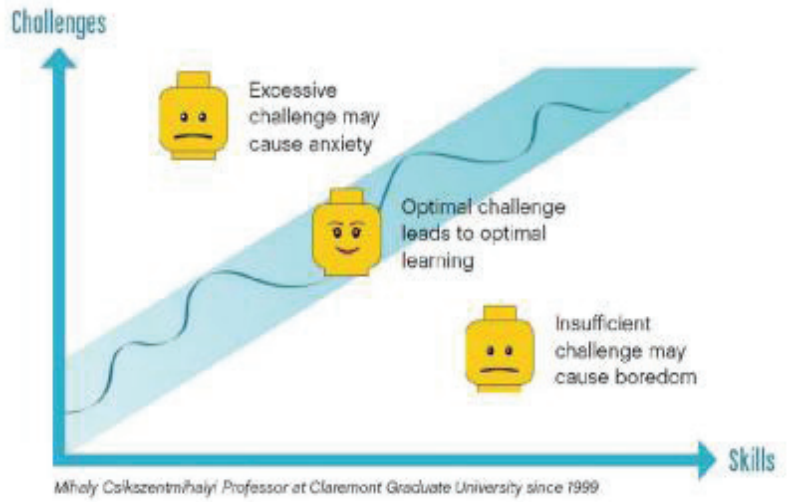
Šiuo metu mokyklose vyrauja nuostata, kad žaidimas ir mokymo procesas – tai du skirtingi dalykai. Tačiau taip tikrai nėra, nes vaikas ikimokykliniame amžiuje pakankamai sėkmingai mokosi ir be mokyklos. Kieme vaikai žaidžia futbolą, nors niekas jiems to neliepė daryti. Žaidimas – tai motyvatorius mokymo procese, žaidimas – tai smagus mokymosi procesas, išlieka smagi emocija ir keičiasi vaikų požiūris į mokymąsi.

Smagus mokymasis sukuriamas ne tik per žaidimą. Tam reikalinga atitinkama mokymosi aplinka, priemonės, turinys, metodika ir atitinkamas mokytojo požiūris. Konstrukcionizmo teorijoje keičiasi mokymosi paradigma, kurioje besimokantysis yra aktyvus ir savarankiškas, o mokytojas – pagalbininkas ir bendražygis.

Srautas (Flow)

„Srauto“ sąvoką (angl. „Flow“) pirmasis pradėjo naudoti vengrų kilmės JAV psichologas profesorius Mihalas Čiksentmihalis (1991), tyrinėjęs laimės fenomeną. Srauto teorija taikoma labai plačiai. Nuo profesionalios veiklos iki žaidimo, kūrybiškumo išraiškos ir mokymosi. Srauto teorija ir susiję tyrimai susilaukė daug dėmesio švietime, mokymosi

metodikose, nes tikima, kad srautas sustiprina mokymosi procesą, kad laimė ir prasmė turi būti neatsiejama mokymosi proceso dalis.



Srautas – tai srovė, kuri neša laimės pojūčio link į gyvenimo pilnatvę. Tai būsena, kai prarandame laiko nuovoką, pamirštame visus rūpesčius, nerimą, baimes, kai jaučiamės tarsi nešami kosminės energijos srovės. Šis išgyvenimas toks nuostabus, kad nesinori net grįžti į realybę... Kuo dažniau jaučiame gyvenimo pilnatvę, tuo mažiau gyvenimiškosios energijos iššvaistome veltui ir daugiau patiriame džiaugsmo. Pasak psichologo M. Čiksentmihalio, to galima išmokti. Laimė kuriama tik vienu būdu: mintimis. Žmonės patys renkasi, kaip vertinti tai, kas jiems nutinka. Pinigai, kitos materialinės gėrybės ir netgi gera sveikata, anot jo, laimės negarantuoja. *„Taip, jie gali pagerinti gyvenimą, tačiau jeigu nemokate suvaldyti savo psichinės energijos, jums vis tiek kažko trūks.“*

M. Čiksentmihalis tyrinėjo, kodėl vieni sugeba jaustis laimingi, o kiti ne. Tuomet buvo ištirtos dvi žmonių grupės. Pirmoji – tie, kurie gyvenime prarado viską, įskaitant sveikatą ir artimuosius, tačiau nepalūžo. Jie atrado naują gyvenimo prasmę, naujų stimulų tobulėti ir tapo laimingesni už tuos, kuriuos likimas, regis, lepino. Antroji grupė – kūrybos žmonės: dailininkai, muzikantai, rašytojai. Kai jie panirdavo į savo veiklą, visas išorinis pasaulis dingdavo. Juos apimdavo ypatinga būsena, panaši į apsvaigimą. Po daugybės metų tyrimų jis priėjo išvadą, kad ir vienos, ir kitos grupės žmonių pilnatvės raktas yra gebėjimas valdyti savo psichinę energiją.

Tačiau srautas nėra tik akademinis dalykas. Paskelbus srauto teoriją, jau po kelerių metų praktiškai ją imta taikyti įvairiose srityse. Kai siekiama pagerinti gyvenimo kokybę, srauto teorija puikiai gelbsti. Ji įkvėpė kurti eksperimentinę mokymo programą, laisvalaikio gaminius ir paslaugas, mokyti įmonių vadovus. Srautas praverčia kuriant idėjas ir veiklas klinikinėje psichoterapijoje, jis naudingas nepilnamečių teisės pažeidėjų reabilitacijos programose, organizuojant veiklą senelių namuose, išdėstant muziejaus eksponatus ir taikant ergoterapiją neįgaliesiems. Visa tai įvyko per keliolika metų po to, kai moksliniuose žurnaluose pasirodė pirmieji straipsniai apie srautą, o tendencijos rodo, kad ateityje teorijos įtaka bus dar stipresnė.

Kaip pavyzdį galima pateikti žaidimą kompiuteriu. Iš šalies gali atrodyti, kad tai nekaltas užsiėmimas. Tačiau jis leidžia nuolat ir labai lengvai patirti „srautą“ (geriausi kompiuteriniai žaidimai naudoja srauto teorijos principus) – tobulėjant žaidėjo įgūdžiams, didėja iššūkių lygis, todėl žaisti nenusibosta tol, kol žaidimas nepasidaro pernelyg paprastas arba pernelyg sudėtingas.

Kuo dažniau jaučiate gyvenimo pilnatvę, tuo mažiau gyvenimiškosios energijos iššvaistote veltui ir tuo daugiau patiriate džiaugsmo. Pasak psichologo, to nėra labai sunku išmokti. Reikia laikytis tik penkių taisyklių, kad pasiektumėte srauto būseną:

- Užsiimkite tuo, kas jums gerai sekasi, kad nešvaistytumėte energijos abejonėms savimi.
- Susikoncentruokite į procesą, kad sąmonėje nebūtų vietos nereikšmingai tuo momentu informacijai.
- Turėkite tiksliai suformuluotus tikslus ir galimybę greitai gauti atgalinį ryšį.
- Svarbiausia – kad jūsų darbas ne tik atitiktų jūsų įgūdžius, bet ir palaipsniui reikalautų tobulėti.
- Į darbą žiūrėkite kaip į žaidimą. (Biochemikas Lainusas Polingas (Linus Pauling), dukart Nobelio premijos laureatas, dirbo visą gyvenimą be perstojo ir mirė būdamas 93 metų. Kartą jis prasitarė niekada nejautęs nuovargio, nes į darbą žiūrėjęs kaip į žaidimą).

Robotikos mokymo procese srautas pasiekiamas dviem aspektais.

Pirma, vaikai mokosi konstruodami „LEGO Education Mindstorm EV3“ robotus. Konstravimo procesas ir LEGO kaladėlių fenomenas įveda ne tik vaikus, bet ir suaugusiuosius į srauto būseną. Turbūt daugeliui iš jūsų teko stebėti ar pačiam patirti konstravimo magiją, kai prarandama laiko nuovoka ir pasineriama į kūrybos procesą.

Antra, LEGO robotai suteikia galimybę kiekvienam sukurti savo asmeninį sprendimą, kuris atitinka jo kompetenciją ir galimybes. Konstruojant robotą sprendimų yra daug ir visi jie gali būti teisingi. Pavyzdžiui, šešios aštuonių dalių (2x4) LEGO kaladėlės gali būti sujungtos 915, 103, 765 variantais.

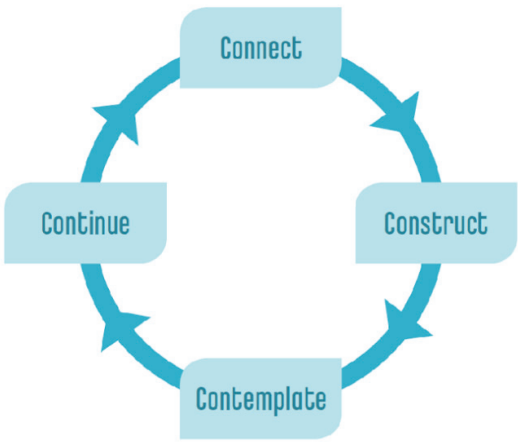
Norint taikyti srauto metodiką mokymosi procese, reikalingos ne tik atitinkamos priemonės, bet ir užduotys turi būti tinkamai parengtos. Atviri klausimai, lanksčios užduotys, keli atsakymo variantai, daug sprendimo variantų. Tai suteiks vaikams galimybę mokytis pagal savo gebėjimus ir tempą, eksperimentuoti ieškant geriausio įmanomo sprendimo ir pan. Pradinis užduoties lygis suformuojamas pagal grupės gebėjimų, įgūdžių ir žinių lygį. Taikant srauto metodiką, mokytojas turi suvokti, kad kiekvieno mokinio rezultatai bus skirtingi, nes kiekvienas dirbs pagal save ar pagal grupės lygį. Daugeliu atvejų grupinis darbas suniveliuoja kompetencijų skirtumus, nes stipresni vaikai padeda silpnesniems, tai suteikia jiems ne tik emocinį pasitenkinimą, bet ir įtvirtina jau įgytas žinias per praktinį pritaikymą.

Pilnatvės srautas yra stiprus išgyvenimas, sukeliantis savotišką priklausomybę, todėl reikia būti budriems, per gilus buvimas sraute gali atitraukti nuo realaus gyvenimo. Svarbu išmokti ne tik pagauti srovę, bet ir išsilaisvinti iš jos, kai tik realybė to pareikalauja.

„4C“ mokymosi procesas

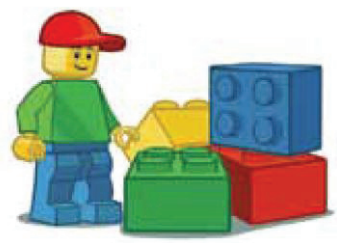
„4C“ mokymosi procesas – tai kertinė dalis „LEGO Education“ mokymosi metodikoje. „4C“ – tai mokymosi proceso struktūra, kuri užtikrina geriausią mokymosi procesą ir resursų panaudojimą. Tai pažinimo ir susipažinimo procesas, kuris orientuotas į įgimtus mokinių gebėjimus konstruoti ir kurti, tirti priežastis ir poveikį, eksperimentuoti

sprendžiant problemas. „4C“ procesas – tai konstrukcionistinio mokymo susistemintas procesas, sudarytas iš keturių mokymosi etapų: susiek (angl. *connect*), konstruok (ang. *construct*), mąstyk (ang. c*ontemplate*) ir tęsk (angl. *continue*). Tai lyg iteracinė mokymosi spiralė, kur galimi ir daliniai, ir viso proceso pakartojimai.



Susiek (Connect)

Konstrukcionistinio mokymo mokomojo turinio susiejimas su realiais gyvenimo pavyzdžiais yra vienas iš svarbiausių elementų, nes tuomet mokymasis tampa prasmingas ir mokiniai įsitraukia į mokymo procesą nuo pat pirmųjų akimirkų. Susiejimo fazėje mokytojas pateikia atvirus klausimus ir idėjas. Tai sužadina vaikų smalsumą ir skatina tyrinėti ir eksperimentuoti ieškant sprendimų. Sužadintas smalsumas įtraukia moksleivius į savarankiško mokymosi procesą ir sukuria vidinę motyvaciją mokytis. To siekia viso pasaulio švietimo sistemos, nes toks mokymasis yra pats efektyviausias.



Savimotyvacija prasideda sužadinus smalsumą, susiejant mokomojo turinio naudą ir pritaikymo sritis. Susiejant mokymosi temas svarbu, kad vaikas būtų įsitikinęs, kad jis galės vienaip ar kitaip atlikti šią užduotį ir kad sėkmė priklauso tik nuo jo asmeninių pastangų. Sujungti jau esamas žinias ir jau įgytus gebėjimus yra taip pat svarbu, nes tinkama pradžia suteiks daugiau pasitikėjimo savimi ir palengvins, pagreitins, įtrauks į srauto būseną. Ruošiant susiejimo fazę svarbu atsižvelgti į vaikų socialinį ir demografinį kontekstą, nes sėkmingai susiejama tik tuomet, kai besimokančiajam tai yra aktualu. Pavyzdžiui, užduotis apie automobilių spūstis miestuose gali būti visiškai neaktuali kaimo vietovės vaikams, kur tokių reiškinių nebūna. Kad visa klasė įsitrauktų į mokymosi procesą, svarbu atsižvelgti ir į socialinius skirtumus .

Susieti – tai parodyti prasmę, naudą ir kad tai bus panaudojama gyvenime. Tuomet tai tampa svarbu ir įsimintina visam gyvenimui.

Konstruok (Construct)

„Vaikams konstruojant realaus pasaulio artefaktus, tuo pačiu konstruojamos žinios ir suvokimas“ (ištrauka iš „The System for Learning“ Manifesto by LEGO Education, 2011). Konstravimo fazėje moksleiviams suteikiama galimybė konstruoti atsakymus į pateiktus klausimus. Per konstravimą ir eksperimentus moksleiviai lengviau suvokia priežasties ir pasekmės ryšį, supranta juos supančio pasaulio mechanizmų veikimo principus ir konstruoja žinojimą.

Konstravimas – tai konstruktyvistinio mokymo pagrindas. Kaip teigė Žanas Pjažė, *„Neįmanoma perkelti žinių kitiems – kiekvienas turi sukonstruoti žinojimą sau ir pagal save, tam, kad jie tai įsisavintų.“* O konstrukcionistinio mokymo įkūrėjas Seymouras Papertas teigia, kad *„Mokymas ir mokymasis tai tik konteksto kūrimas, kuris sudaro sąlygas mokytis*“. Tai reiškia, kad besimokantysis yra aktyvus mokymosi kūrėjas, o ne pasyvus klausytojas.

Konstravimo metu moksleiviams pateikiami įrankiai ir sistema objektų kūrimui, rekonstravimui ar konstravimui. Sukonstruoti sprendimai suteikiant galimybę vaikams modeliuoti ir eksperimentuoti, kad suvoktų priežasties ir pasekmės ryšius ir geriau suvoktų realiame pasaulyje egzistuojančius reiškinius ar mechanizmų veikimo principus. Konstravimas padeda formuoti įgūdį, įgytas žinias taikyti praktinėje veikloje ir taip užtikrinti ilgalaikį rezultatą.

SEQ Figure * ARABIC 14

Konstravimas sužadina ne tik emocinę atmintį, būnant sraute, bet ir aktyvuoja motorinę atmintį. Motorinė atmintis – tai sugebėjimas įsiminti, laikyti atmintyje judesius ir jų sekas. Kuo daugiau atminties tipų aktyvuojama mokymosi metu, tuo didesnė tikimybė, kad įgytos žinios išliks ilgalaikėje atmintyje ir bus pritaikytos praktiškai.

„4C“ proceso metu aktyvuojamos visų tipų atmintys. Žodinė atmintis aktyvuojama susiejimo fazės metu užduodant klausimus ir dalinimosi fazės metu reflektuojant ir dalinantis patirtimi su draugais. Srauto būsena aktyvuoja emocinę atmintį, kuomet patirtis teikia savirealizacijos džiaugsmą. Konstravimo metu aktyvuojama motorinė atmintis, tai lyg banko kortelės kodo fenomenas, daugelio žmonių paklausus banko kortelės kodą, ne kiekvienas atsakytų iš karto, tačiau priėjus prie bankomato kortelės kodas atkuriamas akimirksniu. Na ir ketvirtas vaizdinės atminties tipas sukuriamas vizualizuojant sprendimus ir konstruojant objektus.

Konstravimas neturi būti pamokos tikslas, tai tiesiog mokymo priemonė, tai galimybė kiekvienam ir savaip eiti pažinimo keliu. Konstravimą procesą galima suskirstyti į tris tipus:

Tyrinėjamasis – paprasti mechanizmai modernizuojami ir tobulinami. Šiuo būdu moksleiviai susipažįsta su pagrindinėmis koncepcijomis ir mechanizmų veikimo principais.

Konstravimas pagal instrukcijas – pateikiamos konstravimo instrukcijos, roboto konstravimas naudojamas tik srauto būsenai pasiekti ir personalizuoti įžvalgas, o sukonstruotas objektas naudojamas mokymo proceso metu.

Problemų sprendimo – moksleiviai kuria objektą nuo pradžių, nuo dizaino iki programos. Šiuo atveju mokytojo indėlis yra svarbiausias, nes tik nuo jo priklauso kaip ir ar pasieks iškeltus tikslus.

Šie trys metodai gali būti naudojami ir tai pačiai pamokai vesti, kiekvienu atveju sunkinant keliamas užduotis ir akcentuojant skirtingus mokymosi uždavinius. Taip tai suteikia mokytojui laisvę ta pačia pamoka mokyti skirtingo lygio klasę, parenkant jiems tinkamiausią variantą.

Dalinkis (Contemplate)

Dalinimasis yra daugiau negu pasidalinimas informacija, bet ir konstravimo patirties apmąstymas, ir aptarimas. Mokytojas inicijuoja ir moderuoja konstravimo įspūdžių aptarimą ir refleksiją. Moksleiviai pasidalija savo patirtimi, kaip jiems sekėsi spręsti užduotis, dirbti ir komunikuoti komandoje. Labai svarbus yra mokytojo vaidmuo moderuojant reflektavimo ir pasidalijimo patirtimi procesą. Svarbu užduoti tinkamus klausimus, kurie skatintų moksleivius draugiškai dalintis savo įžvalgomis.

Vien tik konstravimo procesas, neaptariant patirties, būtų neefektyvus. Svarbu, kad konstravimo fazės metu įgytos žinios būtų verbalizuojamos dėl trijų priežasčių:

Tai yra vienas iš būdų, kaip patikrinti mokymosi proceso eigą ir mokinių pasiekimus, sužinoti, kaip moksleiviai mąsto ir suvokia mokomąjį turinį. Šiame etape mokytojas gali koreguoti užduotį ar mokymosi eigą kitoje fazėje – „ Tęsk“. Pavyzdžiui, jeigu moksleiviams užduotis buvo per lengva, tuomet tobulinimo fazėje pateikiama sudėtingesnė užduotis. Kita vertus, jeigu užduotis per sunki, tuomet mokytojas turi išsiaiškinti, kokia užduoties dalis buvo sunkiau suvokiama, ar kurį darbą buvo sunkiau atlikti. Priklausomai nuo atsakymo mokytojas gali iškelti diskusiją grupėje, tuomet būtų prieita prie kolektyvinio sprendimo: ar pakoreguoti tęsimo fazės užduotį, fokusuojantis į aktualią temą.

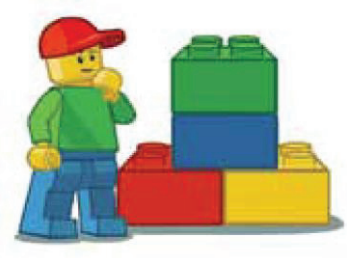
Tai yra galimybė vaikams užpildyti spragas. Grupės diskusijoje, kai visi dalijasi savo patirtimi, įgyjama grupinė patirtis. Sprendimo būdai galėjo buvo skirtingi, nes kiekvieno pradinės žinios ir patirtis skiriasi, o ką jau kalbėti apie unikalią mąstymo struktūrą. Pavyzdžiui, vienai grupei geriau sekėsi konstruoti mechaninius sprendimus, kitai sukurti algoritmą ir parašyti roboto programą. Refleksijos metu pirma komanda pasisems idėjų programai sukurti, o antroji patobulintų savo roboto konstrukciją. Dalijimosi ir apmastymo fazė būtent ir skirta tam, kad moksleiviai mokytųsi ir iš klasės patirties.

Refleksijos ir dalijimosi metu, mokytojui moderuojant, moksleiviai jungia savo įgytą patirtį ir žinias, susieja jas su realaus pasaulio pavyzdžiais, suvokia taikomąją prasmę.

Dalijimosi fazė – tai ir apmąstymo fazė, angliškai naudojamas terminas kontempliuoti (Contemplate), tačiau pagal savo esmę tai labiau pasidalijimo patirtimi procesas. Be to, tai palaiko savarankiško mokymosi procesą, nes ne mokytojas padeda atsakyti į klausimus, o bendramoksliai su savo unikaliais sprendimais papildo esamą patirtį ir praplečia žinojimo akiratį. Taip pat šioje fazėje netgi rekomenduojama moksleiviams fiksuoti tarpinius rezultatus ir aprašyti savo įžvalgas. Tai rekomenduojama dėl kelių priežasčių: pirma, kad būtų galima palyginti tarpinius ir galutinius rezultatus ir antra, esant sudėtingesnėms pamokoms dažnai prireikia kelių ar daugiau pamokų, tuomet aptarimo fazėje fiksuojami tarpiniai pasiekimai, o kitos pamokos metu tęsiama pradėta pamoka ar jų ciklas.

Mokiniai apsvarsto, ką jie išmoko, kalba apie tai ir bendrina įžvalgas. Mokiniai yra skatinami užduoti klausimus apie procesą ir palengvinti mokymąsi iki šiol. Šioje fazėje mokytojo dalyvavimas yra aktyviausias iš visų keturių fazių.

Tęsk (Continue)



Tęsimo fazė – tai įgytos patirties ir žinių (konstravimo ir dalijimosi fazėse) pritaikymas ir panaudojimas sukurtam objektui tobulinti ar užbaigimui. Dažniausiai šioje fazėje konstravimas – tai laisva kūryba, paremta pamokos rėmuose įgyta patirtimi ir žiniomis. Tęsimo fazėje moksleiviai įgyja kompetenciją taikyti įgytas žinias kartu su įgyta patirtimi. Vienas iš šios fazės privalumų yra kūrybiška esmė, nes moksleiviai kuria ar tobulina remdamiesi savo patirtimi ir tai, kas jiems atrodo įdomu ir prasminga, o tai veda link srauto būsenos, kai moksleiviai išlaiko savo susidomėjimą ir aktyvumą iki pamokos pabaigos. Tęsimo fazė – tai užbaigimo fazė, ir kiekvienas turi savo pabaigą, kuri suteikia vidinį pasitenkinimą savimi, savo jėgomis ir žiniomis, ir įprasmina visą mokymosi procesą. Laiko valdymo prasme tęsimo fazė yra viena pažeidžiamiausių, nes užduotis kūrybiška ir ši fazė yra paskutinė, pamokos pabaigoje.

„4C“ metodikos taikymas gali būti ir kartotinis, tarkime, dalijimosi ir tęsimo fazės gali būti kartojamos net keletą kartų, priklausomai nuo užduoties pobūdžio, pamokos tikslų ir tam skirto laiko. Ištęstinio mokymosi metu, kai vienai temai išdėstyti reikalinga kelios ar daugiau pamokų arba projektinio mokymo metu, kai projektas gali tęstis ir mėnesį, atskirą pamoką privaloma baigti bent jau dalijimosi faze, tačiau rekomenduojama pereiti visas fazes. Kituose šios metodinės medžiagos skyriuose bus pateikti pamokų pavyzdžiai, kurie sukurti pagal „4C“ principą.

III. Edukacinių robotų aprašymas

Šiame skyriuje pateikiami edukacinių robotų, dažniausiai naudojamų mokyklose, aprašymai ir techninės specifikacijos. Robotų aprašymas padės mokytojams pasirinkti mokomuosius robotus pagal mokinių amžiaus grupes ir mokymosi tikslus.

„Lego education WeDo 2“ robotas

Roboto aprašymas

Kompanija „Lego Group“, nuolat stebinanti novatoriškais kūriniais, lavinančiais vaikų vaizduotę, sukūrė patobulintą „WeDo“ mokomojo roboto versiją – „WeDo 2.0“.

Modernus „Lego“ kūrėjų išradimas – „WeDo 2.0“ – reprezentuoja „žaisk gerai“ (vertimas iš frazės danų kalba „leg godt“, iš kurios kilo „lego“ pavadinimas) koncepciją. Neribotų galimybių žaislas į vientisą kūrinį yra formuojamas pasitelkus patvarias bei universalias detales, kurios ne tik suteikia galimybę plėsti žinias ir lavinti kūrybinius įgūdžius, bet ir, dėl unikalių savybių, padeda mokytis programavimo, inžinerijos ir, žinoma, robotų technikos pagrindų.

Kūrėjų teigimu, „WeDo 2.0“ robotas žaislas yra linksmybių ir mokslo sąjunga, paremta novatoriško mokymosi patirtimi. Konstruktorius remiasi „Naujosios kartos mokslo standartais“, todėl realizuoja geriausią įmanomą aplinką vaiko ugdymui. Atsižvelgiant į autentiškas „WeDo 2.0“ ypatybes nekyla abejonių, jog kūrėjai neklysta. Modernus, iš „Lego“ detalių surenkamas, žaislas pasižymi novatoriškais interaktyvumo bruožais. Tačiau svarbiausia, jog inovatyvus „WeDo 2.0 turi integruotą bevielio ryšio įrankį (angl. Documentation Tool), gebantį išsaugoti informaciją ir naudojamą mokinių progresui fiksuoti. Taip pat, efektyviai bei smagiai vaikų edukacijai užtikrinti, yra integruota dizaino biblioteka (angl. Design Library) bei programuojamas „SmartHub“ rinkinys.

Be to, unikalu ir tai, jog sistema yra sukurta taip, kad dėl unikalaus mokymo plano vaikai turi priimti individualius dizaino ir problemų analizavimo sprendimus, todėl pamažu ne tik lavina kodavimo ar inžinerijos žinias, bet ir ugdo pasitikėjimą savimi.

Su šiuo robotikos ir kitų tiksliųjų mokslų mokomuoju rinkiniu namuose, klasėje ar bet kur kitur galite mokslą padaryti gyvu, susietų su realybe ir interaktyviu. Naujausia „LEGO Education“ metodika paremtas rinkinys lavina 21 amžiuje reikalingiausias kompetencijas, tokias kaip:

- Problemų projektavimas trimatėje erdvėje (Designing)
- Galimų sprendimų būdų tyrinėjimas (Investigating)
- Sprendimų modeliavimas (Modeling)
- Algoritminio mąstymo lavinimas, rašant programas (Computing)
- Kūrybiškumas
- Prezentaciniai įgūdžiai
- Pasitikėjimas savimi ir savo gebėjimu kurti

Išskirtiniu šį LEGO mokomąjį rinkinį padaro naujausias technologijas naudojantys elektroniniai komponentai: išmanusis procesorius (SmartHUB), motoras ir net du jutikliai: atstumo jutiklis (Motion sensor) ir pozicijos (Gyro sensor). Sujungus šiuos išmaniuosius komponentus ir jau visam pasauliui žinomas LEGO dalis, Jūsų jaunasis išradėjas ir būsimasis mokslininkas gali pasinerti į mokslo ir atradimo pasaulį. Kiekvieną dieną sukonstruodamas po naują mokslinį eksperimentą, po naują robotą, po naują atradimą. Taip pat, kad mokymosi procesas būtų kuo labiau įtraukiantis ir lavinantis, prie „WeDo 2.0“ einanti programinė įranga supažindina išradėją su grafinio programavimo pagrindais ir leidžia rašyti pirmąsias programas, kurios atgaivina sukurtus LEGO išmaniuosius robotus ir projektus. „WeDo 2.0“ programavimo aplinkoje taip pat rasite įžanginių pamokėlių pradedantiems, kurios padės sėkmingai pradėti konstruoti ir programuoti savo pirmuosius projektus. „WeDo 2.0“ – tai rinkinys, skirtas smalsiems vaikams, išmaniems ir inovatyviems tėveliams ir mamytėms bei pažangioms ir į rezultatus orientuotoms mokykloms. Pradėkite auginti draugišką išradėją jau šiandien ir tapkite sumania šeima!

Roboto komplektacija

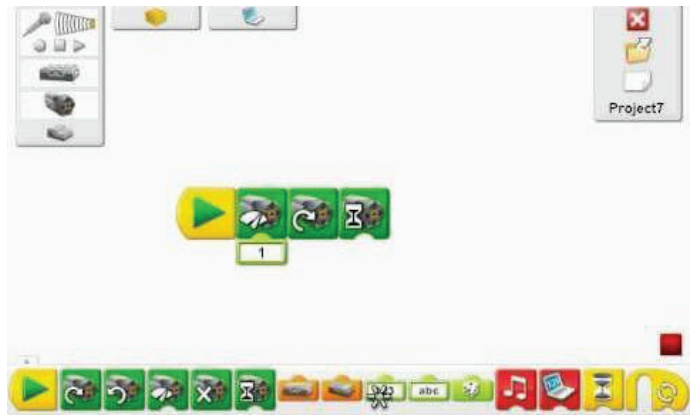
„LEGO Education WeDo2“ (gamintojo kodas 45300) Rinkinys sudarytas iš:

- Aukščiausio plastiko kokybės „LEGO Education“ daugkartinio naudojimo ir **transportavimo dėžė** su daugybe mažesnių sekcijų, skirtų skirtingiems „LEGO System“ komponentams.
  - 280 „LEGO System“ dalių viename rinkinyje.
- **Išmanusis procesorius** (SmartHub) – tai svarbiausia robotų rinkinio dalis, naudojanti naujausią BTL (Bluetooth Low energy) technologiją, kuri be jokių slaptažodžių leidžia Jūsų kompiuteriui arba planšetei prisijungti prie išmaniojo procesoriaus ir jį programuoti. Procesorius yra maitinamas 2AA baterijomis arba galima įsigyti pakraunamas specialias šiam rinkiniui skirtas baterijas su įkrovikliu. Taip pat šis procesorius turi 2 jungtis, skirtas motorams ir jutikliams, LED lemputę, kuri gali šviesti net 10 skirtingų spalvų, kurios, kaip ir motorai bei jutikliai, gali būti valdomos programavimo programa arba aplikacija.
- **Variklis** – vidutinio dydžio LEGO motoras, skirtas suteikti judėjimo galimybių Jūsų išradėjo kūriniams. Gali suktis pagal ir prieš laikrodžio rodykles. Variklio darbas yra valdomas programavimo būdu.
- **Atstumo jutiklis** – tai jutiklis, skirtas matuoti artėjančius ir tolėjančius objektus nuo jutiklio tam, kad sukurti projektai / robotai galėtų geriau suvokti juos supančią aplinką. Matomas atstumtas yra nuo 1 iki 15 cm.
- **Pozicijos jutiklis** – su šiuo jutikliu sukurti robotai / projektai gali reaguoti net į 6 skirtingas pozicijas (aukštyn, žemyn, į kairį šoną, į dešinį šoną, ramybės būseną ir sudrebėjimą), naujiena šiame rinkinyje yra gebėjimas jausti sudrebėjimą. Tai vienas iš dažniausiai randamų jutiklių tokiuose įrenginiuose kaip išmanusis telefonas. Dėl šio jutiklio telefonas žino, kada, ką ir į kurią pusę reikia pasukti.
- **Lipdukai** skirti sužymėti LEGO detalių vietas dėžėje.

Visos detalės pritaikytos atlikti programinėje aplinkoje randamoms užduotims. Rinkiniu naudotis vienu metu gali iki 2 vaikų.

Programavimas ir aplikacijos

„WeDo 2.0“ programavimo programa suteikia puikią platformą, skirtą mokytis tiksliesiems mokslams ir siūlo inovatyvų būdą mokiniams mokytis, kaip modeliuoti realybę, realius gamtos mokslo reiškinius, kaip atlikti tiriamuosius eksperimentus ir kaip juos dokumentuoti. Visa tai yra pateikiama žaidimo būdu.



Gebėjimas programuoti yra pati geidžiamiausia 21-ojo amžiaus kompetencija. „WeDo 2.0“ programavimo aplinka moko jaunuosius išradėjus, kaip programuoti labai intuityviu būdu. Greitai ir lengvai parašytos programos suteikia momentinį rezultatą, atgaivindamos mokinių sukurtus kūrinius. Teisingai parašius programą, mokinių robotai gali judėti, jausti aplinką, skaičiuoti, transliuoti vaizdus, sekti istorijas ir daugybę kitų dalykų. Svarbiausia lavinama savybė – tai algoritminis mąstymas, gebėjimas planuoti į priekį ir suvokimas, kaip mąsto robotai ir kaip veikia programos. Programavimo aplinka yra grafinė, todėl mokiniai, naudodamiesi pelyte arba savo pirštais, gali tempti programavimo blokelius ir taip rašyti savo programas, susipažinti su pagrindinėmis programavimo funkcijomis.

Taip pat „WeDo 2.0“ programa leidžia Jums:

Įgyvendinti pirmuosius pradedančiojo projektus (4 pamokėlės, supažindinančios su pagrindiniais komponentais ir programavimo niuansais). Programuoti ir konstruoti Jūsų sugalvotus projektus. Prisijungti prie daugiau nei 40 valandų turinio su 17 skirtingų projektų įvairiausiems lygiams ir įvairiausiomis temomis. Ugdyti dokumentavimo kompetencijas (fotografuoti, aprašinėti, filmuoti savo rezultatus).

Techniniai reikalavimai

Rekomenduojami minimalūs techniniai reikalavimai personaliniam kompiuteriui:

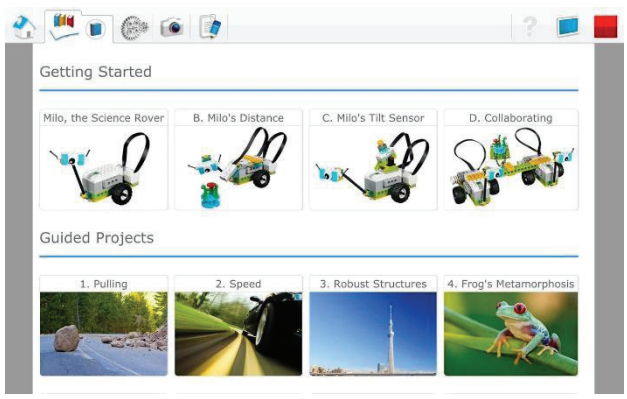
- Operacinė sistema – „Mac OSX 10.10“ arba „Windows 7 service pack 1“ (32/64-bit)
- Centrinis procesorius – „1.5GHz Intel® Core Duo“ arba spartesnis
- Laikina atmintis – 1.5 GB RAM arba daugiau
- Pastovi atmintis – 2 GB
- Bevielis ryšys – „Bluetooth 4.0 low power“
- Operacinės sistemos

Rekomenduojami minimalūs reikalavimai planšetiniam kompiuteriui:

- iOS Planšetės:
    - Įrenginiai: iPad 3, iPad Air 3, iPad Mini 3 arba naujesni
    - Operacinė sistemos versijos: iOS 8.1 arba aukščiau
- Android planšetės:
    - Planšetė su „Blueetooth lowe power“ technologija
    - Priekinė ir galinė kamera
    - 8 arba daugiau colių ekranas
    - Operacinė sistema: Android 4.4.2 arba naujesnė

Turinys ir mokomoji vertė

„WeDo 2.0“ programavimo programa yra 2 versijų: bazinė versija, kurioje yra 4 įžanginiai mokomieji užsiėmimai, supažindinantys su pagrindinėmis rinkinio funkcijomis, kaip valdyti motorus ir jutiklius. Taip pat visa programavimo aplinka rašyti savo programoms ir kurti savo projektus.



„WeDo 2.0“ nesutrumpintoje versijoje yra:

- 40 valandų trukmės visiškai paruoštų mokymosi projektų, apimančių mokslines sritis: biologiją, fiziką, mechaniką, inžineriją, gamtosaugą, pasaulio pažinimą, atsinaujinančius energijos šaltinius ir daugybę kitų.
- 17 integruotų projektų.
- Daugiau nei 20 skirtingų skaitmeninių instrukcijų unikaliems projektams.
- Projektai yra paruošti pagal nacionalines Didžiosios Britanijos, Jungtinių Amerikos Valstijų ir Vokietijos formalaus švietimo programas.

Integruotos programos pritaikytos formaliajam ugdymui, nes suskirstytos į 1 pamokos (45 minučių) veiklas. Tačiau tinkama ir neformaliojo ugdymo veikloms.

„Lego education Spike“ robotas

Roboto aprašymas

„LEGO Education SPIKE Prime“ rinkinys yra skirtas 6–8 klasių mokiniams. Derinant spalvingus LEGO statybinius elementus, lengvai naudojamą aparatūrą ir intuityvią „Scratch“ pagrindu sukurtą programavimo kalbą, „SPIKE Prime“ skatina mokinius mąstyti kritiškai ir spręsti sudėtingas problemas, nepriklausomai nuo jų mokymosi lygio. Nuo paprastų projektų iki neribotų kūrybinio dizaino galimybių „SPIKE Prime“ padeda mokiniams smagiai išmokti esminių STEAM ir kitų 21-ojo amžiaus įgūdžių, reikalingų norint neatsilikti nuo naujausių technologijų.

„SPIKE Prime“ sujungia LEGO kaladėles, programavimą „Scratch“ kalba ir STEAM mokymąsi problemų sprendimui, kuriam itin svarbu kritinis mąstymas ir pasitikėjimas savo jėgomis.

Pradedant nuo lengviausių „SPIKE Prime“ pamokų, jos puikiai telpa į pamokos laiką, įskaitant konstravimą ir programavimą.

Nuo lengvų pirmųjų pamokų iki neribotų kūrybinių dizainų, „SPIKE Prime“ moko kritiškai mąstyti, analizuoti duomenis ir juos panaudoti sudėtingų problemų sprendimui, o užduotys, susijusios su realaus pasaulio problemomis, tampa tik dar įdomesnės, nes mokiniai mato, kur praktiškai galima pritaikyti sprendinius.

„SPIKE Prime“ sistemos varikliukas yra programuojamas šakotuvas. Šis pažangus ir paprastas naudoti prietaisas turi 6 įvesties / išvesties prievadus, 5×5 šviesos matricą, „Bluetooth“ ryšį, garsiakalbį, 6 ašių giroskopą ir įkraunamą bateriją.

„SPIKE Prime“ rinkinyje taip pat yra labai tikslūs varikliai ir jutikliai, kurie kartu su daugybe spalvingų LEGO statybinių elementų leidžia mokiniams kurti įdomius robotus, dinaminius įrenginius ir kitus interaktyvius prietaisus.

Daugiau šakotuvo priedų, motorų ir sensorių, nauji dideli statymo elementai leidžia sumažinti statymo laiką ir skirti jo daugiau mokymuisi.

Patvarios laikymo dėžės ir 2 rūšiavimo padėklai padeda sutaupyti laiko renkantis detales ar susitvarkant darbo vietą, o mažesni padėklai padeda sutaupyti vietos, ypač kai stalo erdvė yra ribota.

Roboto komplektacija

21 pav. „Spike“ roboto komplektacija

Į komplektą įeina:

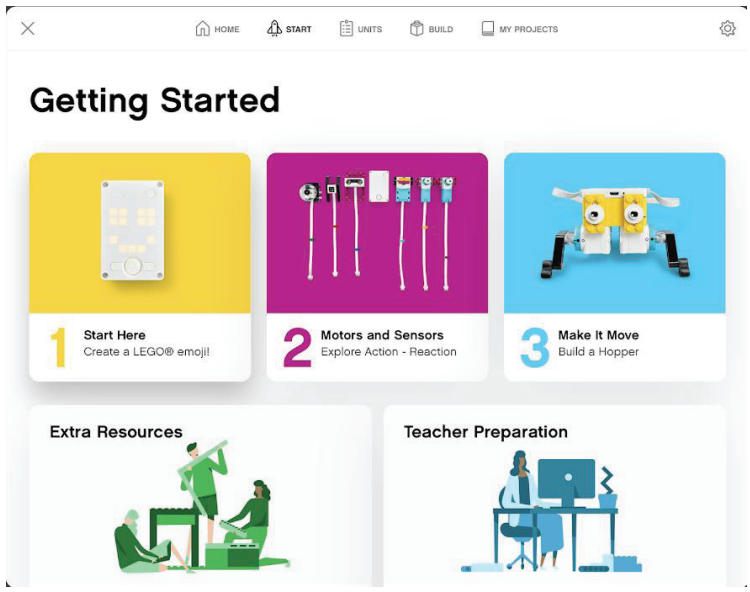
- Daugiau kaip 500 „LEGO Technic“ detalių. Įtrauktos ir naujos, niekada nematytos LEGO detalės!
- Naujas 3×3 rėmas leidžia lengvai keisti konstrukcijos kryptį.
- Nauja 2×4 kaladėlė, kurioje yra skersinės ašies skylė, leidžianti prijungti „Technic“ ir „LEGO System“ elementus, todėl „SPIKE Prime“ leidžia sujungti kitas LEGO kolekcijas ir suteikia daugiau kūrybinės laisvės.
- Nauja pagrindo plokštė užtikrina puikų prototipų paviršių, pirmą kartą buvo pasiūlyta su „Technic“ rinkiniu.
- Nauji rėmai leidžia pradėti statyti be jokių pasiruošimų ir statyti dar didesnius modelius.
- Nauji ratai lengvai montuojami su varikliu, tikslesni posūkiai ir geresnis manevringumas.
- Nauji vielos gnybtai su nauju spalvų asortimentu palengvina laidų sujungimų tikrinimą.

Turinys ir mokomoji vertė

Mokymosi tikslas yra įtraukti mokinius, kurie jau turi tris STEAM mokymosi vienetus ir yra orientuoti į inžineriją ir programavimą.

Išsamūs pamokų planai, kuriuose yra daug prieinamos interaktyvios medžiagos, padeda mokytojams pasiruošti pamokoms ir jaustis užtikrintai dėl dėstomo dalyko.

Supažindina savo mokinius su programavimu ir robotika paprastu ir intuityviu būdu su „LEGO Education SPIKE“ programėle. „SPIKE Prime“ planšetinių kompiuterių ir programavimo aplinka yra pagrįsta populiaria „Scratch“ kalba, kurią naudoja milijonai vaikų visame pasaulyje – itin paprasta naudoti!



Suprojektuota 6–8 klasių mokiniams ir optimizuota 45 minučių pamokoms, paspartina STEAM mokymąsi, nuosekliai įtraukdami mokinius kritiškai mąstyti ir išspręsti sudėtingas problemas nepriklausomai nuo jų mokymosi lygio.

Mokiniai kiekvienoje pamokoje sutelks dėmesį į vieną konkretų projektavimo proceso etapą:

- Problemos nustatymas
- Prototipų kūrimas
- Testavimas
- Įvertinimas

Varžybų skyriuje mokiniai bus supažindinti su robotų varžybų pasauliu, palaipsniui išmoks autonominių robotų konstravimo ir programavimo pagrindų naudojant sensorius.

Dirbdami komandoje, kurs varžybų robotą, jį testuos ir atnaujins programas, kurs naujus sprendimus, norėdami įveikti paskirtas misijas. Darbo metu tobulinami komandinio darbo ir bendradarbiavimo įgūdžiai pravers mokinių karjerai.

Išsamūs internetiniai pamokų planai ir daugybė interaktyvios pagalbinės medžiagos suteikia mokytojams viską, ko reikia, kad jaustųsi užtikrinti pamokų metu, nepriklausomai nuo jų patirties.

„SPIKE Prime“ įneša kūrybiškumą ir įsitraukimą į alumnų programas, robotų klubus, programavimo programas ir kūrėjų erdvę.

Robotų konkursų – FIRST LEGO lygos ir Pasaulinės robotų olimpiados – pasiruošimui naudojamas „SPIKE Prime“ išplėtimo rinkinys ir „Competition Ready“ rinkinys padeda mokiniams ir mokytojams pasiruošti, net jei jie yra robotikos naujokai.

Paskutinė konkurso skyriaus pamoka turi tiesioginį susiekimą su kasmetiniu FIRST LEGO lygos robotų žaidimu, todėl tai gali būti puiki varžybų kelionė klasėje.

„Lego education EV3“ robotas

23 pav. „LEGO Education EV3“ robotas

Robotų aprašymas

Šis rinkinys turi visas reikalingas dalis tam, kad būtų galima mokytis konstruoti, programuoti ir išbandyti robotų techniką. Į rinkinį įeina EV3 valdymo blokas, mažas, bet galingas kompiuteris, kuris padeda valdyti variklį ir jutiklius. Jis taip pat turi BT ir belaidį ryšį (NETGEAR Wifi WNA1100 bevielį-N 150) tam, kad galima būtų programuoti ir registruoti duomenis. Instrukcija, kaip sukonstruoti papildomus modelius naudojant šį rinkinį, yra įdiegta į programinę įrangą. Rinkinys yra tvirtoje dėžėje, o jo detalės yra išrūšiuotos taip, kad būtų paprasta naudoti ir saugoti. Baterijos kroviklis yra parduodamas atskirai.

Roboto komplektacija

Į rinkinį įeina:

- 3 interaktyvūs varikliai;
- Sukimosi ir ultragarso jutikliai;
- Spalvos / šviesos, giroskopo ir 2 liečiami jutikliai;
- Įkraunama baterija;
- Jungimo kabeliai;
- LEGO konstravimo detalės, 541 elementas;

Turinys ir mokomoji vertė

Tyrinėdami mokslinius dėsnius ir eksperimentuodami susipažinsite ir suvoksite pagrindinius fizikos dėsnius, įgysite patirties kuriant mechanines konstrukcijas ir išmoksite

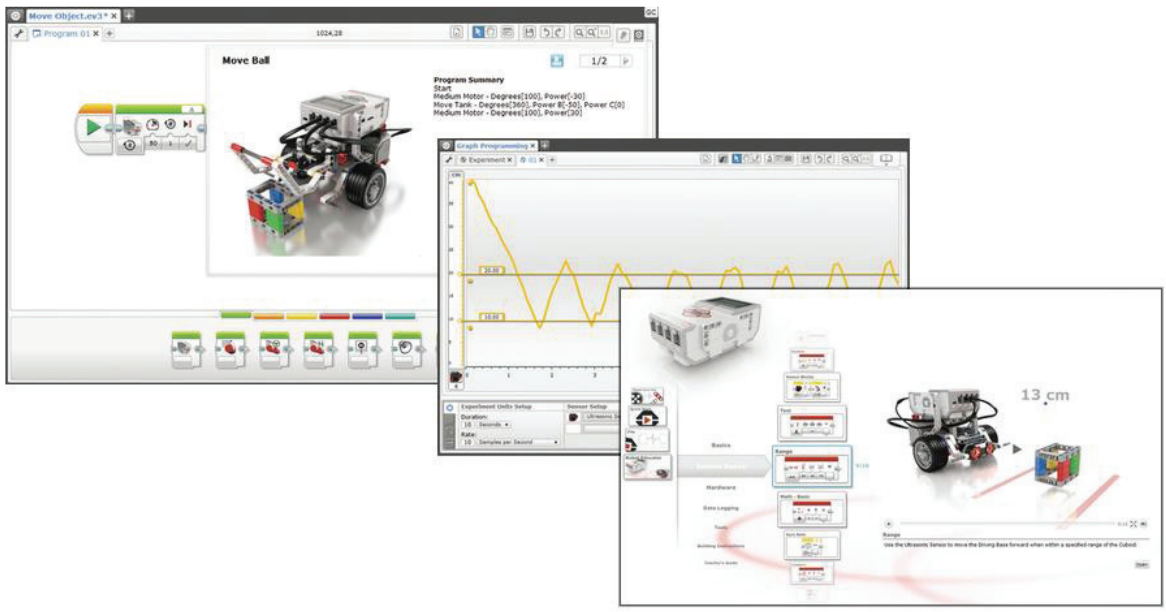
programuoti įvairiausius robotus nuo manipuliatorių iki robotų humanoidų. Rinkinys skirtas vidurinių klasių moksleiviams ir vyresniems.

LEGO centre MINDSTORMS EV3 roboto valdiklyje yra ne tik pažangus, bet ir kompaktiškas bei galingas kompiuteris, kuris leidžia autonomiškai kontroliuoti variklį ir rinkti duomenis iš jutiklių. Tai suteikia neribotą laisvę robotams kurti ir idėjoms realizuoti.

LEGO MINDSTORMS EV3 programinėje įrangoje yra integruotos 48 pamokos ir išsami naudotojo instrukcija, kuri neleis jums pasiklysti technologijų labirintuose.

Programavimas robotikoje nėra pati lengviausia užduotis, todėl „LEGO education“, bendradarbiaudama su LabVIEW ™ kūrėjais, sukūrė unikalią grafinio programavimo aplinką, kuri nereikalauja specialių programavimo žinių (programavimo kalbos sintaksės ir funkcijų). Pačius pirmus algoritmus sukursite akimirksniu naudodami intuityvią drag-n-drop programavimo aplinką.

Programavimo aplinka palaiko Windows ir OSX operacines sistemas bei mobiliuosius IOS įrenginius.



25 pav. EV3 roboto programos aplinka

LEGO MINDSTORMS EV3 programine įranga galėsite:

- Programuoti robotus;
- Stebėti ir aprašyti progresą naudojant integruotą redaktorių;
- Kurti ir redaguoti turinį;
- Mokytis iš elektroninio vadovėlio;
- Registruoti ir apskaičiuoti duomenis realiu laiku.

Su LEGO MINDSTORMS EV3 robotų konstruktoriumi, naudodami inovatyvias ir išmanias technologijas, sukursite ir suprogramuosite savo robotą, kuris padės jums geriau suprasti, kaip technologijos veikia realiame pasaulyje. Tai padės jums geriau suvokti ir interpretuoti dvimačius brėžinius ir iš jų kurti trimačius modelius (kurti, testuoti ir tobulinti; įgyti praktinės patirties naudojant matematines sąvokas, pvz., apskaičiuoti ir išmatuoti nubėgtą atstumą, laiką ir greitį).

„SumoBoy“ robotas

26 pav. „SumoBoy“ robotas

Roboto aprašymas

Roboto prototipų kūrimo komplektą sudaro robotas „SumoBoy“ ir prototipų kūrimo rinkinys, atveriantis galimybę įdomiai praleisti laiką ir galimybę įgyvendinti savo projektus. Rinkinys skirtas mokiniams ir entuziastams, ieškantiems būdo išmokti elektronikos ir programavimo pagrindų. „SumoBoy“ robotas sukurtas pagal tarptautines robotų sumo varžybų taisykles ir yra pasirengęs ten dalyvauti. Į prototipų kūrimo rinkinį įeina lenta be litavimo grandinės konstrukcijai, leidžianti pakartotinai panaudoti skirtingų konstrukcijų komponentus ir naudoti „Arduino Micro“ mikrovaldiklį programavimui. Rinkinys sukurtas naudojant Rygos technikos universiteto Robotikos klubo, vadovaujančio Latvijos robotikos bendruomenei nuo 2008 m., sukauptą mokymo patirtį.

Robotai valdomi gerai žinomu „Arduino Micro“ mikrovaldikliu. Leidžia panaudoti didžiules „Arduino“ bendruomenės žinias. Robotas bus patrauklus tiek pradedantiesiems, tiek patyrusiems entuziastams.

Roboto komplektacija

Roboto komplektą sudaro:

- Plieninės apsaugos;
- Mechaniškai apdoroti aliuminio ratų diskai;
- Specialios silikoninės kompozicijos gumos ratai;
- Penki integruoti į priekį nukreipti jutikliai;
- Specialios konstrukcijos varikliai ir reduktoriai;
- „4 dip“ jungikliai strategijos pasirinkimui;

- „Arduino Micro“ valdiklis;
- Saugi „LiFePO4“ baterija;
- Integruota įkrovimo schema;
- Dokumentacija: daugiau nei 30 pratimų, skirtų mokytis elektronikos ir programavimo;
- Palaiko IR nuotolinio valdymo pultą, atitinkantį tarptautines taisykles;

Pritaikymo galimybės

- Robotikos būreliams;
- Techninės kūrybos centrams;
- Tarptautinio masto robotikos varžybų entuziastams;
- Patraukli informatikos mokymo įranga;
- Puikus praktinis pagalbininkas, mokantis elektronikos pagrindų;
- Pažintinė pramoga su draugais;
- Puiki dovana paaugliui ar studentui, pradedančiam karjerą inžinerijoje.

Techninė specifikacija

- Matmenys AxPxP 42x98x98 mm;
- Svoris 359,80 g;
- Maitinimo šaltinis 6V – 12V;
- Varikliai 6V x 1,2 A;
- Maks. greitis 0,86 m/s;
- Baterijos tipas „LiFePO4“;
- Padangos silikoninės;
- Mikrovaldiklis „Arduino Micro“.

IV. Programavimo pagrindai

Šiame skyriuje pateikiami programavimo pagrindai C kalba. Programavimas yra neatsiejama robotikos dalis. Kiekvienas robotas turi turėti įdiegtą programą, kad galėtų atitinkamai reaguoti į aplinkos pokyčius ir atlikti veiksmus. C kalba pasirinkta todėl, kad „Arduino“ naudoja C ir C++ programavimo kalbos „mišinį“. Su „Arduino“ žaisti galima kaip su LEGO – dėlioji detales ir iš karto matai rezultatą. Be to, „Arduino“ ypatingas tuo, kad jis nėra toks brangus ir prieinamas įsigyti visoms mokykloms.

Įvadas

Programavimo kalba – kompiuteriui suprantama dirbtinė kalba, skirta programoms užrašyti. Programos valdo elektroninius įrenginius. Programavimo kalbos susideda iš sintaksinių bei semantinių taisyklių, kurios atitinkamai nusako jos struktūrą ir prasmę. Programavimo kalbų sintaksinės ir semantinės taisyklės yra tam tikra forma aprašomos jų specifikacijose.

Programavimo kalbų savybės

Funkcionalumas: Programavimo kalba yra kalba, skirta užrašyti programas kompiuteriui, kurios leistų panaudoti kompiuterio resursus, atliekant tam tikrus skaičiavimus arba algoritmus ir leistų valdyti išorinius įrenginius.

Paskirtis: Programavimo kalbos komandos leidžia žmonėms bendrauti su kompiuteriais ar kitais elektroniniais įtaisais. Kai kurios programavimo kalbos yra naudojamos bendraujant vienam įrenginiui su kitu.

Programavimo kalbų skirstymas

Skirstymas pagal vykdymo tipą: Vykdymo tipas – būdas, kaip programos kodas paverčiamas kompiuteriui suprantamomis instrukcijomis ir vykdomas. Teorinės kalbos – aprašytos, tačiau nevykdomos. Kompiliuojamos kalbos – kompiliatorius iš pradinio teksto sukuria vykdomą programą (pvz., C, „Pascal“). Interpretuojamos kalbos – jų išeities tekstas analizuojamas vykdant (pvz., BASIC, PHP, „Perl“, „Python“, „Ruby“). Mišrios kalbos – naudojamas ir kompiliatorius, ir interpretatorius (pvz., „Java“; taip pat žr. JIT).

Skirstymas pagal abstrakcijos lygį: Kuo kalba abstraktesnė, tuo ji yra aukštesnio lygio. Mašininis kodas – instrukcijų ir duomenų rinkinys, kurį tiesiogiai vykdo mašina. Tai

pati seniausia programavimo kalba, dažnai žemiausio lygio, nors dabartiniai procesoriai dar turi ir mikrokodo lygmenį. Žemo lygio kalbos (pvz., asembleris). Asembleryje operacijos užrašomos žmogui suprantamesnėmis ir labiau įsimintinomis raidėmis, ne dvejetainiu kodu. Sisteminės kalbos (pvz., C) ir objektinės kalbos (pvz., „SmallTalk“).

Skirstymas pagal sudėtingumą: BASIC paprastų algoritmų kūrimui. C, C++ sudėtingų algoritmų kūrimui. „Pascal“, „Python“ programomis galima kurti ir paprastus, ir sudėtingus algoritmus, vidutinio sudėtingumo programavimo kalbos.

Skirstymas pagal paskirtį: Kai kurios kalbos dažniau sutinkamos tam tikrose, joms būdingose, nišose. Vienos kalbos buvo sukurtos specialiai tam tikros srities problemoms spręsti, kitos išpopuliarėjo savo srityje dėl istorinių aplinkybių. Keletas išskirtinių sričių su joms būdingomis kalbomis: Sisteminis programavimas (operacinių sistemų, kompiliatorių) – C, C++; Interneto svetainių programavimas – PHP, „Perl“, „Python“, „Adobe ColdFusion“; Matematiniai skaičiavimai – PROLOG, „Fortran“, MATLAB, MAPLE; Mokymas – „Pascal“, „Logo“.

Programavimo C kalba pagrindai

Pirmą kartą C kalbą aprašė jos autoriai B. W. Karnighan ir D. M. Ritchie 1978 metais išleistoje knygoje „C programavimo kalba“. C kalbos variantas su klasėmis pavadintas C++ kalba. Ši kalba buvo kaip instrumentas programuotojams – praktikams. C++ yra nauja programavimo kalba, pritaikyta sudėtingų programų sistemų ir instrumentinių programavimo priemonių kūrimui, panaudojant objektinio programavimo technologiją. Tačiau šioje kalboje išsaugotas glaudus ryšys ir su klasikine C – efektinga programavimo kalba. Ji leidžia geriausiai išnaudoti kompiuterinius resursus. C kalba parašytos programos yra kompaktiškos ir greitai vykdomos. C – mobilioji programavimo kalba. Tai reiškia, jei programa parašyta šia kalba, ji gali būti lengvai, su nedideliais pataisymais arba visai be jų, perkeliama į kitas skaičiavimo sistemas, pvz., iš IBM kompiuterio perkelti programos veikimą į UNIX. C – galinga ir lanksti programavimo kalba. Didelė dalis galingos UNIX ir „Windows“ operacinės sistemos parašyta C kalba. C kalba parašytos programos naudojamos fizikiniams ir techniniams uždaviniams spręsti, taip pat ir animacijai kurti. C – turi galimybę panaudoti daugybę valdančiųjų konstrukcijų, kurios paprastai asocijuojasi su asembleriu. Asemblerio programavimo kalba labai sudėtinga – tai skaičiai ir kodai, kuriuos suprasti net specialistui sunku, kadangi ji rašoma procesoriaus kalba.

Sintaksė ir terminai

C kalba sudaryta iš įvairių į žodžius panašių vienetų, vadinamų žodžiais arba tokenais. Tokenų grupės sudaro sakinius, frazes, išraiškas ir kitas kalbos dalis. Kompiliatorius skaitydamas programos tekstą suskirsto jį į tokenus ir tarpus. Tarpas (whitespace) yra bendras

vardas eilei ASCII simbolių ir C kalbos konstrukcijų, tai: tarpelis, horizontali ir vertikali tabuliacija, naujos eilutės simbolis ir komentarai. Komentarai C kalboje yra dviejų rūšių: turintys atidarymo (/*) ir uždarymo (*/) simbolius ir komentaras, prasidedantis (//) ir pasibaigiantis eilutės pabaigos simboliu. Komentarai naudojami programos paaiškinimams užrašyti. Programa su komentarais yra aiškesnė, lengviau suprantama. Komentarai yra naudingi ne tik kitam žmogui, nagrinėjančiam programą, bet ir programos autoriui, nes praėjus laikui viską prisiminti yra neįmanoma, o gerai parašyti komentarai žymiai sutrumpina laiką, kurio reikia programai suprasti. Šiuolaikinės programos yra sudėtingos, jas sudaro šimtai, o kartais ir šimtai tūkstančių eilučių bei dešimtys ar šimtai failų. Tokias programas kuria kolektyvai, todėl komentarai yra ne tik pageidaujami, bet tiesiog privalomi.

Vardai

C kalboje vardus turi kintamieji, konstantos, funkcijos ir kitos kalbos dalys arba objektai. Vardai turi būti unikalūs tam tikroje programos ar funkcijos dalyje. Dažniausiai vardai būna unikalūs visoje programoje ar funkcijoje. Vardas turi prasidėti lotyniškos abėcėlės raide arba pabraukimo (underscore _) simboliu, visi kiti vardo simboliai gali būti parinkti iš šių grupių: a ... z mažosios lotyniškos raidės, A ... Z didžiosios lotyniškos raidės, skaičiai nuo 0 iki 9, _ pabraukimas (underscore). Reikėtų parinkti prasmingus vardus, tada programa tampa aiškesnė ir sugaištama mažiau laiko jos veikimui suprasti. Yra ir „nusistovėję“ vardai, t. y. tokie vardai, kuriuos naudoja dauguma 10 programuotojų.

Skyrybos ženklai

Skyrybos ženklai (punctuators), kai kada dar vadinami skirtukais (separators), C kalboje yra tokie: [] () {} , ; : ... * = #

- Laužtiniai skliausteliai (brackets) [ ]: pažymi vienmačius ir daugiamačius masyvus.
- 4.2 Lenktiniai skliausteliai (parentheses) ( ): yra naudojami grupuoti išraiškas, izoliuoti sąlygines išraiškas, pažymėti funkcijos iškvietimą, jais taip pat apgaubiami funkcijos parametrai. Lenktiniai skliausteliai yra rekomenduojami makroprocedūrose, kad būtų išvengta klaidų jas išplečiant.
- 4.3 Figūriniai skliausteliai (braces) { }: nurodo mišrių veiksmų ar veiksmų grupės pradžią ir pabaigą, kitaip tariant, apgaubia veiksmų grupę. Atidarantis skliaustelis pažymi veiksmų grupės pradžią, o uždarantis skliaustelis grupės pabaigą, todėl po jo kabliataškis (;) nereikalingas. Išimtį sudaro struktūrų ar klasių deklaracija. Figūriniai skliausteliai taip pat apgaubia funkcijos veiksmus. Funkcijos veiksmas prasideda už atidarančiojo skliaustelio ir baigiasi prieš uždarantįjį skliaustelį. Skliaustelių trūkumas ar perteklius taip kaip ir kabliataškiai yra gan dažna programavimo klaida. Nors šiuolaikiniai kompiliatoriai gerai seka skliaustelius, tačiau pranešimas apie klaidą ne visada būna aiškus, o kartais apie skliaustelius net nekalbama. Programuojant reikia laikytis vienos taisyklės – jei skliaustelį atidarei, reikia ir uždaryti. Tam labai pasitarnauja tvarkingas programavimas, dar vadinamas programavimo stiliumi, kai veiksmų grupės atitraukiamos nuo krašto, atitinkamai atitraukiami skliausteliai, vienodai naudojami tarpeliai ir t. t.

- Kablelis (comma) (,): atskiria funkcijos argumentus, vienodo tipo kintamųjų vardus ir kintamųjų reikšmes.
- Kabliataškis (semicolon) (;): yra reiškinio ar išraiškos pabaigos ženklas. Kiekvienas C kalbos reiškinys ar išraiška visada baigiasi (;), įskaitant ir „tuščią operatorių“, kuris žymimas (;). Kabliataškis dažnai naudojamas kaip tuščias operatorius cikluose laiko intervalų sudarymui. Tuščias operatorius yra sutransliuojamas į asemblerio NOP (No OPeration) ir yra vykdomas per vieną procesoriaus taktą. Jei parašyti keli kabliataškiai, bus atliekami keli tušti operatoriai ir tai užtruks tiek pat procesoriaus taktų. Tokie ciklai naudojami laiko intervalų formavimui, dažniausiai senesniuose mikroprocesoriuose ir mikrovaldikliuose, kurie neturi arba turi nepakankamai laikmačių. Praleistas kabliataškis išraiškos gale yra viena dažniausiai pasitaikančių klaidų tarp pradedančiųjų, ir ne tik, programuotojų.
- Dvitaškis (colon) (:): parodo žymę programoje, t. y. pažymi vietą, į kurią gali būti perduotas programos valdymas. Jo naudojimas yra laikomas prastu programavimo stiliumi, nes gali sukelti begalinį ciklą, kurį sunku aptikti.
- Daugtaškis (ellipsis) (…): trys taškai be tarpų. Daugtaškis naudojamas pažymėti formaliems parametrams funkcijų prototipuose. Jis pažymi, kad funkcija turi kintamą parametrų skaičių arba, kad kintamieji yra skirtingų tipų.
- Žvaigždutė (asterisk) (*), priklausomai nuo to, kur yra naudojama, gali turėti skirtingas reikšmes.
    - naudojama kaip daugybos ženklas
    - naudojama rodyklių į kintamuosius deklaracijai
    - naudojama įvairaus gylio rodyklėms deklaruoti
    - naudojama ir reikšmei iš rodykle aprašyto kintamojo gauti
- Lygybės ženklas (equal, initialize) (=): naudojamas inicializuoti (suteikti reikšmes) kintamiesiems ir atskiria kintamojo deklaraciją nuo kintamojo reikšmių. Kitaip negu matematikoje, kur ženklas „=“ reiškia, kad reiškinys kairėje lygybės pusėje yra lygus reiškiniui dešinėje pusėje, C kalboje lygybė reiškia, kad kintamajam kairėje pusėje yra priskiriama (suteikiama) dešinėje pusėje esančio reiškinio reikšmė. C kalbos standartas nustato, kad funkcijose pirmiausia yra deklaruojami kintamieji ir tik po to galima rašyti programos veiksmus. Kaip ir visur yra keletas išimčių. Lygybė taip pat naudojama kaip priskyrimo operacija išraiškose.

Priešprocesoriaus komandos

Priešprocesoriaus komandos (preprocessor commands) yra pažymimos numerio ženklu # (įvairiose šalyse jis yra vadinamas skirtingai: pound sign, sharp, ceche. Programavimo literatūroje sutinkami visi), žymi preprocesoriaus direktyvą, bet gali būti naudojamas kaip paprastas simbolis tekstinėse eilutėse ir komentaruose. Preprocesoriaus direktyvai atpažinti kai kuriuose kompiliatoriuose šis ženklas visada turi būti pirmas simbolis eilutėje. Preprocesoriaus direktyvos dažniausiai būna programos teksto pradžioje, tačiau gali būti naudojamos bet kurioje programos vietoje. Pagrindinės preprocesoriaus direktyvos yra:

- *# null* direktyva: eilutė, kurioje yra tik šis simbolis, yra ignoruojama.
- *#define, #undef #define* direktyva: nustato makro arba konstantas. Makro yra mechanizmas, leidžiantis pakeisti vieną tokeną kitu arba keliais su argumentais ar be argumentų.
- *#ifdef, #ifndef* sąlyginės kompiliacijos direktyva: leidžianti kompiliuoti programos tekstą priklausomai nuo to, ar identifikatorius nustatytas, ar ne.
- *#if, #elif*, *#else* ir *#endif* prekompiliatorius turi sąlygines direktyvas, kuriomis galima keisti programos tekstą priklausomai nuo sąlygų.
- *#include* direktyva: įjungia į programos teksto kompiliavimą papildomus įrašus, dažniausiai antraščių failus arba kitus (papildomus arba išorinius) programų failus.
- *#error* direktyva: nutraukia kompiliavimą ir grįžta su įrašytu klaidos pranešimu.

Aprašytos makrokomandos

Šios makrokomandos yra aprašytos ir negali būti pakeistos. __LINE__ Dešimtainis skaičius, atitinkantis programos teksto eilutę. __FILE__ Programos teksto failo vardas (eilutė). __DATE__ Programos failo kompiliavimo data (eilutė). Formatas „mmm dd yyyy“ toks pats, kaip generuojamas funkcijos asctime(). __TIME__ Programos failo kompiliavimo laikas (eilutė). Formatas „hh:mm:ss“ toks pats, kaip funkcijos asctime(). __STDC__ Dešimtainė konstanta lygi 1. Naudojama parodyti, kad naudojamas standartinis C kalbos kompiliatorius. Jos dažniausiai yra naudojamos programų versijų kontrolei, derinimo pranešimuose, derinant sudėtingas programas, programų serverių pranešimų failų įrašams formuoti.

Duomenų tipai

Skaičiai būna sveikieji ir realieji. Realieji skaičiai yra trupmeniniai skaičiai. Paprastosios trupmenos, pavyzdžiui, viena trečioji (1/3), kompiuteriuose dažniausiai nenaudojamos. Kompiuteriuose dažniausiai naudojamos dešimtainės trupmenos realiesiems skaičiams atvaizduoti. C kalboje, kaip ir daugelyje kitų programavimo kalbų, visi kintamieji turi turėti tipą. Tipas – tai ne tik užimamos kompiuterio atminties dydis, bet ir skaičių ar simbolių atvaizdavimo galimybė, o taip pat skaičiavimo tikslumas. Taip kaip kintamieji, tipus turi turėti ir funkcijos, nes jos dažniausiai grąžina reikšmes jas iškvietusiai programai. Griežtas kintamųjų tipizavimas gali atrodyti nereikalingas ir net trukdantis, beje, taip ir yra. Tačiau toks griežtumas labai padidina programų patikimumą, o taip pat padeda išvengti klaidų programuojant.

Kintamųjų deklaracija: Prieš naudojant kintamąjį ar funkciją jį pirmiausiai reikia deklaruoti, pranešti programai apie jo egzistavimą. Duomenų tipas yra suteikiamas deklaruojant kintamąjį ar funkciją.

Kintamojo deklaracijos sintaksė yra:

duomenų-tipas: vardas
duomenų-tipas: vardas = reikšmė

Funkcijų deklaracija kiek sudėtingesnė, bet iš esmes tokia pat kaip ir kintamųjų. Funkcijos deklaracijos sintaksė:

duomenų-tipas: funkcijos_vardas (parametras1, ...)

Duomenų tipas, nurodytas prieš funkcijos vardą, nurodo, kokio duomenų tipo atsakymą grąžins funkcija. Duomenų tipai, nurodyti už funkcijos vardo lenktiniuose skliausteliuose, nurodo, kokio tipo parametrai turi būti perduodami funkcijai.

Baziniai duomenų tipai: Skaičiavimams 8 bitų pakanka retai, nes aštuoni bitai yra skaičius nuo 0 iki 255 arba nuo -127 iki +127 arba ASCII simbolis. Todėl C kalboje yra daugiau duomenų tipų, skirtų įvairiems skaičiavimams ir loginiams veiksmams. Pagrindiniai C kalbos duomenų tipai pateikti lentelėje žemiau.

Duomenų tipus gali susikurti ir pats vartotojas. Tuo tikslu C kalboje yra numatyta priemonė – raktinis žodis „typedef“, nurodantis, kad yra nustatomas naujas duomenų tipas. Skirtingi duomenų tipai yra reikalingi saugoti skirtingo dydžio skaičiams ir skirtingo tikslumo skaičiavimams.

TIPAS	DYDIS BITAIS	RIBOS		PANAUDOJIMAS
unsigned char	8	0	255	Maži sveikieji teigiami skaičiai ir visi ASCII simboliai
Char	8	-127	127	Maži sveikieji teigiami ir neigiami skaičiai ir pagrindiniai ASCII simboliai
Enum	16	-32,768	32,768	Tvarkinga sveikųjų skaičių eilė
unsigned short int	16	0	65,535	Nedideli sveikieji skaičiai ir teigiamų skaičių eilė
short int	16	-32,768	32,768	Nedideli sveikieji skaičiai su ženklu
unsigned int	32	0	4,294,967,295	Dideli sveikieji skaičiai ir teigiamų skaičių eilė
Int	32	2,147,483,648	2,147,483,648	Dideli sveikieji skaičiai su ženklu
unsigned long	32	0	4,294,967,295	Labai dideli sveikieji skaičiai. Astronominiams atstumams

Long	32	2,147,483,648	2,147,483,648	Labai dideli skaičiai su ženklu
Float	32	3.4x10-38	3.4x1038	Nedidelio tikslumo realieji skaičiai. Moksliniams skaičiavimams 7 skaitmenų tikslumu
double	64	1.7x10-308	1.7x10 308	Didelio tikslumo realieji skaičiai. Moksliniams skaičiavimams 15 skaitmenų tikslumu
long double	80	3.4x10-4932	3.4x10 4932	Labai didelio tikslumo realieji skaičiai. Finansiniams skaičiavimams 19 skaitmenų tikslumu

Duomenų tipas (bool): turi dvi fiksuotas reikšmes „true“ ir „false“. Tipas „bool“ yra priskiriamas sveikųjų skaičių tipams. Reikšmę „**false**“ atitinka skaičius **0**, o reikšmę „**true**“ atitinka skaičius **1**.

Duomenų tipas (void): C kalboje yra dar vienas specialus duomenų tipas „void“. Šis tipas naudojamas:

- Pažymėti, kad funkcija neturi parametrų, funkcijos deklaracijoje: int func(void);
- Deklaruojama funkcija negrąžina jokios reikšmės: void func(int a);
- Kaip bendra rodyklė (generic pointer) į bet kurį tipą: void *ptr;
- Tipų priskyrimui (typecasting): extern int errfunc().

Rodyklės (pointers): Rodyklė yra specialus kintamojo tipas, nurodo vietą arba adresą kito kintamojo arba tašką, kur kintamasis prasideda. Kintamojo rodyklė sudaroma radus adresą, kur šis kintamasis yra kompiuterio atmintyje. Tam C kalboje yra specialus operatorius **&**.

NULL: Nors iš tikrųjų tai yra skaičius „0“, tačiau NULL tipas yra neapibrėžtas (void), todėl ši reikšmė yra naudojama su įvairaus tipo duomenimis. Viena iš NULL reikšmių yra ta, kad rodyklė į kintamąjį ar objektą nenustatyta.

Konstantos (const): Konstanta yra toks duomenų tipas, kuriam vieną kartą priskyrus reikšmę, jos daugiau pakeisti nebegalima.

Specialūs C kalbos simboliai:

- **b** ištrinti simbolį už kursoriaus ir nustatyti kursorių į jo vietą [backspace BS]
- **f** naujas puslapis [form feed FF (also clear screen)]
- **n** nauja eilutė [new line NL (like pressing return)]
- **r** nustatyti kursorių į eilutės pradžią [carriage return CR (cursor to start of line)]

- **t** horizontali tabuliacija, nutylint 8 pozicijos [horizontal tab HT]
- **v** vertikali tabuliacija [vertical tab (not all versions)]
- **"** dvigubos kabutės [double quotes (not all versions)]
- **'** viengubos kabutės [single quote character ']
- \\ dešinysis įstrižas brūkšnys [backslash character \]
- **ddd** aštuonetainis kodas gali būti ir ASCII
- **xddd** šešioliktainis kodas gali būti ir ASCII

Vartotojo tipas (typedef): C kalba leidžia aprašyti savo sukurtą tipą ar pervadinti jau esantį, naudojant „typedef“ direktyvą.

Duomenų tipo pakeitimas (type cast): Kaip buvo minėta, C kalboje visi kintamieji ir funkcijos yra tipizuoti, t. y. visi kintamieji, konstantos ir funkcijos turi priklausyti kokiam nors duomenų tipui. Tačiau kartais reikia pakeisti kintamojo duomenų tipą, dažniausiai tai pasitaiko, kai reikia pasinaudoti funkcija, kurios argumentai yra kitokio tipo negu turimas kintamasis.

Duomenų struktūros

Baziniai duomenų tipai yra duomenų struktūros, įdiegtos kuriant programavimo kalbą. Kaip ir daugelyje programavimo kalbų, C kalboje yra šie baziniai duomenų tipai: simbolinis (char), sveikas (short, integer, long), realusis (float, double, long double), dvejetainis (bool), rodyklės tipas (pointer) ir du specialūs duomenų tipai – tuščias (void) ir nulis (NULL). Tačiau vien šių duomenų tipų kartais nepakanka. Todėl naudojamos duomenų struktūros:

Masyvai (Arrays): C kalboje masyvus galima sudaryti iš bet kurio tipo kintamųjų. Masyve gali būti tik vieno tipo kintamieji. Masyvai gali būti vienmačiai ar daugiamačiai. Masyvai gali būti inicializuojami deklaruojant arba po deklaracijos. Kai inicializuojami keli masyvo elementai, jie atskiriami kableliais ir apgaubiami figūriniais skliausteliais. Masyvai yra labai dažnai naudojami programose, kadangi masyvas yra sudarytas iš to paties tipo elementų ir yra labai patogu pasiekti bet kurį jo elementą. Kiekvienas masyvo elementas turi tam tikrą indeksą, kuris priklauso nuo vietos, kurioje yra elementas. Masyvuose labai patogu saugoti kokius nors vienarūšius duomenis, pavyzdžiui, grupės ar kurso visų egzaminų rezultatus arba kokius nors statistinius rezultatus. Skirtingų rūšių elementus galima surašyti į skirtingus masyvus, o sąryšiui tarp skirtingų masyvų naudojami indeksai. Tuo pačiu indeksu skirtinguose masyvuose reikia saugoti vieno objekto duomenis. Panaudojant ciklus labai patogu apdoroti duomenis, surašytus masyvuose. Pirmojo masyvo elemento indeksas yra 0. Indeksas dažniausiai yra kintamasis, kurio tipas turi būti „int“. Teksto apdorojimui jis yra surašomas į simbolių („char“ tipo) masyvą. Simbolių masyvas dar vadinamas eilute, o senesnėje literatūroje vadinamas literatu. C kalboje eilutė yra vienintelis būdas teksto apdorojimui. Deklaracijos metu labai paprasta inicializuoti simbolių eilutę (priskirti reikšmę), pakanka tekstą apgaubti dvigubomis kabutėmis ir figūriniais skliausteliais.

Struktūros (Structures): Duomenų struktūros gali sujungti kelių skirtingų tipų duomenis. Struktūros aprašymo sintaksėje yra naudojamas C kalbos raktinis žodis „struct“. Struktūros dalyviai (elementai) pasiekiami naudojant išrinkimo operatorius (.) ir (–>). Rodyklė užrašoma iš simbolio minus (–) ir ženklo daugiau (>) be tarpelio (–>). Operatorius (.) yra vadinamas tiesioginio priėjimo (direct access) ir naudojamas pasiekti struktūros, pažymėtos s, dalyviams, o operatorius (–>) netiesioginio priėjimo (indirect access) ir naudojamas pasiekti tos pačios struktūros, tik pažymėtos kaip rodyklė ***sptr**, dalyviams.

Junginiai (Unions): Junginiai savo sintakse yra labai panašūs į struktūras. Pagrindinis skirtumas yra tas, kad tuo pačiu momentu gali būti „aktyvus“ tik vienas junginio dalyvis. Junginio dydis toks, koks yra didžiausio junginio dalyvio dydis. Deklaruojant junginį yra naudojamas C kalbos raktinis žodis **union**.

Bitų laukas (bit field): Kitas į struktūrą panašus tipas. Bitų laukas gali turėti skaičius be ženklo ir su ženklu. Bitų laukas patogus naudoti mikrovaldiklių ir mikroprocesorių būsenos registrų analizei ir valdymo registrų reikšmių nustatymui. O taip pat, kai reikia „suspausti“ didelį kiekį duomenų išlaikant tam tikrą struktūrą. Pavyzdžiui, telefoninių pokalbių apskaitos duomenyse: telefono numerį pakeitus iš simbolinio (20 ženklų) į skaičių pakanka 62 bitų, 6 skambučio rūšis galima užkoduoti 3 bitais, 4 bitais galima užkoduoti 16 mokėjimo planų, 3 bitais paros tarifus, šešiaženklį pokalbių laiką 18 baitų, iš viso 90 bitų (12 baitų). Naudojant simbolinį formatą, reikėtų 30 baitų, skaitiniam – 15 baitų. Jei per vieną dieną įrašoma vienas milijonas įrašų, lyginant su skaitiniu formatu yra sutaupoma trys megabaitai per dieną.

Išvardijimo (enum) tipas: Išvardijimo tipas „enum“ yra kilęs iš anglų kalbos trumpinio „enumerated data“. Aukšto lygio kalbose yra naudojami žodžiai vietoje skaičių ir raidžių. Tam ir yra skirtas **enum** tipas.

Kintamieji: lokalūs ir globalūs (variables)

Programavime kintamieji yra reikalingi duomenims saugoti, jie yra pažymimi vardais, kurie dar vadinami identifikatoriais. C kalboje duomenų tipai yra naudojami kintamųjų deklaracijoje. Kintamojo deklaracija rezervuoja vietą kompiuterio atmintyje tiek baitų, kiek yra baitų kintamojo duomenų tipas. Kintamieji, kurie yra deklaruoti funkcijoje, yra lokalūs kintamieji ir yra prieinami tik toje funkcijoje, kurioje yra deklaruoti. Globalūs kintamieji turi būti deklaruoti prieš main() funkciją.

Operatoriai

Operatoriai yra simboliai, kurie nurodo, kokius veiksmus reikia atlikti su kintamaisiais, tarp kurių jie yra parašyti. C kalboje yra ne vien aritmetiniai operatoriai, bet ir loginiai, sąlyginiai, operacijų su bitais ir kiti. C++ kalbai realizuoti yra papildomi operatoriai objektų klasių ir klasių dalyvių priskyrimui.

Aritmetinių **veiksmų operatoriai:** Veiksmų operatoriai C kalboje yra šie: = + – * / % ++ – –

- **Priskyrimo operatorius (=):** Kitaip nei matematikoje, šis operatorius reiškia ne lygybę, bet priskyrimą, arba reikšmės kintamajam suteikimą. Kairėje lygybės pusėje esančiam kintamajam yra priskiriama dešinėje lygybės pusėje esančio reiškinio reikšmė.
- **Atimties operatorius (–):** Iš reiškinio, esančio kairėje operatoriaus (–) pusėje, atima reiškinį, esantį dešinėje pusėje.
- **Ženklo pakeitimas (–):** Pakeičia reikšmės ženklą į priešingą.
- **Daugybos operatorius (*):** Sudaugina reiškinius, esančius abiejose operatoriaus „*“ pusėse.
- **Dalybos operatorius (/):** Kairėje dalybos ženklo pusėje esantį reiškinį padalina iš dešinėje pusėje esančio reiškinio. Sveikųjų skaičių dalybos rezultatas visada yra sveikas skaičius. Jei dalmuo arba daliklis yra skaičius su slankiu kableliu, rezultatas bus skaičius su slankiu kableliu.
- **Liekanos operatorius (%):** Grąžina sveikųjų skaičių dalybos liekaną.
- **Operatoriai ++ (increment) ir – – (decrement):** Didinimo operatorius ++ (increment) turi kintamojo reikšmę.
- **Bitų operatoriai:** C kalboje yra grupė operatorių darbui su bitais.

Postūmio operacijos (shift): Kad būtų paprasčiau, nagrinėsime, kaip vyksta postūmio operacijos baite (8 bitų grupėje). Kiekvienas bitas yra atvaizduojamas atskirame langelyje antroje eilutėje, o viršutinėje eilutėje yra surašytos bitų vertės (svorio koeficientai). Kodas bitais atitinka verčių ir atitinkamų bitų reikšmių sandaugų sumą.

Bitų loginės operacijos (AND, OR, XOR, COMPLEMENT): Bitų loginės operacijos kartais dar vadinamos operacijomis su bitų kaukėmis (bit mask).

AND & a = a & m a &= m
OR | a = a | m a |= m
XOR ^ a = a ^ m a ^= m
NOT ~ a = ~a a = ~a

Loginiai operatoriai: Loginiai operatoriai reikalingi, kai programoje, esant skirtingoms sąlygoms, kurios yra aprašomos kintamųjų reikšmėmis, reikia atlikti skirtingus veiksmus. Pavyzdžiui, visi žinome, kad dalyba iš nulio negalima. Bet programose gali pasitaikyti tokių situacijų, kuriose daliklis pasidaro lygus nuliui. Tada kompiuteryje kyla perpildymo klaida. Taigi norint tokios situacijos išvengti, reikia patikrinti, ar daliklis yra lygus nuliui. Tokiems ir panašiems atvejams yra naudojami loginiai operatoriai. C kalboje loginiuose operatoriuose yra naudojami šie raktiniai žodžiai: if, else, switch, case, break, default. Loginiuose operatoriuose sąlygoms patikrinti naudojamos sąlygų operacijos, kurių rezultatas yra loginis teigimas (true) arba loginis neigimas (false).

Sąlygų operacijos yra naudojamos loginiuose operatoriuose. Jos yra tokios:

< mažiau
<= mažiau arba lygu (ne daugiau)
== lygu
!= nelygu

>= daugiau arba lygu (ne mažiau)
> daugiau

Loginės aritmetikos operatoriai yra:

&& IR Loginė daugyba
|| ARBA Loginė sudėtis
! NE Neigimas

Loginė aritmetika yra skirta skaičiavimams dvejetainėje skaičiavimo sistemoje, t. y. sistemoje, kurioje yra tik du skaičiai 1 ir 0, arba būsenos Taip ir Ne. Pavyzdžiui, elektros jungiklis gali būti įjungtas arba išjungtas. Prisiminkime – 1 žymime Teisingą teiginį, o 0 žymime Neteisingą teiginį. Angliškai šie operatoriai vadinami: AND operatorius IR, OR – ARBA, NOT – NE.

Programos vykdymo valdymo operatoriai

Operatoriai *if ... else*: Tai sąlygos operatorius, kuris reiškia – jei sąlyga teisinga (if()) vykdyti pirmąją veiksmų grupę – priešingu atveju (else) vykdyti antrąją veiksmų grupę. Loginis operatorius „if“ gali būti naudojamas vienas, kai tuo tarpu **else** tik kartu su **if**.

Operatoriai *switch, case, break, default*: Šie operatoriai dar kartais vadinami perėjimo pagal lentelę operatoriais, iš analogijos su asemblerio kalba. Jų veikimas yra labai efektyvus, nes po switch() operatoriaus programos valdymas iš karto yra perduodamas į tą eilutę, kurioje yra atitinkama case X: reikšmė.

Ciklai

Dažnai programoje reikia valdyti besikartojantį procesą. Sprendimas yra ciklai. Ciklai leidžia sudaryti veiksmų seką, kuri gali kartotis ir kartotis, su sąlygų mechanizmu, kuris leidžia užbaigti ciklo vykdymą. C kalboje yra trijų rūšių ciklai: *while, do ... while* ir *for*.

- ***while ()* ciklas:** yra paprasčiausias iš visų ciklų. Ciklo sąlyga (pavyzdžiui, a > b) yra apskaičiuojama kiekvieną kartą, kai pradedamas vykdyti ciklas. Ciklo veiksmas yra vykdomas, kol sąlyga yra teisinga. Kai sąlyga pasidaro neteisinga, arba reiškinys lenktiniuose skliausteliuose už žodžio *while* tampa lygus nuliui (0), ciklo vykdymas nutraukiamas, ir vykdoma kita programos eilutė po ciklo veiksmą apgaubiančių figūrinių skliaustelių sekantis_veiksmas.
- ***do ... while()* ciklas:** Šį ciklą galima apibūdinti, kaip atlikti veiksmą, kol sąlyga teisinga. do { veiksmas; } while (sąlyga); Reikia atkreipti dėmesį, kad šiame cikle sąlyga yra tikrinama ciklo pabaigoje. Taigi ciklas bus visada vykdomas bent vieną kartą, net kai ciklo pradžioje sąlyga yra Neteisinga (0). Tai jo vienintelis skirtumas nuo *while()* ciklo.
- ***for (...;...;...)* ciklas:** Šis ciklo operatorius yra sudėtingiausias iš visų ciklo operatorių. Ciklas *for* visada turi kintamąjį, kuriuo manipuliuoja kiekvienoje ciklo iteracijoje. Jis yra vadinama ciklo kintamuoju. Paprastai, naudojant *for* operatorių, parametrai turi tokias reikšmes: *reiškinys1,* paprastai čia yra inicializuojamas ciklo kintamasis, t. y. nustatoma jo reikšmė ciklo pradžiai, pavyzdžiui (i = 0). *Sąlyga* – tai sąlyga, panaši į

while ciklo, ji yra apskaičiuojama kiekvienai ciklo iteracijai ir, kai sąlyga tampa Neteisinga (0), ciklas baigiamas. Tai dažniausiai yra paprasta sąlyga, pavyzdžiui (i < 20). *Reiškinys2* – tai kažkoks reiškinys, kuris keičia ciklo kintamojo reikšmę. Dažniausiai koks nors paprastas reiškinys, pavyzdžiui (i++, i *= 20, i /=0.5).

- ***break*:** Šį operatorių jau matėme, kai kalbėjome apie *switch* operatorių, jo paskirtis yra lygiai tokia pat. Operatorius *break* nutraukia ciklo vykdymą bet kurioje ciklo iteracijoje ir bet kurioje veiksmo vietoje.
- ***continue:*** Programuojant reikia ne tik baigti ciklą ankščiau laiko, bet ir tęsti nevykdant visų vėliau aprašytų veiksmų. Tam yra naudojamas operatorius *continue*.
- ***goto label:*** Besąlyginio perėjimo operatorius *goto* perduoda programos vykdymą į pažymėtą programos vietą. Šis operatorius C programuotojų yra nemėgstamas ir nenaudojamas. Teisingai programuojant šio operatoriaus nereikia.

Funkcijos

Funkcija yra blokas arba modulis, kuris programoje atlieka tam tikrą veiksmą. Ji yra priemonė izoliuoti vieną programos bloką ar modulį nuo kitų nepriklausomų programos dalių. Tai suteikia C kalbai dvi svarbias galimybes:

1. padaryti dalį programos nepriklausomą nuo kito programos kodo ir pavesti jai atlikti tam tikrą užduotį;
2. padaryti dalį programos, kuri programos kode gali būti panaudota kiek nori kartų be jokių pakeitimų.

Funkcijos, kaip ir kintamojo, vardas turi prasidėti lotyniškos abėcėlės raide arba pabraukimo (underscore (_)) simboliu, o kiti simboliai gali būti parinkti iš šių grupių:

a ... z (mažosios lotyniškos raidės nuo a iki z)
A ... Z (didžiosios lotyniškos raidės nuo A iki Z)
0 ... 9 (skaičiai nuo 0 iki 9)
_ (pabraukimas) (underscore).

Funkcijos iškvietimas: Funkcija yra nepriklausoma programos dalis, kuri programoje gali būti panaudota tiek kartų, kiek reikia be jos kodo pakeitimo. Funkcija yra iškviečiama su tam tikrais argumentais, kurie kiekvieną kartą gali turėti skirtingas reikšmes. Iškvietus funkciją programos valdymas yra perduodamas funkcijai, o funkcijai pasibaigus grąžinamas į kitą programos eilutę po funkcijos iškvietimo.

- ***return (…)* operatorius:** Priverčia funkciją užsibaigti ir grąžinti reikšmę funkciją iškvietusiai funkcijai ar programai. Šio operatoriaus neturi būti funkcijose, kurios negrąžina reikšmės (void func()).

Programavimo priemonės

C kalbos kompiliatoriai: C kalbos kompiliatorių yra labai daug ir labai įvairių. Bet visi jie turi tam tikras vienodas sudedamąsias dalis. Pakalbėsime apie pagrindines. Funkciniu požiūriu atskiros dalys yra laikomos atskiruose kataloguose (directory). Kataloge *bin* yra C kompiliatorius ir papildomų programų. Programų pavadinimai gali būti gan įvairūs:

- *cpp, cpp32, bcc, bcpp32, gcc, cc* ir t. t. – taip yra vadinami C kompiliatoriai;
- *link, ilink, mink, glink* ir t. t. – taip vadinamos komponavimo programos;
- *asm, tasm, masm, iasm, gasm* ir t. t. – taip vadinami transliatoriai iš asemblerio;
- *lib, mlib, ar, mplib* ir t. t. – taip vadinamos bibliotekų surinkimo programos;
- *make* – taip vadiname kompiliavimo automatizavimo priemonę.

Šiame kataloge paprastai būna daugiau programų, bet tai kiekvieno programavimo paketo ypatybės. Programuojant kur kas svarbesnės už programas yra antraštės ir bibliotekos.

Antraštės (headers) ir bibliotekos (libraries): Rašant programas, programos teksto pradžioje visada yra tokios ar panašios eilutės, prasidedančios preprocesoriaus direktyva *#include: #include #include* – tai antraščių įtraukimas į programos tekstą. C kalboje galima naudotis tik objektais (kintamaisiais, konstantomis, funkcijomis ir t. t.), kurie yra deklaruoti. Tačiau pačių parašytos funkcijos yra deklaruojamos arba rašomos pirmiau negu *main()* funkcija. C kalbos bibliotekų antraščių failai dažniausiai yra kompiliatoriaus *include* kataloge. Antraščių failų vardai dažniausiai atitinka bibliotekų vardus, skiriasi tik plėtiniais: *.h ir *.hcc – antraštėms ir *.l ir *.lib – bibliotekoms. Antraščių failai yra tekstiniai, juos galima pažiūrėti su bet kokiu teksto redaktoriumi. Tačiau redaguoti galima tik su simboliniais redaktoriais, nes tokie redaktoriai kaip „MS Word“ gali sugadinti. Antraštės faile yra surašytos makrokomandos, konstantiniai kintamieji, funkcijų deklaracijos, struktūrų aprašymai ir t. t. Bibliotekų failai yra dvejetainiai ir jų redaguoti negalima, o ir žiūrėti nėra reikalo. Jie yra sudaryti iš funkcijų objektinių modulių, naudojant specialias programas bibliotekininkus. Su šiomis programomis galima sudaryti savo bibliotekas, išimti, įdėti ar pakeisti objektinius modulius.

Programavimo procesas: Programos sukūrimas yra gan ilgas ir sudėtingas procesas, kuris schematiškai pavaizduotas 5 paveiksle. Kaip ir bet kuris darbas, jis prasideda nuo užduoties, kurioje turi būti išaiškinta, ką reikia padaryti. Išsiaiškinus, ką reikia padaryti, smulkiai aprašomi visi įeinantys duomenys ar signalai, valdymas ir išeinantys duomenys ar signalai. Ši informacija sudaro techninę užduotį, kuri yra pagrindas sudarant programos veikimo algoritmą. Sudėtingų programų algoritmų sudarymui naudojamos įvairios programinės priemonės, kurios naudoja UML (Universal Modeling Language) kalbą. Sudarant algoritmą, būtina išanalizuoti ne tik darbinius duomenis ir signalus, bet ir avarines būsenas, kurioms susidarius programos veikimas neturi sugriūti. Avarinė būsena turi būti atpažinta, programa turi pranešti apie avariją ir jos priežastį ir nesugriuvusi tęsti darbą. Sudarius algoritmą yra sprendžiama, kokia programavimo kalba rašyti programą. Pasirinkus programavimo kalbą, pagal algoritmą yra parašomas programos tekstas. Programų tekstų rašymui galima naudoti ir paprastą teksto redaktorių, tačiau yra daugybė redaktorių skirtų programų.

Programos kompiliavimas: Parašytas programos tekstas yra kompiliuojamas. Kai reikia sukompiliuoti programą, kurios programos ir antraščių tekstai yra keliuose failuose, yra naudojama *make* programa. Kompiliavimo procesas yra aprašomas faile, kurio vardas dažniausiai yra *Makefile*, jame yra surašyta programos dalių kompiliavimo ir surinkimo

procedūrų eilės tvarka. Jei rašant programą yra padaryta klaidų (dažniausiai taip ir būna), kompiliatorius apie tai praneša. Kompiliavimas kartojamas tiek kartų, kiek reikia, kol programa sukompiliuojama sėkmingai. Kad programa sukompiliuojama sėkmingai, dar nereiškia, kad ji teisingai veikia. Kai programa sukompiliuojama, pradedamas programos derinimas. Programos derinimas yra programos veikimo tikrinimas visiems galimiems įėjimo duomenims ir signalams, o taip pat visiems duomenims ir signalams, kurie gali atsirasti avariniame režime. Jei reikia, yra daromi pakeitimai programos tekste ar algoritme. Kompiliavimas iš tikrųjų nėra viena operacija. Jau minėjome, kad sudėtingoms programoms kompiliuoti yra naudojama *make* programa, kuri skaito nurodymus iš *Makefile* ir iškviečia reikalingas programas veiksmams atlikti.

Pagrindiniai veiksmai ir programos, kurios vykdo šiuos veiksmus, yra:

- Preprocesorius (preprocessor) skaito programos tekstą, ieškodamas jam skirtų komandų, kurios prasideda simboliu (#), parašytu būtinai prie kairiojo krašto (be tarpų ir tabuliacijos). Dažniausiai jis į programos tekstą įtraukia visus išvardintus antraščių failus. Tačiau tarp preprocesoriaus komandų yra ir kompiliatoriaus valdymo komandos arba taip vadinamos sąlyginės kompiliacijos komandos.
- Kompiliatorius (compiler) iš programos teksto padaro objektinį failą, kurio plėtinys dažniausiai yra *.obj*. Tai dar nėra programa, kuri gali veikti kompiuteryje, bet tai jau nebe tekstinis, o dvejetainis failas. Kad objektinis failas būtų paverstas programa, reikalingas komponuotojas.
- Komponuotojas (linker) iš objektinių failų surenka programos failą, kuris gali būti vykdomas kompiuteryje. Tikriausiai jau matėte failus su plėtiniu *.lib*, juos padaro bibliotekininkas, programa, kuri sujungia kelis objektinius failas į vieną su *.obj* plėtiniu. Tai daroma tam, kad nereikėtų nurodyti dešimčių ar šimtų failų vardų.

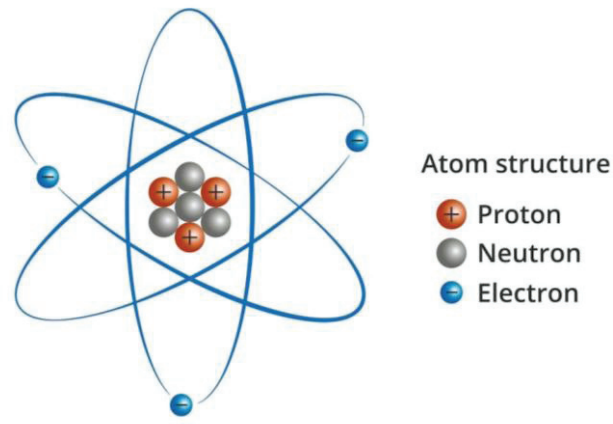
V. Elektronikos pagrindai

Šis, elektronikos pagrindus aprašantis, skyrius skirtas vyresniems moksleiviams, kurie nori kurti robotus iš atskirų elektronikos komponentų. Šiame skyriuje pateikiami dažniausiai naudojamų komponentų aprašymai, specifikacijos ir pajungimo būdai. Taip pat pateikiami daugiausia elektrinių grandinių kūrimo principai ir metodai.

Įvadas

Elektronika – tai mokslo ir technikos šaka, tirianti ir praktiškai naudojanti reiškinius, kurie vyksta krūvininkams judant įvairioje aplinkoje (vakuume, dujose, skysčiuose bei kietuose kūnuose). Elektroniniu elementu vadinamas atskiras įtaisas, kurio elektrinį laidumą lemia elektronai, judantys vakuume, dujose, skysčiuose bei kietuose kūnuose.

Kūnų elektrinis laidumas priklauso nuo laisvųjų krūvininkų skaičiaus. Visos medžiagos yra sudarytos iš atomų, kurių kiekvieną sudaro teigiamai įelektrintas branduolys ir neigiamai įelektrinti elektronai. Elektronas – tai elementari dalelė, kurios masė yra 9,1·10–31 kg. Elektronai atome juda tam tikromis trajektorijomis, kurios vadinamos orbitomis.



SEQ Figure * ARABIC 30 pav. Atomo

Išorinėmis orbitomis skriejantys elektronai yra vadinami valentiniais elektronais. Jų ryšys su branduoliu yra silpniausias. Dėl įvairių poveikių tik valentiniai elektronai gali atsiskirti nuo atomo ir tapti laisvaisiais elektronais. Jie medžiagoje juda netvarkingai, įvairiomis kryptimis. Kryptingą elektronų judėjimą sukelia sukurtas elektrinis laukas. Puslaidininkiais vadinami cheminiai elementai arba junginiai, kurių varža yra tarp laidininkų ir izoliatorių. Tai germanis, silicis, boras, fosforas, arsenas, stibis, siera, selenas ir kt.

Elektromagnetinių krūvių dėsnis

Dauguma objektų paprastai turi neutralų arba nulinį krūvį, tai yra juose yra tiek pat elektronų, kiek ir protonų. Jei būtų galima padaryti elektronų perteklių, kūnas būtų neigiamai įkrautas. Kita vertus, jei trūktų elektronų, tada turėtume protonų perteklių, kūnas būtų teigiamai įkrautas.

Jei paimsime bet kurį teigiamai įkrautą kūną ir priartinsime jį prie neigiamo krūvio kūno, abu kūnai trauks vienas kitą. Kita vertus, jei abu objektai turi panašius krūvius, jie

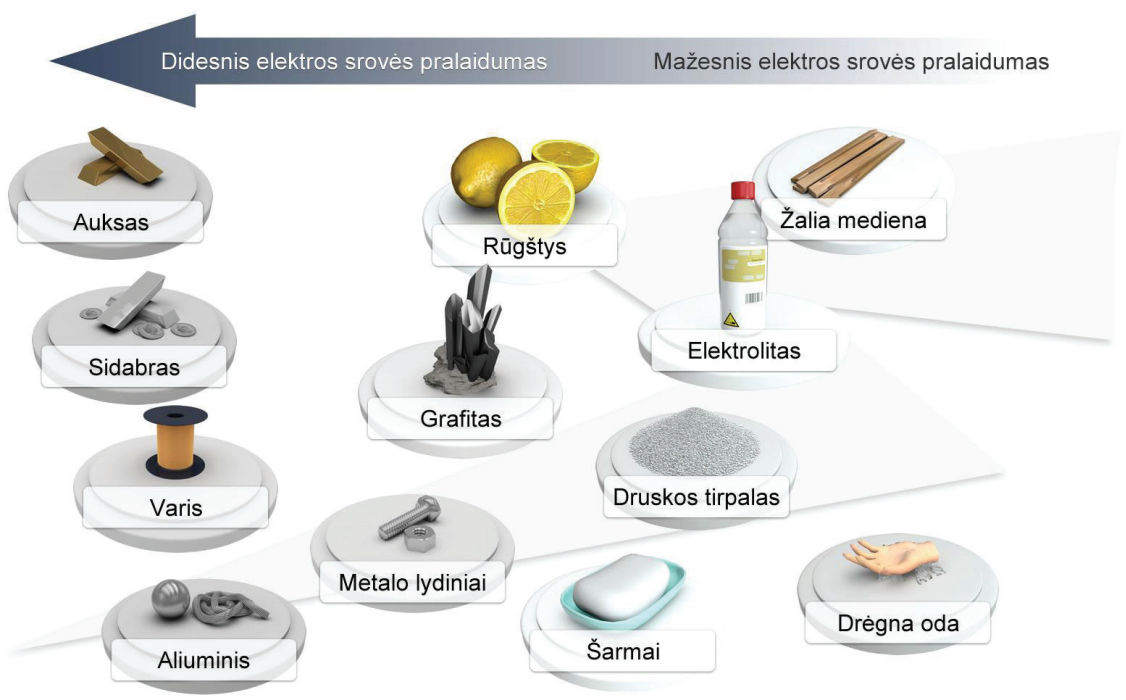
atstums vienas kitą. Šios dvi reakcijos sudaro pirmojo elektros dėsnio, žinomo kaip elektromagnetinių krūvių tvermės dėsnis, pagrindą. Šis dėsnis teisingas visiems žinomiems fizikiniams procesams. Dideliuose medžiagos kiekiuose krūviai susikompensuoja ir medžiaga yra elektriškai neutrali, tačiau gali būti poliarizuota, jei krūviai kūno srityse pasiskirstę netolygiai.

Potencialų skirtumai

Jei sujungsime varinį laidą tarp dviejų priešingai įkrautų kūnų, atsiras elektronų srautas. Elektronai tekės iš neigiamai įkrauto kūno į teigiamai įkrautą kūną. Taip yra todėl, kad teigiamai įkrautas kūnas, turintis elektronų trūkumą, pritraukia elektronų perteklių iš neigiamai įkrauto kūno. Šis veiksmas tęsiasi tol, kol išnyksta elektronų trūkumas ir perteklius, o du kūnai tampa neutralūs. Šį potencialų skirtumą vadiname ELEKTROS SLĖGIU, arba ĮTAMPA.

Laidininkai, dialektrikai ir elektrinė varža

Elektros laidininkas – medžiaga, turinti laisvųjų elektrintųjų dalelių, arba krūvininkų. Atsiradus elektriniam laukui, elektronai (krūvininkai) laidininke juda dvejopai: be chaotiško labai intensyvaus šiluminio judėjimo jie lėtai slenka viena kryptimi – tai vadinamas elektronų dreifas. Geriausi laidininkai yra metalai. Prie laidininkų taip pat priskiriami elektrolitų tirpalai, drėgnas oras, plazma, žmogaus kūnas, žemė ir kt. Metaluose teigiamuosius jonus supa laisvieji elektronai. Kai išorinis elektrinis laukas lygus nuliui, laisvieji elektronai juda metalu netvarkingai, t. y. įvairiomis kryptimis. Elektronų šiluminio judėjimo intensyvumas priklauso nuo temperatūros. Laidininko viduje elektrinio lauko nėra, nes elektronų ir jonų sukuriami elektriniai laukai kompensuoja vieni kitus.



SEQ Figure * ARABIC 31 pav. Srovės pralaidumas

Dielektrikas – medžiaga, nelaidi elektros srovei, nes neturi laisvų elektros krūvių. Tačiau dielektriką veikiant elektriniu lauku, surištieji krūviai, esantys atomuose ir molekulėse, labai nežymiai pasislenka, o jis pats tampa elektrinio lauko šaltiniu. Dielektrikai, kaip ir visos medžiagos, sudaryti iš atomų. Pastarieji sudaryti iš teigiamų ir neigiamų elektringųjų dalelių: elektronų ir protonų. Sudarius dielektrikui išorinį elektrinį lauką, pastarasis veikia šias elektringąsias daleles. Atomai (arba ištisos molekulės) tampa dipoliais. Dielektrikas poliarizuojasi. Dielektrikai būna dviejų rūšių:

- Nepoliniai – dielektrikų atomai ir / ar molekulės nėra dipoliai tol, kol nėra juos veikiančio išorinio elektrinio lauko. Paveikus tokius dielektrikus elektriniu lauku, atomai tampa dipoliais, kurių dipoliniai momentai orientuojasi lauko kryptimi.
- Poliniai – dielektrikų atomai ir / ar molekulės yra dipoliniai momentai nesant išorinio elektrinio lauko. Todėl išorinis elektrinis laukas poliniame dielektrike tik orientuoja jo elektrinius dipolinius momentus lygiagrečiai lauko kryptimi. Kai kurios medžiagos gali poliarizuotis ir be išorinio lauko. Tokie dielektrikai skirstomi į:
- Pjezoelektrikus. Tai kristalai, kuriuose pasireiškia pjezoelektrinis reiškinys – jie poliarizuojasi mechaniškai deformuojami (spaudžiant, tempiant).
- Feroelektrikus (segnetoelektrikus). Tai poliniai dielektrikai, kurie savaime poliarizuojasi tam tikrame temperatūros intervale. Kiekvienam feroelektrikui būdinga individuali temperatūra, dar vadinama Kiuri temperatūra. Esant šiai temperatūrai feroelektrinės savybės išnyksta, ir jis tampa tiesiog poliniu dielektriku.
- Elektretus − pastovių magnetų analogai, kurių aplinkoje yra elektrinis laukas.

Elektrinė varža – kūno savybė priešintis elektros srovei. Ji egzistuoja todėl, kad laisvieji elektronai sąveikauja su laidininko atomais. Kiekvieną kartą išvaduojant elektroną iš atomo įtakos, panaudojama šiek tiek energijos, kuri virsta šiluminiais nuostoliais, o kartais ir šviesa. Ji dar vadinama aktyviąja varža. Jos didumas priklauso nuo medžiagos, iš kurios sudarytas kūnas, savitosios elektrinės varžos ir kūno formos. Varžos matavimo vienetas yra omas (Ω). Judėdami kūnu, elektronai atsitrenkia į jo atomus ir perduoda jiems energiją, kartu įkaitindami kūną ir naudodami šaltinio energiją. Elektrinė varža yra objekto geometrinių matmenų ir savitosios varžos funkcija

SEQ Figure * ARABIC 32
pav. Rezistorius

$$P = (J * p) / S$$

l – laidininko ilgis metrais;
S – plotas kvadratiniais metrais;
ρ – medžiagos savitoji elektrinė varža, matuojama omais (Ω).

Elektrinė varža taip pat gali būti išreikšta iš Omo dėsnio

$$R = \frac{U}{I}$$

U – potencialų skirtumas išilgai objekto, skaičiuojamas voltais;
I – elektros srovė, tekanti per objektą, skaičiuojama amperais;

Įtampa ir srovė

Elektrinė įtampa apibūdina darbą, kurį atlieka (arba gali atlikti) elektros krūvis tekėdamas grandine. Žinodami grandine pratekėjusį elektros krūvį q ir atliktą darbą A, galime apskaičiuoti įtampą:

$$U = \frac{A}{q} = \frac{A_k}{q} + \frac{A_p}{q}$$

A_k – kuloninių jėgų atliktas darbas.
A_p – pašalinių jėgų atliktas darbas.
U – elektrinė įtampa. Kitas dažnai pasitaikantis žymėjimas raide V (dažniausiai ne Europos šalys).

Elektrinę įtampą galima apibūdinti kaip potencialo verčių skirtumą pradiniame ir galiniame taške.

Elektros srovė – kryptingas elektros krūvių judėjimas. Dar kitaip įvardijamas kaip kryptingas laisvųjų elektringųjų dalelių judėjimas. Srovę galima paskaičiuoti pagal formulę:

$$I(t) = \frac{sq(t)}{dt}$$

dq – mažas krūvio pokytis,
dt – laiko pokytis (laikant abu pokyčius pakankamai mažais, kad krūvį q būtų galima laikyti pastoviu).

Elektros srovė grandine teka iš teigiamojo elemento poliaus neigiamojo link. Ją nešančios dalelės juda šia kryptimi, jei jų elektros krūvis teigiamas. Metaluose srovę perduoda neigiamą krūvį turintys elektronai, kurie juda priešinga kryptimi.

Elektros srovės tankis – skaliarinis dydis, lygus elektros krūviui, kuris praeina pro laidininko skerspjūvio plotą per laiko vienetą. Sąvoką įvedė Georgas Omas 1827 m.

SI sistemoje elektros srovė matuojama amperais.

$$1\text{A} = 1\text{Cs}^{-1}$$

Tačiau srovė apibrėžiama ne kaip krūvio kitimas, o per magnetizmą. Jei dviem be galo ilgais (pakankamai ilgais, jog tolesnis ilginimas nedarytų įtakos rezultatui daugiau nei matavimo paklaidos) lygiagrečiais laidais, kurie nutolę 1 m atstumu vienas nuo kito, teka 1A srovė, tai tie du laidai sąveikauja 2 10-7 N jėga.

Elektros šaltiniai

Elektros energijos šaltinis – tai įtaisas elektros energijai gauti iš kitų energijos rūšių. Pagal verčiamos energijos rūšį elektros energijos šaltiniai būna cheminiai ir fizikiniai.

Cheminiai elektros energijos šaltiniai cheminę oksidacijos-redukcijos reakcijų energiją verčia elektros energija. Būna pirminiai (galvaninis elementas ir jų baterijos), antriniai (elektros akumuliatorius) ir kuro elementas. Pirminiai šaltiniai yra vienkartiniai, antriniai – daugkartiniai (juos galima įkrauti elektros srove).

Fizikiniai elektros energijos šaltiniai mechaninę, šiluminę, šviesos, branduolinės reakcijos ar kitokią energiją verčia elektros energija. Prie tokių elektros energijos šaltinių priskiriami, pvz., nuolatinės ir kintamosios srovės generatoriai (elektros energija verčiama mechaninė energija, veikimas grindžiamas elektromagnetine indukcija), magnetohidrodinaminiai generatoriai (verčiama šiluminė energija, veikimas grindžiamas elektromagnetine indukcija plazmoje, elektrolite, skystajame metale), fotoelektriniai generatoriai (verčiama šviesos energija, veikimas grindžiamas fotoelektriniais reiškiniais), izotopiniai generatoriai (verčiama nuklidų radioaktyviojo skilimo šiluminė energija) ir kitas, baterijos paprastai gaminamos 6,3 arba 12,6 volto.

Nuolatinė elektros srovė

Nuolatinė elektros srovė – elektros srovė, kurios stipris ir kryptis laikui bėgant nekinta. Nuolatinė elektros srovė atsirasti ir tekėti gali tik tada, kai medžiagoje yra laisvųjų elektringųjų dalelių. Jeigu teigiamieji ir neigiamieji krūviai susiję vienas su kitu atomuose arba molekulėse, tai jų judėjimas nesukels elektros srovės. Bet laisvųjų krūvių egzistavimo dar nepakanka srovei atsirasti. Elektringųjų dalelių tvarkingam judėjimui sukelti ir palaikyti reikia jėgos, veikiančios jas tam tikra kryptimi. Kai ši jėga nustoja veikti, kryptingas elektringųjų dalelių judėjimas nutrūksta dėl varžos, kurią sukelia metalų kristalinės gardelės jonai arba elektrolitų neutraliosios molekulės. Paprastai elektrinis laukas laidininko viduje yra priežastis, sukelianti ir palaikanti elektringųjų dalelių kryptingą judėjimą. Tik statikos atveju, kai krūviai nejuda, elektrinis laukas laidininko viduje lygus nuliui.

Elektrinėje grandinėje vyksta įvairūs energijos virsmai. Kai elektringosios dalelės laidininke juda kryptingai, elektrinis laukas atlieka darbą. Šį darbą įprasta vadinti srovės darbu. Srovės darbas A grandinės dalyje lygus srovės stiprio I, įtampos U ir laiko t, per kurį atliekamas darbas, sandaugai:

$$A = I * U * \Delta t$$

Pagal energijos tvermės dėsnį šis darbas turi būti lygus nagrinėjamos grandinės dalies energijos pokyčiui. Todėl energija, išsiskirianti nagrinėjamoje grandinės dalyje per laiką Δt, lygi srovės darbui. Bet kuris elektrinis prietaisas, lempa, elektros variklis ir t. t. yra apskaičiuoti tam tikram energijos kiekiui suvartoti per laiko vienetą. Todėl kartu su srovės darbu labai svarbią reikšmę turi srovės galios sąvoka. Srovės galia lygi srovės darbo per laiką Δt ir to laiko santykiui:

$$P = A/\Delta t$$

Taikant Omo dėsnį grandinės daliai, šią galios išraišką galima užrašyti keliomis ekvivalenčiomis formulėmis:

$$P = I * U = I^2 * R = U^2/R$$

Omo dėsnis – fizikinis dėsnis, nusakantis įtampos, srovės stiprio ir laidininko varžos priklausomybę elektros grandinėje. Pavadintas šio dėsnio atradėjo – Georgo Omo – vardu.

Omo dėsnis grandinės daliai:

$$U = I * R$$

U – įtampa arba potencialų skirtumas, [V] – Voltas;
I – srovės stipris, [A] – Amperas;
R – laidininko varža, [Ω] – Omas.

Omo dėsnis visai grandinei:

$$I = \varepsilon / (R + r)$$

ε – grandinės EV (elektrovara), [V];
I – srovė, [A];
R – visų grandinės elementų varža, [Ω];
r – šaltinio varža, [Ω].

Įtampos, srovės ir varžos matavimas

Dažniausiai naudojamas ir paprasčiausias matavimo prietaisas yra multimetras. Multimetras yra matavimo prietaisas, galintis išmatuoti kelias elektrines savybes. Įprastas multimetras gali matuoti įtampą, varžą ir srovę, tokiu atveju jis taip pat žinomas kaip voltų-omų milimetras (VOM), nes įrenginyje yra voltmetro, ampermetro ir ommetro funkcijų. Kai kuriose funkcijose matuojamos papildomos savybės, tokios kaip temperatūra ir talpa.

Analoginiai multimetrai naudoja mikroampermetrą su judančia rodykle rodmenims rodyti. Skaitmeniniai multimetrai (DMM, DVOM) turi skaitmeninius ekranus, todėl analoginiai multimetrai beveik paseno, nes yra pigesni, tikslesni ir fiziškai tvirtesni nei analoginiai multimetrai.

Multimetrai skiriasi dydžiu, funkcijomis ir kaina. Tai gali būti nešiojami rankiniai prietaisai arba labai tikslūs stendiniai instrumentai.

- **Įtampos matavimas** – tai elektros potencinės energijos matavimas. Įtampa dar vadinama potencialų skirtumu, todėl matuojama dviem elektrodais.
- **Srovės matavimas.** Srovės matavimas – tai krūvių srauto greičio matavimas. Elektros inžinerijoje – elektronų judėjimas. Elektros srovė matuojama nuosekliai, pertraukiant grandinę.
- **Varžos matavimas.** Varžos matavimas – tai pasipriešinimo elektros srovei matavimas. Varžą pamatuoti galima tik kiekvieno elemento atskirai.

Elektrinė grandinė

Elektrine grandine laikoma tokių tarpusavyje sujungtų elementų visuma, kurioje vykstantys procesai apibūdinami srovės ir įtampos priklausomybėmis.

Elektrinės grandinės elementai skirstomi į aktyvinius ir pasyvinius. Pirmieji yra elektros energijos šaltiniai, o antrieji vartoja arba kaupia energiją.

Grandinės elementai:

- **Varžos elementas.** Varžos elementu grandinių teorijoje vadinamas idealizuotas elementas, kurio įtampos ir srovės ryšys išreiškiamas Omo dėsniu;
- **Induktyvumo elementas.** Induktyvumo elementu grandinių teorijoje vadinamas idealizuotas elementas, apibūdinamas tokia srovės ir įtampos priklausomybe:

$$u = L * (\frac{di}{dt})$$

- **Talpumo elementas.** Talpumo elementu grandinių teorijoje vadinamas idealizuotas elementas, apibūdinamas tokia srovės ir įtampos priklausomybe:

$$i = C * (\frac{du}{dt})$$

- **Įtampos šaltinis.** Idealiu įtampos šaltinių grandinių teorijoje vadinamas toks aktyvinis grandinės elementas, kurio momentinės įtampos *u* kitimo dėsnis visiškai nepriklauso nuo tekančios srovės;
- **Srovės šaltinis.** Idealiu srovės šaltinių grandinių teorijoje vadinamas toks aktyvinis grandinės elementas, kurio momentinės srovės *i* kitimo dėsnis visiškai nepriklauso nuo įtampos.

Grandinių klasifikacija:

- Elektrinė grandinė, nagrinėjama dviejų polių atžvilgiu, vadinama dvipoliu;
- Elektrinė grandinė, nagrinėjama keturių polių atžvilgiu, vadinama keturpoliu;
- Grandinės, kuriose varžos, talpos ir induktyvumo savybėmis pasižymi tik atskiri varžos, talpumo ir induktyvumo elementai, vadinamos sutelktųjų parametrų grandinėmis;
- Grandinės, kuriose kiekviena dalis turi talpumą, induktyvumą bei varžą. Jos vadinamos paskirstytųjų parametrų grandinėmis;
- Grandinės, sudarytos iš tiesinių elementų, vadinamos tiesinėmis;
- Grandinės, kuriose yra bent vienas netiesinis elementas, vadinamos netiesinėmis;
- Grandinės, kuriose yra bent vienas parametrinis elementas, vadinamos parametrinėmis;
- Vien iš reaktyvinių elementų sudarytos grandinės vadinamos reaktyvinėmis;

- Grandinės, kuriose yra bent vienas varžos elementas, vadinamos grandinėmis su nuostoliais.

Elementai, kurių parametrai ar savybės nepriklauso nei nuo tekančios srovės, nei nuo veikiančios įtampos, vadinami tiesiniais. Ir priešingai: elementai, kurių parametrai priklauso nuo tekančios srovės ar veikiančios įtampos, vadinami netiesiniais. Yra tiesinių elementų, kurių parametrus galima keisti pagal pageidaujamą dėsnį, nepriklausomą nuo srovės ar įtampos. Jie vadinami parametriniais elementais.

Pagrindiniai grandinių dėsniai:

Omo dėsnis išreiškia srovės ir įtampos priklausomybę varžos elemente arba dvipolyje, sudarytame iš varžos elementų:

$$U = IR$$

Pirmasis Kirchhofo dėsnis taikomas grandinės išsišakojimo mazgams. Kadangi krūviai mazguose nesikaupia, į mazgą sutekančių srovių algebrinė suma lygi nuliui: sumuojant įtekančios ir ištekančios srovės turi būti su skirtingais ženklais.

$$\sum_n I_n = 0$$

Antrasis Kirchhofo dėsnis taikomas bet kuriam šakotinės grandinės uždarajam kontūrui. Sąlyginai nurodžius atskirose grandinės dalyse srovės kryptį, pasirenkama teigiama kontūro apėjimo kryptis, t. y. srovės ir elektrovaros ta kryptimi teigiamos. Apeinant uždarą kontūrą ratu, potencialų skirtumas, algebrinė srovės stiprių ir varžų sandaugų atskirose uždarojo kontūro dalyse suma yra lygi tame kontūre esančių elektros šaltinių elektrovarų algebrinei sumai:

$$\sum_n I_n R_N = \sum_N \varepsilon_n$$

Grandinės poveikiai ir charakteristikos. Elektrinėje grandinėje veikiančios šaltinių įtampos ar srovės vadinamos poveikiais, o poveikių sukeltos srovės grandinės šakose ar įtampos tarp mazgų vadinamos reakcijomis. Elektrinių grandinių poveikiai gali būti skirstomi šitaip:

- Nuolatinis poveikis. Grandinėje veikia nuolatinės srovės ar įtampos šaltiniai. Nagrinėjamoji reakcija – nuolatinės šakų srovės ar įtampos tarp mazgų;
- Harmoninis poveikis. Grandinėje veikia harmoninės (sinusinės ar kosinusinės) srovės ar įtampos šaltiniai. Nagrinėjamoji reakcija yra harmoninės šakų srovės ar įtampos tarp mazgų;
- Periodinis neharmoninis poveikis. Grandinėje veikia periodinės neharmoninės srovės ar įtampos šaltiniai. Nagrinėjamoji reakcija yra periodinės šakų srovės ar įtampos tarp mazgų;
- Neperiodinis poveikis. Grandinėje veikia neperiodinės srovės ar įtampos šaltiniai. Neperiodiniai poveikiai apima labai plačią poveikių klasę;
- Atsitiktinis poveikis. Grandinėje veikia tokie srovės ar įtampos šaltiniai, kurių srovė ar įtampa neišreiškiama determinuotomis funkcijomis, o yra atsitiktinio pobūdžio.

Grandinių komponentai

Aktyvūs ir pasyvūs komponentai:

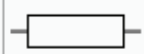
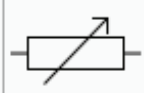
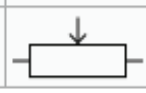
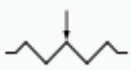
PAGRINDAS	AKTYVŪS KOMPONENTAI	PASYVIEJI KOMPONENTAI
Šaltinio pobūdis	Aktyvūs komponentai suteikia grandinei galios ar energijos.	Pasyvieji elementai naudoja energiją ar energiją grandinėje.
Pavyzdžiai	Diodai, tranzistoriai, integriniai grandynai ir kt.	Rezistorius, kondensatorius, induktorius ir kt.
Komponento funkcija	Prietaisai, kurie gamina energiją įtampos arba srovės pavidalu.	Prietaisai, kuriuose energija saugoma įtampos arba srovės pavidalu.
Galia	Jie gali suteikti galios padidėjimą.	Jie negali suteikti galios.
Srovės srautas	Aktyvūs komponentai gali valdyti srovės srautą.	Pasyvieji komponentai negali kontroliuoti srovės srauto.
Išorinio šaltinio reikalavimas	Jie reikalauja išorinio šaltinio operacijoms.	Jie nereikalauja jokių išorinių šaltinių operacijoms.
Energijos pobūdis	Aktyvūs komponentai yra energijos donoras.	Pasyvieji komponentai yra energijos priėmėjas.

Optoelektroniniai ir elektromechaniniai komponentai. Optoelektronika – tai fotonikos ir elektronikos mokslų sintezė. Optoelektroniniai komponentai informacijai apdoroti naudoja ne elektronus, o fotonus. Optinio diapazono bangų ilgis nuo 1 mm iki 1 nm. Elementai – fotodiodai, fotorezistoriai, fototranzistoriai, fotodetektoriai ir t. t. Elektromechanika – tai mechanikos ir elektronikos mokslų sintezė. Elektromechaniniai įrenginiai, kuriems veikiant naudojama arba kuriama išorinė mechaninė jėga.

Pagrindinių elementų žymėjimas ir jungimas

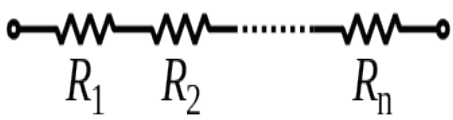
Rezistorius, arba varža – tam tikros varžos prietaisas, naudojamas reikiamai įtampai sudaryti. Jis priešinasi per jį tekančiai elektros srovei, sukurdamas įtampos kritimą tarp kontaktų. Šis sąryšis aprašomas Omo dėsniu $U = I * R$

Rezistorių varžos dydžiai ant korpuso žymimi pagal kelias spalvinių žymenų sistemas. Rezistorių žymėjimo elektrinėse grandinėse standartai Europoje ir JAV skiriasi:

	Europa	JAV
Rezistorius		
Kintamos varžos rezistorius		
Potenciometras		

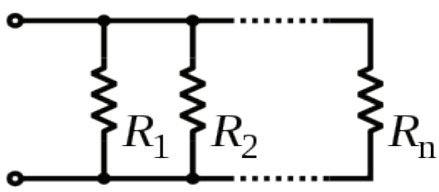
Nuoseklūs ir lygiagretūs jungimai, jungiant rezistorius nuosekliai, jų varžos susideda:

$$R = R_1 + R_2 + \cdots + R_n$$



Jungiant rezistorius lygiagrečiai, bendra varža skaičiuojama pagal tokią formulę:

$$\frac{1}{R_{eq}} = \frac{1}{R_1} + \frac{1}{R_2} + \cdots + \frac{1}{R_n}$$

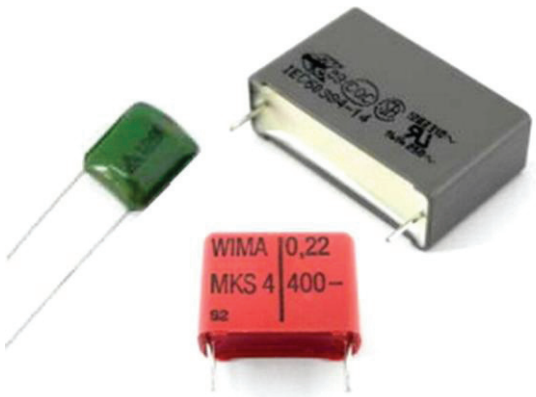


Rezistoriaus naudojama galia lygi išilgai jo prijungtos įtampos kvadrato ir juo tekančios srovės santykiui:

$$P = I^2 * R = I * V = \frac{V^2}{R}$$

Kondensatorius – dviejų laidininkų, vadinamų elektrodais, sistema, kuri turi savybę kaupti energiją tarp tų elektrodų sukurtame elektriniame lauke. Prijungus įtampą ant elektrodų (arba „plokštelių“) kaupiasi krūviai, kurių absoliutinės vertės lygios, tik priešingi ženklai. Pagal formą kondensatoriai skirstomi į:

- Plokščiuosius
- Sferinius
- Cilindrinius



SEQ Figure * ARABIC 37 pav. Kondensatoriai

Talpa. Kondensatoriaus elektrinė talpa apibrėžiama panašiai, kaip ir pavienio laidininko: ji lygi vieno elektrodo krūvio ir potencialų skirtumo tarp elektrodų santykio moduliui. Modulis imamas dėl to, jog elektros krūvis gali būti tiek teigiamas, tiek neigiamas, o kondensatoriaus elektrinė talpa yra tik teigiamas dydis:

$$C = \frac{q}{U}$$

C – elektrinė talpa;
q – vieno elektrodo krūvis;
U – potencialų skirtumas tarp elektrodų.

Talpos matavimo vienetas SI sistemoje – faradas (F). Kadangi faradas yra gana didelis dydis, dažniausiai kondensatoriaus talpa nusakoma mikrofaradais (μF), nanofaradais (nF), pikofaradais (pF).

Pagal naudojamą dielektriką dažniausiai skirstomi į dvi grupes:

- Elektrolitinius
- Keramikinius

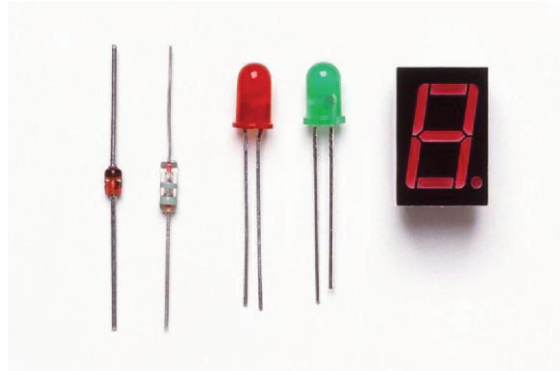
Nuoseklus kondensatorių jungimas: šakos, kurioje du kondensatoriai sujungti nuosekliai, talpai galioja ši formulė:

$$C = C_1 + C_2$$

Lygiagretus kondensatorių jungimas: šakos, kurioje du kondensatoriai sujungti lygiagrečiai, talpai galioja:

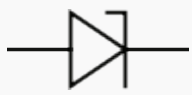
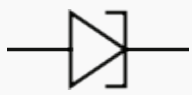
$$\frac{1}{c} = \frac{1}{c_1} + \frac{1}{c_2}$$

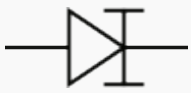
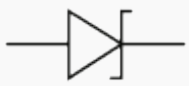
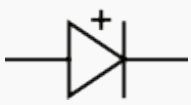
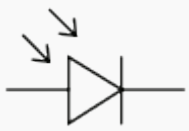
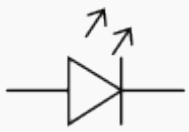
Puslaidininkis diodas – puslaidininkis prietaisas, turintis vieną elektroninę skylinę sandūrą, kuriame panaudotos pn sandūros savybės. Diodas paprastai turi du kontaktus, nebent viename korpuse jų būtų pagaminta keletas. Dažniausiai pritaikoma jų savybė praleisti elektros srovę tik viena kryptimi (atbulinis laidumas labai mažas), tačiau esama



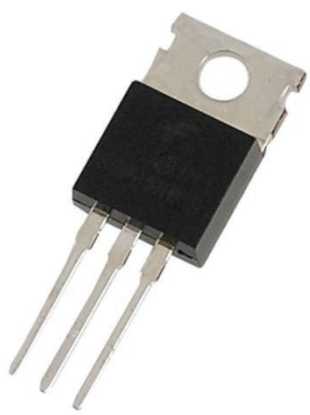
ir įvairesnių naudojimo būdų.

Diodų klasifikacija žemiau pateiktoje lentelėje:

EIL. NR.	DIODO TIPAS	ŽYMUO	SANDŪROS SAVYBĖS PANAUDOJIMAS
1	Lygintuviniai impulsiniai universalūs		Paprastai tariama, jog jie tiesiog praleidžia srovę tik viena (ženkle – trikampio viršūnės rodoma) kryptimi. Jie naudojami srovės lygintuvuose, loginiuose elementuose ir kitose srityse. Šiuose dioduose taikomos pn sandūros vienpusio laidumo ypatybės.
2	Stabilitronas		Kintant diodu tekančiai srovei, palaiko pastovią jam tenkančią įtampą. Šie diodai jungiami „atvirkščia“ (srovė teka iš katodo į anodą) kryptimi.
3	Varikapas		Šis tipas dirba kaip įtampa valdomas kondensatorius ir dažnai naudojamas automatiniam programų derinimui televizoriuose ir radijo imtuvuose. Uždaryta pn sandūra pasižymi nemaža talpa, kuri priklauso nuo diodui tenkančios įtampos. Varikapo talpa gali būti maždaug nuo 3 iki 300 pF ir kintant valdymo įtampai keičiasi apie dešimt kartų. Valdymo įtampa gali siekti iki 30 V.
4	Tunelinis diodas		Šio diodo voltamperinėje charakteristikoje yra neigiamos diferencialinės varžos sritis. Tos srities ribose, mažinant diodui tenkančią įtampą, juo tekanti srovė stiprėja. Tuneliniai diodai tinka signalui sustiprinti. Įdomu, jog panašią charakteristiką turi gyvųjų ląstelių

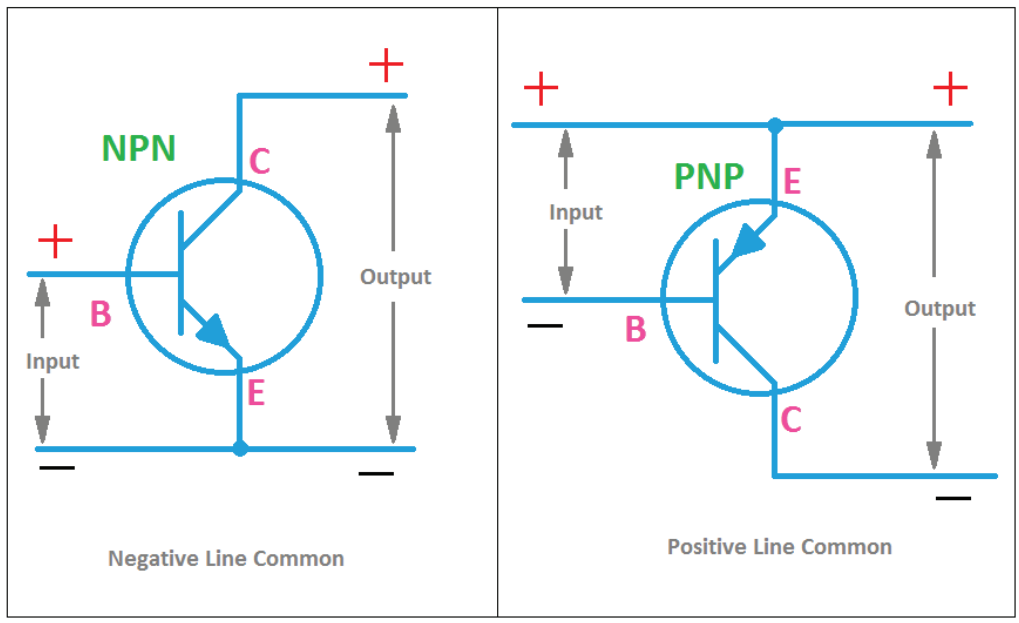
			joniniai kanalai.
5	Inversinis diodas		Padidintos koncentracijos puslaidininkių sandūros savybių panaudojimas.
6	Šotkio diodas		Šie diodai pasižymi labai mažu įtampos kritimu, srovei per juos tekant tiesiogine kryptimi. Tai svarbu loginiuose elementuose, impulsiniuose maitinimo keitikliuose.
7	Magnetinis diodas		Magnetinių reiškinių panaudojimas puslaidininkio tūryje.
8	Fotodiodas		Veikia kaip šviesos daviklis (apšvietus padidėja laidumas atbuline kryptimi). Paprasti diodai kartais irgi naudojami kaip fotodiodai, jei jų korpusas pagamintas iš šviesai laidžios medžiagos. Skirtingai nuo fotorezistoriaus, fotodiodas į šviesą reaguoja labai greitai. Pavyzdžiui, elektrinėje gitaroje jų greičio pakanka registruoti stygos virpesiams.
9	Šviesos diodas		Srovei tekant tiesiogine kryptimi, šie diodai šviečia. Dėl didelio greitaeigiškumo tinka ir duomenims perduoti.

Tranzistorius – puslaidininkinis įtaisas, paprastai naudojamas elektroniniams signalams sustiprinti ar nukreipti. Tranzistorius yra fundamentali kompiuterių ir kitų modernių elektroninių prietaisų detalė. Išrastas 1948 m.



Dvipolis tranzistorius. Tranzistorius susideda iš puslaidininkio, kuriame yra trys skirtingo pralaidumo sritys. Pralaidumo sritys gali būti n (neigiamo) arba p (teigiamo) tipo. N tipo laidumo puslaidininkiai turi laisvųjų elektronų, kurie gali laisvai judėti, tuo tarpu p tipo turi vadinamąsias skyles, t. y. vietas, kurios turėtų būti išpildytos elektronais, tačiau nėra. Verta pažymėti, jog

patys puslaidininkiai, nepriklausomai nuo tipo, yra neutralūs. Tranzistorius gali būti sudarytas iš dviejų n ir vieno p tipo (npn) arba iš dviejų p ir vieno n (pnp), tačiau abiejų tipų tranzistoriuose visada bus dvi np sandūros. Puslaidininkio galuose yra kolektorius ir emiteris, o viduryje esanti puslaidininkio dalis yra prijungta prie bazės, kuri nulemia, ar srovė tarp kolektoriaus ir emiterio tekės.



Veikimas. Ties np sandūromis pertekliniai elektronai užpildo vadinamąsias skyles taip sukurdami teigiamą ir neigiamą sritis. Sritis aplink np sandūrą su užpildytomis sritimis yra įelektrinta neigiamai, o sritis, netekusi elektronų, – teigiamai. Dėl šio įelektrinimo elektronai nebegali toliau keliauti ir užpildyti likusių skylių. Natūralios būsenos, dėl esančio barjero, tranzistorius nepraleidžia ar praleidžia tik dalį elektros srovės. Tačiau jeigu prijungsime NPN tipo tranzistoriaus bazę prie mažo galingumo šaltinio teigiamo kontakto, pertekliniai elektronai bus pajėgūs įveikti barjerą ne tik dėl to, kad pats barjeras susitrauks ir dėl papildomos traukos (teigiamas krūvis pagreitins elektronus).

Indukcinė ritė arba induktyvinė ritė (iš lotynų k. *inductio* – sužadinimas) – pasyvus iš dviejų galų sudarytas elektroninis komponentas, kuris geba kaupti energiją magnetiniame lauke, per jį tekant elektros srovei. Indukcinę ritę paprastai sudaro į ritę suvyniota izoliuota viela. Žymimas elektroniniu simboliu  Keičiantis rite tekančiai srovei per laiką kintantis magnetinis laukas ritėje sukelia elektrovarą (elektrinę įtampą), kuri priešinasi ją sukūrusios srovės pokyčiui. Tokiu būdu induktoriai priešinasi bet kokiems per juos vykstantiems srovės pokyčiams. Indukcinę ritę apibūdina induktyvumas: įtampos ir srovės kitimo greičio santykis. SI sistemoje induktyvumo vienetas yra henris (H), pavadintas XIX a. amerikiečių mokslininko Džozefo Henrio vardu. Matuojant magnetines grandines, šis vienetas yra lygus vėberiui / amperui. Indukcinių ričių induktyvumas paprastai svyruoja nuo 1 µH (10−6) iki 20 H. Daugeliu atvejų ričių viduje yra iš geležies arba ferito pagaminta magnetinė šerdis, padidinanti magnetinį lauką, taigi ir induktyvumą. Kartu su kondensatoriais ir rezistoriais indukcinės ritės yra vienas iš trijų pasyvių linijinių grandinių elementų, sudarančių elektrines grandines.

VI. 3D spausdinimo ir projektavimo (piešimo) pagrindai

3D spausdinimo skyriuje supažindinama su 3D spausdinimo technologija ir pateikiamas trumpas įvadas į 3D objektų kūrimą „ThinkerCad“ aplinkoje. 3D projektavimo ir spausdinimo įgūdžiai suteikia galimybę kurti skirtingas robotų formas, pritaikytas ir optimizuotas pagal paskirtį. Tai padeda sumažinti robotų gabaritus ir padidinti patikimumą nustatytų užduočių atlikimui.

Įvadas

3D spausdinimas arba adityvioji (pridėtinė) gamyba (angl. *additive manufacturing*) – trimačio, vientiso, praktiškai bet kokios formos objekto gaminimo procesas iš skaitmeninio modelio. 3D spausdinimo principas – skirtingomis formomis sudedami sluoksniai. Tuo 3D spausdinimas skiriasi nuo tradicinių apdorojimo technikų, kuriomis medžiagos dažniausiai apdorojamos pjovimo ar atėmimo būdu (angl. *subtractive manufacturing*). 3D spausdintuvas yra pramoninio roboto rūšis, kuri pagal kompiuterio komandas sugeba atlikti papildomas funkcijas.

Terminologija. Terminas „adityvi gamyba“ apibrėžia technologijas, kuriomis kuriami objektai naudojant nuoseklaus sluoksniavimo techniką. Tokia gamyba gali būti taikoma įvairiuose gamybos cikluose: tiek priešgamybinėje (t. y. greitų prototipų) plataus masto produkcijoje, tiek mechaniniame ar pogamybiniame apdirbime. Gamybos, ypač mašininio apdirbimo sferos, šalinimo metodai siejami su tradiciškesniais metodais. Terminas „atėmimo gamyba“ yra neseniai išvystytas retronimas, siekiant atskirti jį nuo naujųjų adityvių gaminimo technikų. Nors gamyba įtraukė metodus, kurie iš esmės yra „adityvūs“ jau daugelį amžių (pavyzdžiui, plokščių, lakštų, kaltinių ir valcavimo darbų sujungimas kniedėmis ar varžtais, juos naudoja kalviai ar suvirintojai), tai neįtraukė informacinių technologijų maketais paremto komponento apibrėžimo. Tradiciškai apdirbimas (tam tikrų formų kūrimas) buvo atimamasis (drožimas, tekinimas, frezavimas, gręžimas, šlifavimas).

3D spausdinimas

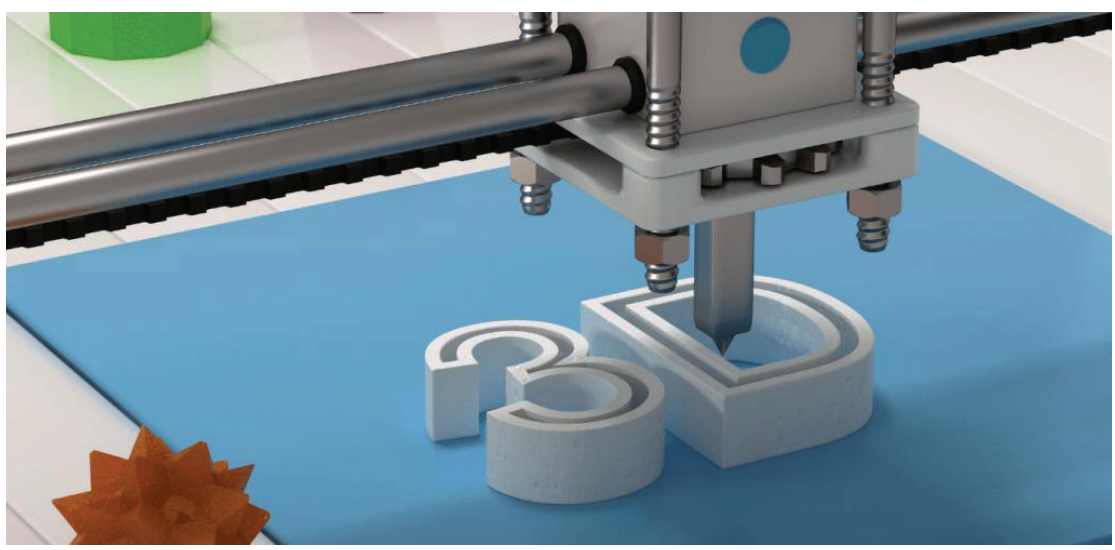
Adityvus spausdinimas naudoja **virtualius modelius** iš projektavimo sistemų arba programų ir supjausto juos į skaitmeninius skersinius pjūvius tam, kad mašina galėtų juos

naudoti kaip spausdinimo gaires. Priklausomai nuo spausdintuvo, pamatinė arba įrišamoji medžiaga paprastai yra laikoma padėta ant tam skirto paviršiaus tol, kol medžiagos sluoksniavimas nėra užbaigtas ir paskutinis 3D modelis nėra gatavai išspausdintas. Standartinė duomenų sąveika tarp CAD programinės įrangos ir pačios mašinos yra STL rinkmenos formatas.

Kad įvyktų **spausdinimo procesas**, mašina „skaito“ projektą iš STL rinkmenos ir kloja nuoseklius skysčio, miltelių, popieriaus ar lapo sluoksnius, siekiant iš virtinės skersinių pjūvių pastatyti modelį. Šie sluoksniai, atitinkantys virtualius skersinius pjūvius iš CAD modelio, yra sujungiami arba automatiškai sulydomi tam, kad būtų sukurta galutinė forma. Pirminis šios technikos pranašumas yra gebėjimas sukurti praktiškai bet kokią formą ar geometrinį objektą. Spausdintuvo raiška nurodo sluoksnių storį ir X-Y raišką, taškų skaičių viename colyje (angl. *dots per inch*) ar mikrometre. Nors standartinis sluoksnio storis yra 100 µm/tašk. (250 taškų colyje), kai kurios mašinos, pavyzdžiui, „Object Connex“ ir „3D System‘s Pro Jet“, gali spausdinti 16 µm/tašk. (1600 taškų colyje) storio sluoksnius. X-Y raiška yra lyginama su lazerinių spausdintuvų raiška. Dalelės (3D taškai) yra nuo 50 iki 100 µm/tašk. (510–250 taškų colyje) skersmens.

Modelio gaminimas gali užtrukti nuo kelių valandų iki kelių dienų, priklausomai nuo jo sudėtingumo ir gaminimo metodo. Adityvios sistemos standartiškai gali sumažinti šį laiką iki kelių valandų, tačiau tai priklauso nuo naudojamo spausdintuvo tipo bei vienu metu gaminamų modelių skaičiaus ir dydžio. Tradiciniai būdai, tokie kaip injekcinis liejimas (angl. *injection molding*), gali būti pigesni gaminant polimero produktus dideliais kiekiais, tačiau adityvus gaminimas gali būti greitesnis, labiau prisitaikantis ir pigesnis gaminant pakankamai mažą kiekį dalių.

3D spausdintuvai suteikia galimybę dizaineriams ir idėjos vystymo komandoms dalis ar visą projektą gaminti naudojant ekrano dydžio spausdintuvą.



41 pav. 3D spausdinimas

Adityvūs procesai. Nuo 8-ojo XX a. dešimtmečio buvo išrasta keletas skirtingų 3D spausdinimo procesų. Spausdintuvai buvo dideli, brangūs bei labai ribotų galimybių. Šiais laikais pasiekiama daug adityvių procesų. Norint sukurti tam tikras dalis, naudojamas

skirtingas sluoksnių išdėstymas bei kitokios medžiagos. Kai kurie metodai ištirpdina arba suminkština medžiagas tam, kad sukurtų sluoksnius, pavyzdžiui, selektyvus lydymas lazeriu (SLM) ar tiesioginis metalo kietinimas lazeriu (DMLS), selektyvus kietinimas lazeriu (SLS), lydyto nusėdimo modeliavimas (FDM), tuo tarpu kiti apdoroja skystas medžiagas naudodami įmantrias technologijas, pavyzdžiui, stereolitografiją (SLA). Naudojant laminuotą objektų gamybą (LOM), ploni sluoksniai yra supjaustomi ir sujungiami (pavyzdžiui, popierius, metalas, polimeras). Kiekvienas metodas turi savo privalumų ir trūkumų, dėl to kai kurios kompanijos siūlo medžiagos, iš kurios bus gaminamas objektas (milteliai, polimeras ir t. t.), pasirinkimą. Kai kurios kompanijos naudoja popierių kaip statybinę medžiagą, gaminant patvarų prototipą.

3D spausdinimo technologijos / būdai

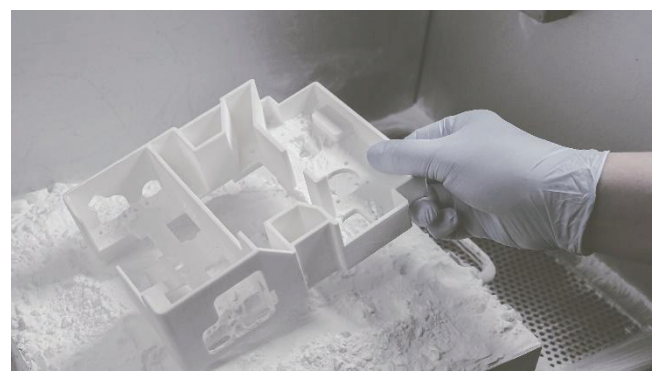
Renkantis įrenginį, pagrindinis dėmesys dažniausiai skiriamas greičiui, 3D spausdintuvo kainai, atspausdinto prototipo kainai bei spalvų ir medžiagų pasirinkimo galimybėms. Spausdintuvai, tiesiogiai naudojantys metalą, yra brangūs, tačiau kai kuriais atvejais ne tokie brangūs spausdintuvai gali būti naudojami gaminant klojinius, kurie vėliau bus naudojami gaminant metalo dalis.

TIPAS	TECHNOLOGIJA	MEDŽIAGOS
Ekstruzinis	Lydyto nusėdimo modeliavimas (FDM)	Termoplastikas (pvz., PLA, ABS), HDPE, eutektiniai metalai, maisto medžiagos, guma, modeliavimo molis, plastilinas, RTV silikonas, porcelianas, metalo molis (taip pat tauriųjų metalų molis)
Laidinis	Laisvų elektronų pluošto gamyba (EBF3)	Beveik bet koks metalo lydinys
Granuliuotas	Tiesioginis metalo lydymas lazeriu (DMLS)	Beveik bet koks metalo lydinys
	Lydymas elektronų spinduliais (EBM)	Titano lydinys
	Selektyvus lydymas lazeriu (SLM)	Titano lydinys, kobalto ir chromo lydinys, nerūdijantis plienas, aliuminis
	Selektyvus šiluminis kietinimas (SHS)[13]	Termoplastiniai milteliai
	Selektyvus kietinimas lazeriu (SLS)	Termoplastikas, metalo milteliai, keramikos milteliai
Miltelių sluoksnio 3D spausdinimas purkštuko galvute	Gipso 3D spausdinimas (PP)	Gipsas

Laminuotas	Laminuota objektų gamyba (LOM)	Popierius, metalo folija, plastiko plėvelė
Polimerizuotas šviesa	Stereolitografija (SLA)	Fotopolimeras
	Skaitmeninis šviesos apdorojimas (DLP)	Polimeras

Ekstruzijos nusodinimas. Lydyto nusėdimo modeliavimą (angl. *fused deposition modeling*) 9-ajame XX a. dešimtmetyje išrado S. Skotas Krampas, o 10-ą dešimtmetį jį komerciniams tikslams pritaikė Stratasys. Dabar, pasibaigus šios technologijos patentui, yra didžiulės atvirų šaltinių bendruomenės (pavyzdžiui, „RepRaps“), komerciniai ir buitiniai variantai, kurie naudoja šio tipo 3D spausdintuvą. Šios technologijos sukūrimas paskatino didelį tos srities įrenginių kainų kritimą. Lydyto nusėdimo modeliavimo metu maketas ar jo dalis yra gaminami lydant spausdinimo medžiagą, kuri greitai stingsta, suformuodama sluoksnį. Termoplastinis plaušelis ar metalinė viela, suvyniota į ritę, yra tiekiama į purkštuko galvutę. Įkaitinta purkštuko galvutė reguliuoja liejamos medžiagos srautą. Tradiciškai purkštuko galvutės padėties valdymui yra naudojamas valdymo variklis. Purkštukas gali būti judinamas vertikalia bei horizontalia kryptimi. Šio mechanizmo kontrolę vykdo kompiuterizuotos gamybos (angl. *computer-aided manufacturing* (CAM) programa, valdanti mikrokontrolerį. Liejimui naudojami skirtingi polimerai, tokie kaip akrilnitrilo butadieno stirenas (ABS), polikarbonatas (PC), polilaktidas (PLA), didelio tankio polietilenas (HDPE) ir polifelinsuflonas (PPSU). Apskritai, polimeras yra plaušo forma, pagaminta iš gamtinės dervos. FDM turi tam tikrus gaminamų formų apribojimus. Pavyzdžiui, FDM dažniausiai negali pagaminti stalaktito formos struktūrų. Tokios struktūros yra vengiamos arba konstruojamos sukuriant pagalbinę medžiagą, kuri gali sulūžti maketo baigimo metu.

Granuliuotų medžiagų kietinimas. Šia technika sulydomos sluoksnių dalys, tada darbas perkeliamas žemyn, pridedant kitą granulių sluoksnį ir kartojant šį procesą tol, kol figūra pagaminta. Šis procesas naudoja nesulydytą terpę iškilimams prilaikyti ir plonoms sienelėms gaminti, taip sumažinant pagalbinių įtaisų poreikį. Kad objektas taptų vientisos masės, reikalingas lazeris. Šio proceso pavyzdžiai yra selektyvus kietinimas lazeriu (SLS) su metalais ir polimerais (PA, PA-GF, kietuoju GF, PEEK, PS, aliuminiu, karbamidu, elastomeru) ir tiesioginis metalo kietinimas lazeriu (DMLS).



Selektyvų kietinimą lazeriu (angl. *selective laser sintering*) 9-ojo dešimtmečio viduryje išrado ir patentavo dr. Carl Deckard ir dr. Joseph Beatman Teksaso universitete, Ostine, jį finansavo DARPA.

Laminavimas. Kai kuriuose spausdintuvuose popierius gali būti naudojamas kaip statybinė medžiaga, sumažinanti spausdinimo išlaidas. XX a. paskutinįjį dešimtmetį kai kurios kompanijos pardavinėjo spausdintuvus, kurie išpjaudavo skersinius pjūvius iš specialaus lipnaus popieriaus, naudojant anglies dioksido lazerius, ir tada juos kartu laminuodavo. 2005 m. Mcor Technologies Ltd. išvystė kitokį procesą, naudodami paprastus popieriaus lapus, volframo karbido geležtę formai išpjauti ir selektyvų nusodinimą klijais bei spaudimą prototipui sukibinti. Taip pat yra daugybė kompanijų, pardavinėjančių spausdintuvus, kurie naudoja plastiką ir metalo plokštes laminuotiems objektams atspausdinti.

Fotopolimerizacija. Stereolitografiją 1986 metais patentavo Čarlzas V. Hulas. Fotopolimerizacija pirmiausia yra naudojama stereolitografijoje (SLA) norint iš skysčio pagaminti kietą kūną. Skaitmeninio šviesos apdorojimo metu (angl. *digital light processing*) skysto polimero indas yra veikiamas šviesos, sklindančios iš skaitmeninio šviesos apdorojimo projektoriaus. Apšviestas skystas polimeras kietėja. Tada sukonstruota forma pamažu juda žemyn ir skystas polimeras yra vėl apšviečiamas. Šis procesas kartojasi tol, kol padaromas maketas. Tada skystas polimeras išdžiovinamas, paliekant vientisą modelį. Kiekvienas fotopolimero sluoksnis yra apdorojamas UV šviesa, taip sukuriant visiškai apdorotus maketus, kurie gali būti iš karto naudojami. Atraminė į gelį panaši medžiaga yra pašalinama rankomis arba vandens srautu.

Spausdintuvų panaudojimas

Spausdintuvų industrinis panaudojimas. Industriniai 3D spausdintuvai egzistavo nuo ankstyvojo 9-ojo dešimtmečio ir buvo naudojami sparčiai prototipų gamybai bei tyrimams. Tai yra didelės mašinos, kurios naudoja patentuotus miltelių formos metalus, liejamas terpes (pavyzdžiui, smėlį), plastiką, popierių arba šerdeles ir yra naudojamos sparčiam prototipų gaminimui universitetuose bei komercinėse kompanijose.

Nuo 2012 metų spalio mėnesio „Stratasys“ gamina industrines adityvaus gaminimo sistemas,

SEQ Figure * ARABIC 44 pav. „Stratasys“ 3D spausdintuvas

kurių kainos svyruoja nuo 2 000 iki 500 000 JAV dolerių. „General Electric“ naudoja aukštos klasės modelius gamindami turbinų dalis. „General Electric“ į šią technologiją investavo 1 milijardą dolerių.

Panaudojimas buityje. Šioje srityje daug nuveikė „Pasidaryk pats“ (DIY) ankstyvos naudotojų bendruomenės. „RepRap“ yra vienas ilgiausiai gyvuojančių projektų. „RepRap“ projektas siekia kurti nemokamą ir atvirojo kodo kompiuterinę įrangą (FOSS) 3D spausdintuvams, kurių techninės specifikacijas patvirtino „GNU General Public License“ licencija. Spartus 3D spausdintuvų vystymasis sulaukia daugelio sričių susidomėjimo, nes jis suteikia didžiules pritaikymo galimybes ir panaudojimą. Nuo 2010 metų 3D spausdintuvų kaina dramatiškai krito. Įrenginiai, kurie kainuodavo 20 000 JAV dolerių, dabar kainuoja mažiau negu 1 000 dolerių. Kadangi 3D spausdintuvų kaina krito, išaugo jų paklausa asmeninių produktų savarankiškai gamybai. Taip pat 3D spausdinimo produktai, naudojami buityje, gali sumažinti gamybos poveikį aplinkai, sumažindami medžiagų naudojimą bei platinimą.

Masinis pritaikymas individualiam vartotojui. Kompanijos yra sukūrusios paslaugų, kur pirkėjai gali pagal savo norus modeliuoti objektus, naudodami nesudėtingą internetinę programinę įrangą, ir užsisakyti sukurtus produktus kaip originalius 3D objektus. Dabar vartotojams leidžiama kurti dėklus savo mobiliesiems telefonams. „Nokia“ išleido 3D dizainus savo dėklams tam, kad vartotojai turėtų galimybę patys susikurti ir gauti atspausdintą 3D dėklą.

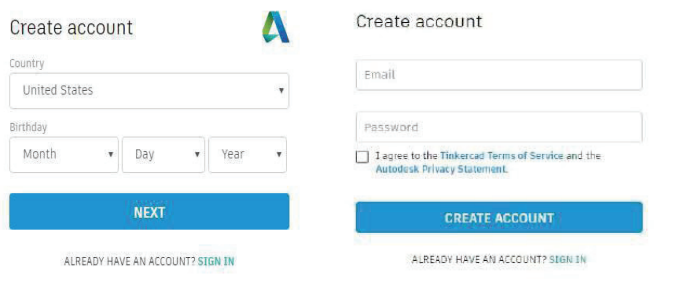
Drabužiai. 3D spausdinimas paplito drabužių industrijoje, madų dizaineriams eksperimentuojant su 3D atspausdintais maudymosi kostiumėliais, batais ir suknelėmis. Komercinėje produkcijoje „Nike“ naudoja 3D spausdinimą kurdami prototipus.

3D projektavimas

Šiame vadovėlyje 3D failų kūrimui naudosime debesų (cloud) pagrindu sukurtą programą „Tinkercad“.

Naudojimąsi programa pradėsime nuo paskyros sukūrimo.

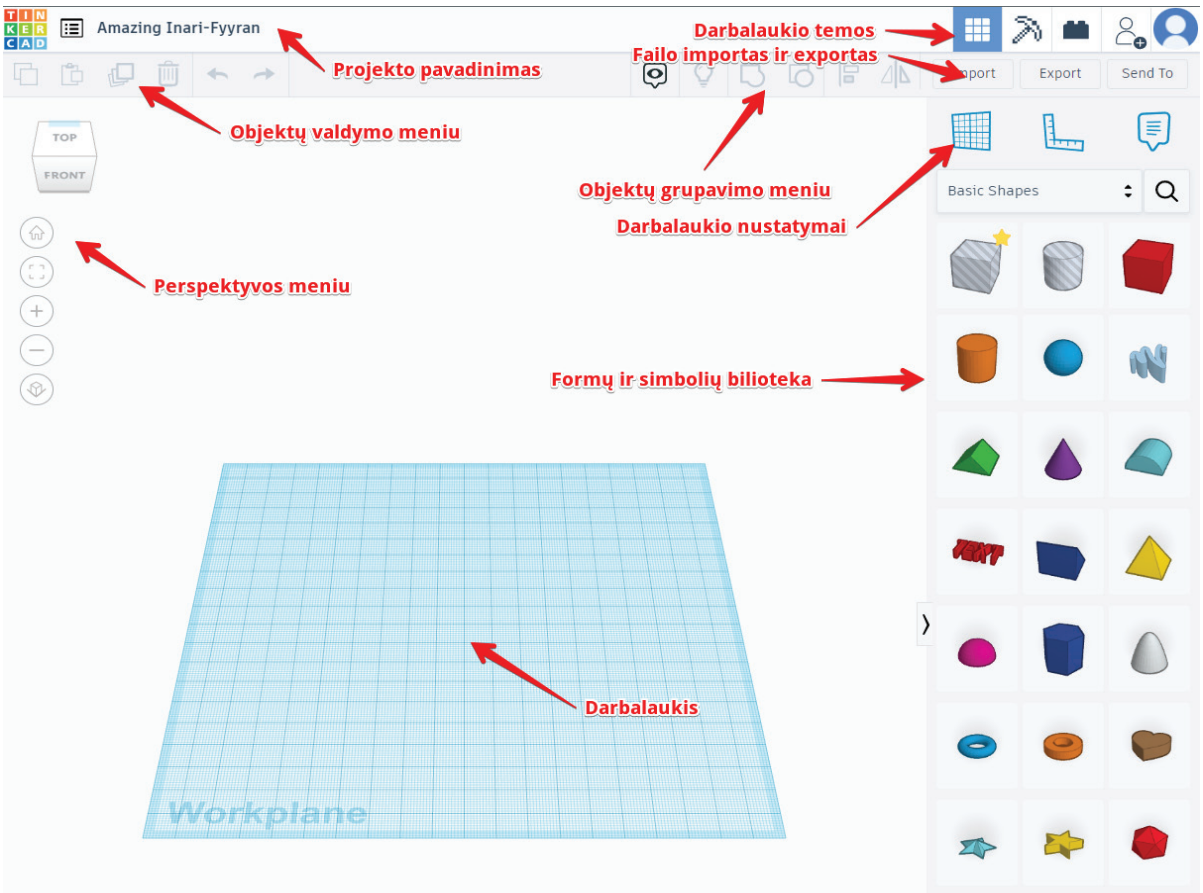
Paskyros sukūrimas:



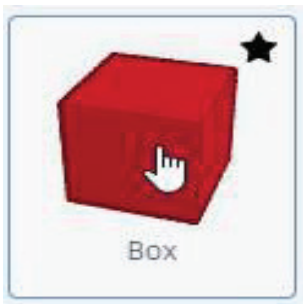
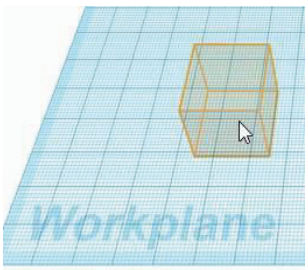
Interneto naršyklėje įveskite adresą *www.tinkercad.com*

Sekite paskyros sukūrimo gidą

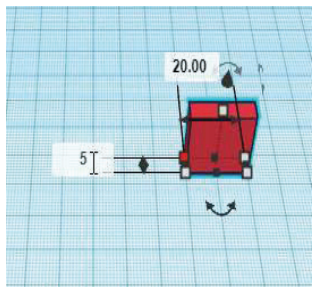
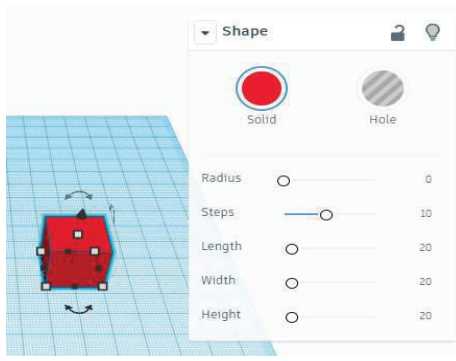
„Tinkercad“ projektavimo aplinkos apžvalga:



Objekto pasirinkimas:

	Pasirinkite formą, kurią norėtumėte pridėti prie savo darbo plokštumos;
	Užveskite pelės žymeklį ant pasirinktos formos; Pažymėkite dešiniu pelės klavišu ir neatleisdami pelės klavišo perneškite formą į darbalaukį; Atleiskite pelės klavišą, kad įkeltumėte formą į redagavimo lauką.

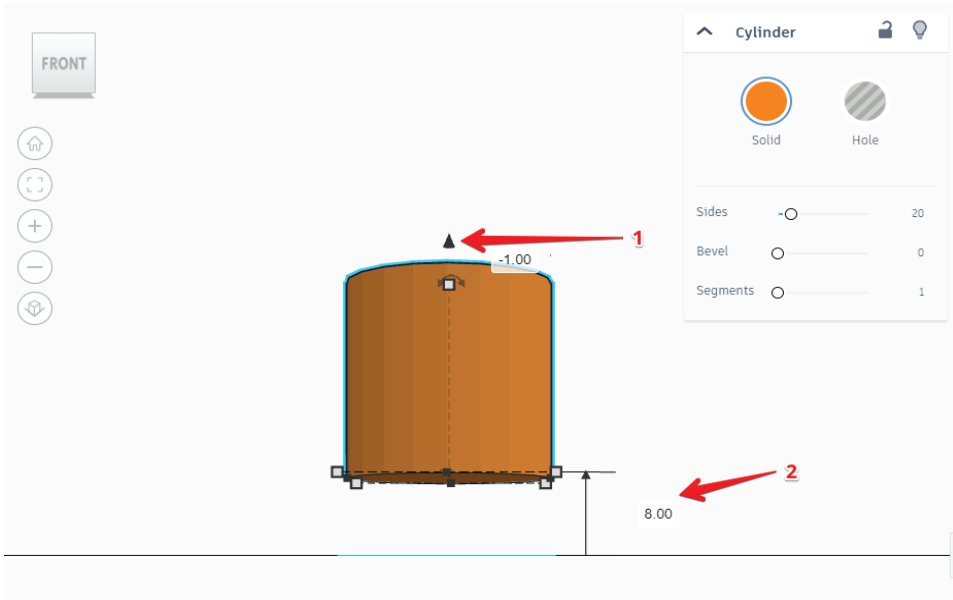
Objekto parametrų keitimas. Yra keli būdai, kaip galima keisti objekto parametrus:

	Pasirinkus objektą, šalia jo atsiranda pilki kvadratėliai. Keičiant pilko kvadratėlio padėtį, keičiasi atitinkamas objekto parametras;
	Antras būdas – keisti parametrus per objekto meniu.

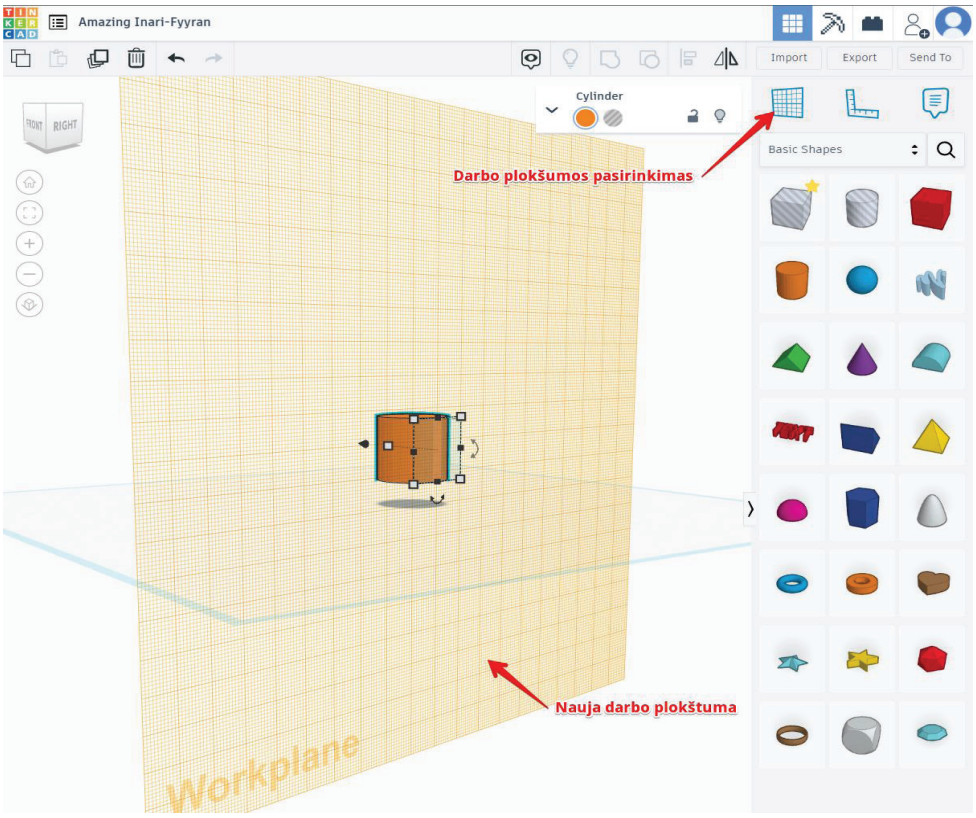
Objekto aukščio padėties keitimas. Objekto aukštį galima keisti dviem būdais:

Keičiant juodos rodyklės padėtį virš pasirinkto objekto;

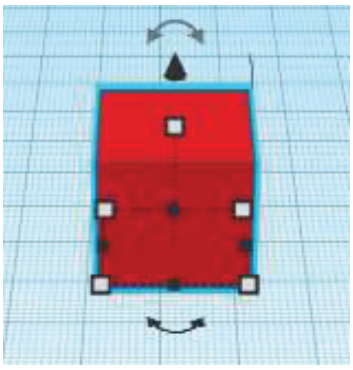
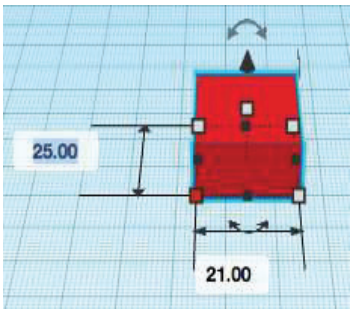
Keičiant atstumo matmenį.

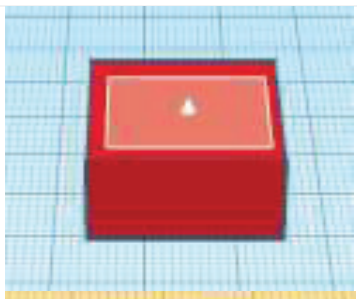
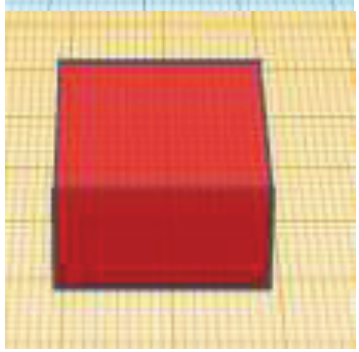
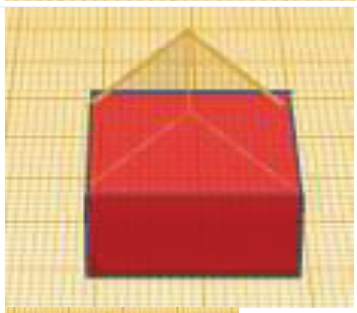
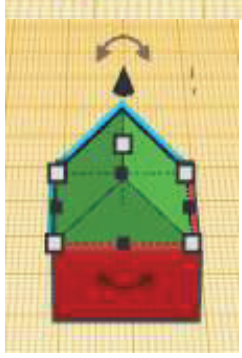
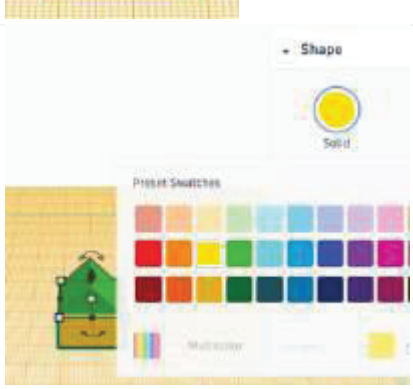


Darbo plokštumos. Kuriant 3D objektus, dažniausiai dirbama keliose plokštumose. Pavyzdžiui, sukūrėte objekto pagrindą ir toliau norite redaguoti vieną iš objekto šonų, tuomet reikia sukurti plokštumą toje objekto dalyje. Norint sukurti naują plokštumą, ekrano viršuje, kairėje, pasirinkite plokštumos piktogramą ir perneškite į atitinkamą objekto pusę.



Paprasto trimačio objekto sukūrimas. Šiame pavyzdyje sukursime namą iš kelių objektų kvadrato ir trikampio:

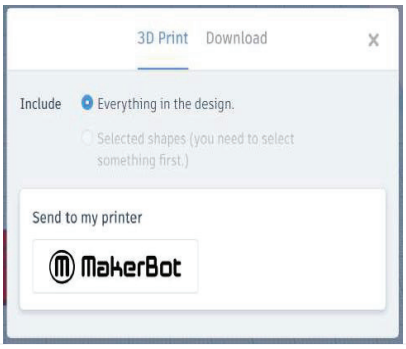
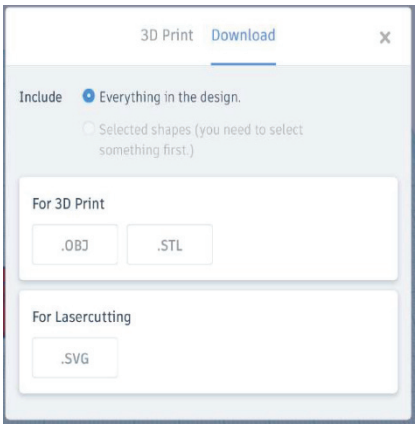
	Pasirinkite kvadrato formą ir perneškite į darbinę plokštumą;
	Pakeiskite kvadrato plotį (25 mm) ir ilgį (21 mm);

 	Sukurkite naują darbo plokštumą viršutinėje kvadrato plokštumoje;
 	Pasirinkite stogo formą (Roof) iš objektų bibliotekos ir perneškite virš kvadrato;
	Savo nuožiūra pakeiskite objektų spalvas.

Sveikiname, jūsų pirmas 3D objektas sukurtas!

Projekto išsaugojimas. Išsaugoti projektą lengva, nes tai automatiškai už jus atlieka programa. Tačiau norint atspausdinti 3D spausdintuvu, projektą reikia eksportuoti į failą.

3D spausdinimas. Kai būsite pasiruošę šiam veiksmui, spustelėkite mygtuką „eksportuoti“ ir atlikite šiuos veiksmus:

	Pasirinkite mygtuką „eksportuoti“; Atsidariusiame meniu viršuje turite kelis pasirinkimus: „3D Print“ – šiuo atveju jūsų projektas bus siunčiamas tiesiai į jūsų pasirinkto 3D spausdintuvo nuotolinę failų spausdinimo platformą.
	Download – pasirinkus „download“ projektas išsaugomas pasirinktu formatu. Išsaugotą failą galima perkelti į 3D spausdintuvą gaminio sukūrimui.

Failo pasirinkimas. Galima pasirinkti dvi kategorijas – 3D spausdinimo ir lazerinių staklių. Norėdami objektą perkelti į lazerines stakles, turite išsaugoti *.SVG* formatu. Jeigu objektą norite spausdinti 3D spausdintuvu, projektą galite išsaugoti *.OBJ* arba *.STL* formatu.

STL ir *OBJ* formatai užkoduoja 3D modelių geometriją per daugiakampius tinklus. Tai reiškia, kad kiekvienas 3D objekto paviršius yra pavaizduotas apytiksliu sujungtų daugiakampių tinklu, o galutinė skiriamoji geba priklauso nuo to, kiek daugiakampių yra.

„Standard Tesselation Language“, arba STL, buvo sukurta specialiai 3D spausdinimui dar devintajame dešimtmetyje. Jis pagamintas tik iš trikampio tinklelio ir yra labai lengvas formatas, kuriame pateikiama mažiau modelio informacijos.

OBJ, arba „Wavefront Object“, pavadintas kompiuterinės grafikos įmonės „Wavefront Technologies“, kuri sukūrė programinės įrangos sprendimus kino pramonei, vardu. Tai atvirojo kodo formatas, gerai įsitvirtinęs skaitmeninės grafikos pasaulyje ir vis labiau populiarėjantis 3D spausdinimo bendruomenėje.

Šie du formatai skiriasi keliais būdais. STL vaizduoja objektų paviršių kaip tinklelį, sudarytą tik iš trikampių, o tai yra pakankamai paprasta geometrija. Didesniam modelio tikslumui reikalingas didesnis trikampių skaičius, todėl failo dydis beveik eksponentiškai

padidėja. Kita vertus, OBJ tame pačiame faile yra keli skirtingi daugiakampiai ir palaiko tikslų paviršiaus kodavimą. Vietoj briaunų formų paviršiai taip pat gali būti apibrėžti NURBS **N in- U** niform **R** ational **B** asis **S**.

„Tinkercad“ projektavimo aplinka yra pakankami intuityvi. Šiame skyriuje pateikti pirmieji žingsniai su šia programa. Tolimesniam programos įvaldymui rekomenduojame susipažinti su integruotomis pamokomis, kurios nuosekliai įveda į 3D projektavimo pasaulį ir išmoko kurti sudėtingus objektus.

https://www.tinkercad.com/learn

„Tinkercad“ projektavimo programa – viena iš nedaugelio programų, kuri turi galimybę projektuoti elektronines grandines virtualioje aplinkoje. Tai padeda tausoti laiką ir resursus, kuomet dirbama klasėje su idėjomis ir prototipais.

https://www.tinkercad.com/circuits

„Tinkercad“ programos unikalus išskirtinumas yra tas, kad joje galima programuoti virtuliai sukurtas grandines. Tai kone greičiausias būdas ištestuoti idėją, ypač dirbant su moksleivių grupėmis.

https://www.tinkercad.com/codeblocks

Vienas iš aktualiausių programos „Tinkercad“ išskirtinumų mokytojams – tai integruotas klasės valdymas. Mokymas naudojant „Tinkercad“ dar niekada nebuvo toks paprastas. Siųskite ir gaukite užduotis, stebėkite mokinių pažangą ir priskirkite naujas veiklas – visa tai „Tinkercad Classrooms“.

https://www.tinkercad.com/classrooms-resources.

VII. Pamokų pavyzdžiai

Šiame skyriuje pateikiami pamokų pavyzdžiai kiekvienam robotų tipui, aprašytam edukacinių robotų aprašymo skyriuje. Pamokas galima naudoti bet kurio mokymosi etapo metu. Nėra specialių reikalavimų išankstiniam pasiruošimui. Svarbu atkreipti dėmesį į rekomenduojamą amžiaus grupę.

1. „LEGO Education WeDo 2“ roboto konstravimo ir programavimo pavyzdinė pamoka – palydovas

Santrauka

Pamokoje supažindinama su konstravimu ir programavimu panaudojant variklį, priverčiant jį suktis nustatytą laiką ir keisti kryptį.

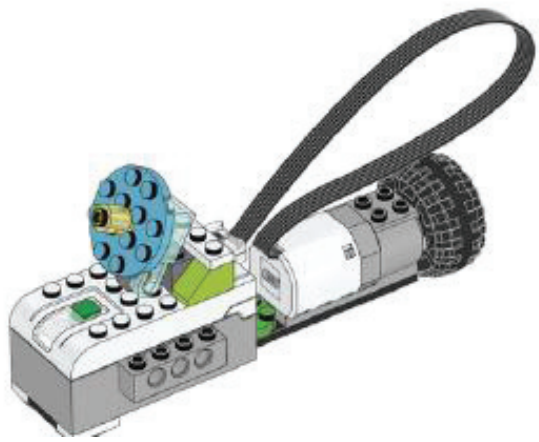
- Trukmė 0–30 min.;
- Lygis – pradedantiesiems (1–2 klasė).

Mokymosi tikslai:

- LEGO modelio konstravimo įgūdžių lavinimas;
- Susieti modelį su realaus pasaulio įrenginiu;
- Variklio funkcijų programavimas.

Priemonės:

- „LEGO Education WeDo 2.0“ konstruktorius;
- „LEGO Education WeDo 2.0“ programa.
- Konstravimo instrukcijos: https://education.lego.com/v3/assets/blt293eea581807678a/blt7967b7fc191ba11e/5ebea0686676f37c355e7f63/moving-satellite-instructions.pdf



Įvadas

Įvadas. Makso ir Mijos istorija skirta įtraukti mokinius į pamokos temą ir sudaryti sąlygas diskusijoms klasėje. Garsiai perskaitykite toliau pateiktą istoriją arba duokite savo mokiniams keletą minučių perskaityti ją patiems. Maksas ir Mija klausosi naujienų. Jie girdi apie mokslininkų valdomus palydovus. Kartais palydovai turi judėti, kad išvengtų meteorų. Sukurkite modelį. Kurdami palydovo modelį mokiniai turėtų vadovautis konstravimo instrukcijomis. Atminkite, kad konstravimo laikas gali skirtis, priklausomai nuo mokinių išankstinių žinių apie statybą iš LEGO kaladėlių.

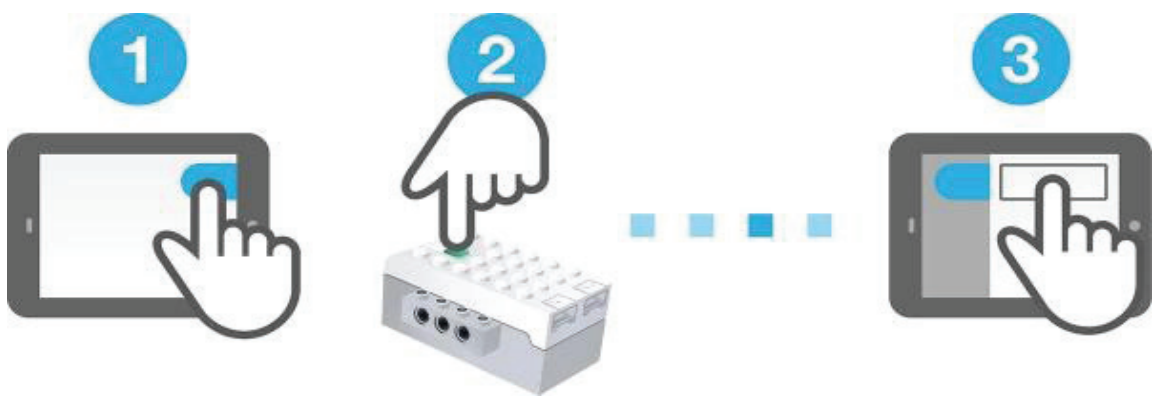
Konstravimas

Atsidarykite **roboto konstravimo instrukcijas**, interneto naršyklėje įvedę adresą:

- https://education.lego.com/v3/assets/blt293eea581807678a/blt7967b7fc191ba11e/5eb ea0686676f37c355e7f63/moving-satellite-instructions.pdf

Programavimas

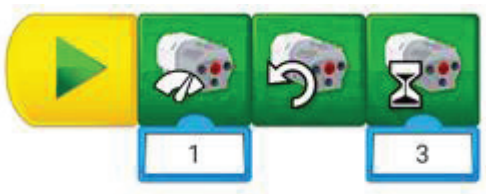
Roboto sujungimas su kompiuteriu:



46 pav. „WeDo2“ roboto sujungimas su kompiuteriu

Patarimas. Kad būtų lengviau, rekomenduojama prieš pamoką pavadinti „Smarthub“ skirtingais pavadinimais (pvz., WeDo001, WeDo002). Tai galima padaryti ryšio centre.

Modelio programavimas. Ši programa privers palydovą suktis „1“ variklio galia 3 sekundes.



47 pav. „WeDo2“ programa

- Sukurkite rodomą programą pernešdami atitinkamą (-us) programos bloką (-us) į pagrindinį programos langą.
- Pakeiskite variklių blokų parametrus, paspausdami variklio bloką, jei norite pakeisti kryptį, arba įveskite skaičių, norėdami pakeisti variklio galingumą arba sukimosi laiką.
- Paspauskite geltoną pradžios bloką, kad paleistumėte programą.

Apibendrinimas

Aptarimas. Padiskutuokite su mokiniais, kaip veikia palydovas. Kai mokiniai sukurs ir paleis savo programas, paprašykite jų pagalvoti ir paaiškinti programą bei palydovo funkciją. Jie turėtų sugebėti atsakyti į tokius klausimus: Kaip programa veikia? Ką veikia skirtingi programų blokai?

Papildoma veikla. Mokiniai gali pakeisti palydovo kryptį vieną kartą paspausdami *Motor ThisWay Block* arba vilkdami *Motor That Way Block* į programos eilutę. Mokiniai gali priversti palydovą suktis ilgesniam laikui, pakeisdami numerį, rodomą *Variklis įjungtas bloke*.

Vertinimas. Savalaikis grįžtamasis ryšys gali padėti mokiniams toliau tobulinti savo naujai įgytus įgūdžius. Tai galima padaryti keliais būdais:

- Stebėti kiekvieno mokinio elgesį, reakciją ir strategijas;
- Užduoti klausimų apie jų mąstymo procesus.

Galite vadovautis toliau pateiktais vertinimo kriterijais:

- Mokinys negali sukurti programos arba tinkamai paaiškinti, kaip veikia modelis (pvz., programos eilutė ir jos skirtingi programos blokai).
- Mokinys, padedant mokytojui, gali sukurti programą arba tinkamai paaiškinti, kaip veikia modelis.
- Mokinys geba sukurti programą ir tinkamai paaiškinti, kaip modelis veikia.
- Mokinys geba sukurti programą ir paaiškinti skirtingus programos blokus bei kokią įtaką jie daro modelio veikimui.

2. „LEGO Education Spike“ roboto konstravimo ir programavimo pavyzdinė pamoka – vėjo greitis

Santrauka

Pamokoje įgyvendinamas vėjo greičio vizualizavimas naudojant kiekybinius duomenis.

- Trukmė 30–45min.;
- Lygis – vidutinis (6–8 klasėms).

Mokymosi tikslas:

- Išmokti valdyti variklį pagal kiekinius orų duomenis.

Priemonės:

- „LEGO Education SPIKE Prime“ konstruktorius;
- Konstravimo instrukcijos:

  https://education.lego.com/v3/assets/blt293eea581807678a/blt4d46952fff51cf5f/5ec9093e15b57c6108323939/wind-speed-bi-pdf-book1of2.pdf

  https://education.lego.com/v3/assets/blt293eea581807678a/blt4d46952fff51cf5f/5ec9093e15b57c6108323939/wind-speed-bi-pdf-book1of2.pdf

  https://education.lego.com/v3/assets/blt293eea581807678a/bltf469d8662a2c7152/5ec9095d6b4f987c36ce7af7/wind-speed-teacher-solution-bi-pdf-book1of1.pdf

49 pav. „Spike robot“ konstruktas

Įvadas

Pamokos planas:

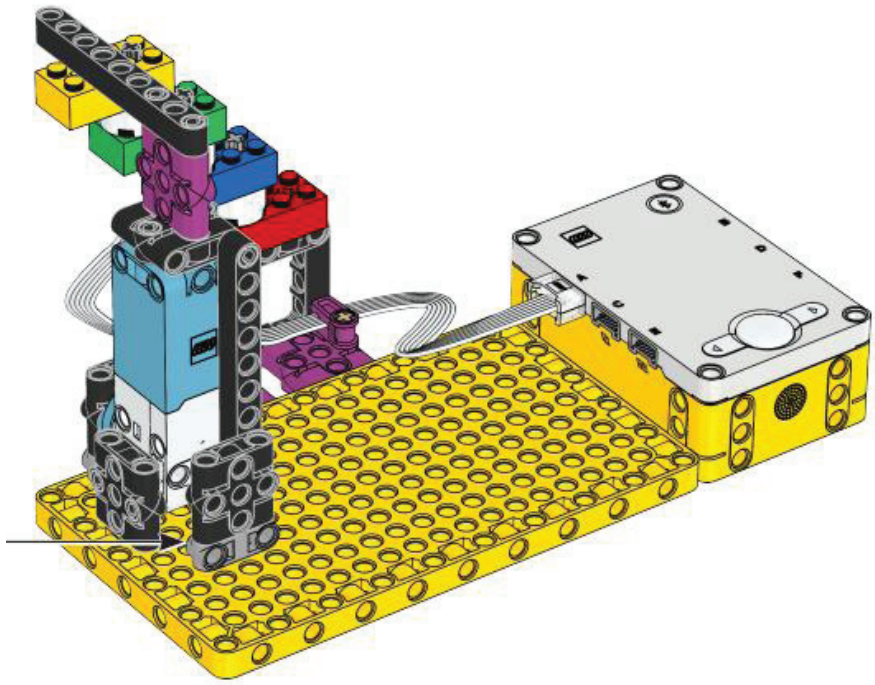
- Sudominti. Pasinaudokite skiltimi toliau esančiame skyriuje „Pradėti diskusiją“ pateiktomis idėjomis, kad įtrauktumėte savo mokinius į diskusiją, susijusią su šia pamoka.
- Tyrinėkite. Paprašykite mokinių dirbti poromis, kad sukurtų vėjo indikatorių. Paprašykite jų paleisti programą. Stebėkite jų reakcijas. Priminkite, kad jie turi įvesti miestą, kad programa veiktų.
- Paaiškinkite. Paprašykite savo mokinių paaiškinti, kaip iš debesies gaunami duomenys rodomi modelyje ir kaip jie atspindi Boforto skalę.
- Patobulinkite. Paprašykite savo mokinių išplėsti savo programas su daugiau IF ELSE sąlygų, kad atsižvelgtų į skirtingus vėjo greičius pagal Boforto skalę. Jie tai gali padaryti padaliję skalę į 4 dalis. Paprašykite mokinių savo programose parodyti vėjo kryptį (pvz., naudodami rodykles ant šviesos matricos).
- Įvertinkite. Pateikite atsiliepimų apie kiekvieno mokinio veiklą. Norėdami supaprastinti procesą, galite naudoti pateiktas vertinimo rubrikas.

Pradėti diskusiją. Pradėkite diskusiją apie vėją. Kalbėkite apie dalykus, kuriuos galite ir ko negalite daryti vėjuotomis dienomis (pvz., skraidinti droną, aitvarą, žaisti futbolą ar beisbolą, rengti vakarėlį lauke). Ištirkite įvairias vėjo greičio klasifikacijas (pvz., Boforto skalę). Paprašykite savo mokinių pagalvoti apie įvairius vėjo matavimo būdus. Paprašykite savo mokinių pažiūrėti šį vaizdo įrašą, kad sužinotumėte, ką jie ketina daryti.

Konstravimas

Atsidarykite **roboto konstravimo instrukcijas**, interneto naršyklėje įvedę adresą:

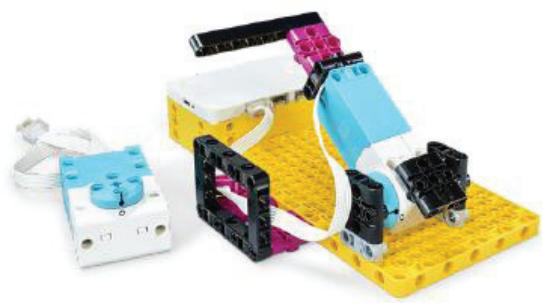
- https://education.lego.com/v3/assets/blt293eea581807678a/blt4d46952fff51cf5f/5ec9093e15b57c6108323939/wind-speed-bi-pdf-book1of2.pdf
- https://education.lego.com/v3/assets/blt293eea581807678a/blt4d46952fff51cf5f/5ec9093e15b57c6108323939/wind-speed-bi-pdf-book1of2.pdf
- https://education.lego.com/v3/assets/blt293eea581807678a/bltf469d8662a2c7152/5ec9095d6b4f987c36ce7af7/wind-speed-teacher-solution-bi-pdf-book1of1.pdf



48 pav. „Spike“ konstruktas

Konstravimo patarimai:

- Teisingai sulygiuokite variklius: įsitikinkite, kad statydami mokiniai tinkamai pastatė variklį. Tai turės įtakos šio modelio programavimui.
- Pakartotinai pritaikykite modelį kitoms pamokoms: jei laikas ribotas, naudokite pagrindinę modelio versiją (t. y. praleiskite simbolių elementus).
- Naudokite šviesos matricą: galite naudoti kitą variklį arba rodykles ant *Hub Light Matrix*, kad būtų rodoma vėjo kryptis.



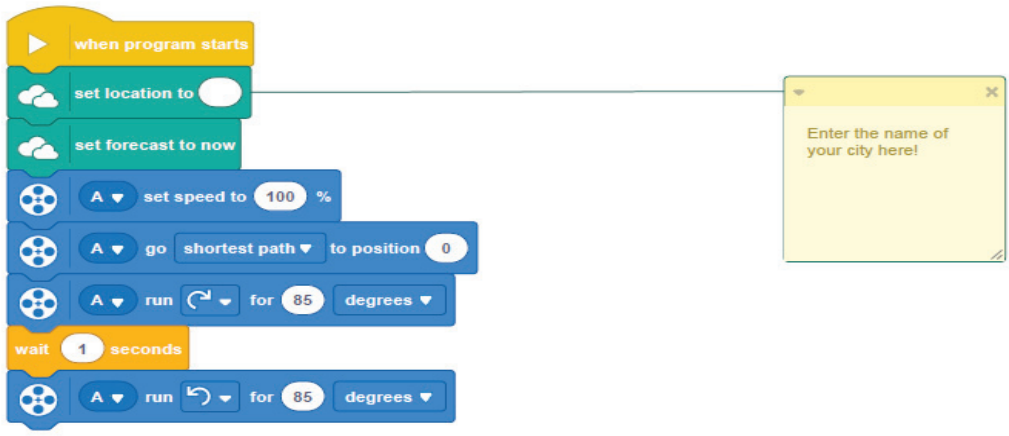
SEQ Figure * ARABIC 50 pav. „Spike robot“ konstruktas

Boforto skalė. Modelio blokų spalvos atitinka Boforto skalę, kurią 1805 m. sukūrė airių hidrografas Francis Beaufort. Ji apibūdinama taip:

- Mėlyna: Boforto skalė 1–3 (1–12 mph arba 0,5–5,5 m/s), nuo lengvo iki švelnaus vėjelio;
- Žalia: Boforto skalė 4–6 (13–31 mph arba 5,5–13,8 m/s) , vidutinio ar gaivaus vėjo;
- Geltona: Boforto skalė 7–9 (31–54 mph arba 13,8–24,4 m/s), stiprus vėjas iki labai stiprus vėjas;
- Raudona: Boforto skalė 10–12 (s), audros iki uragano.

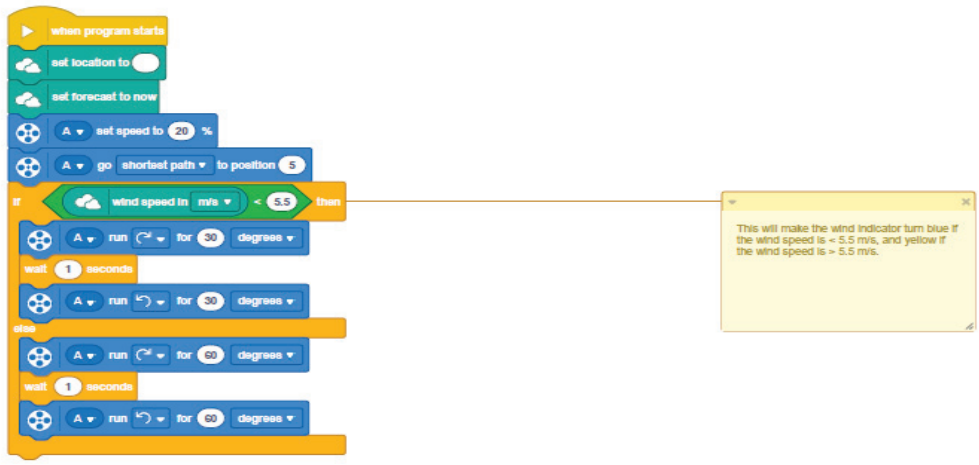
Programavimas

Pagrindinė programa:



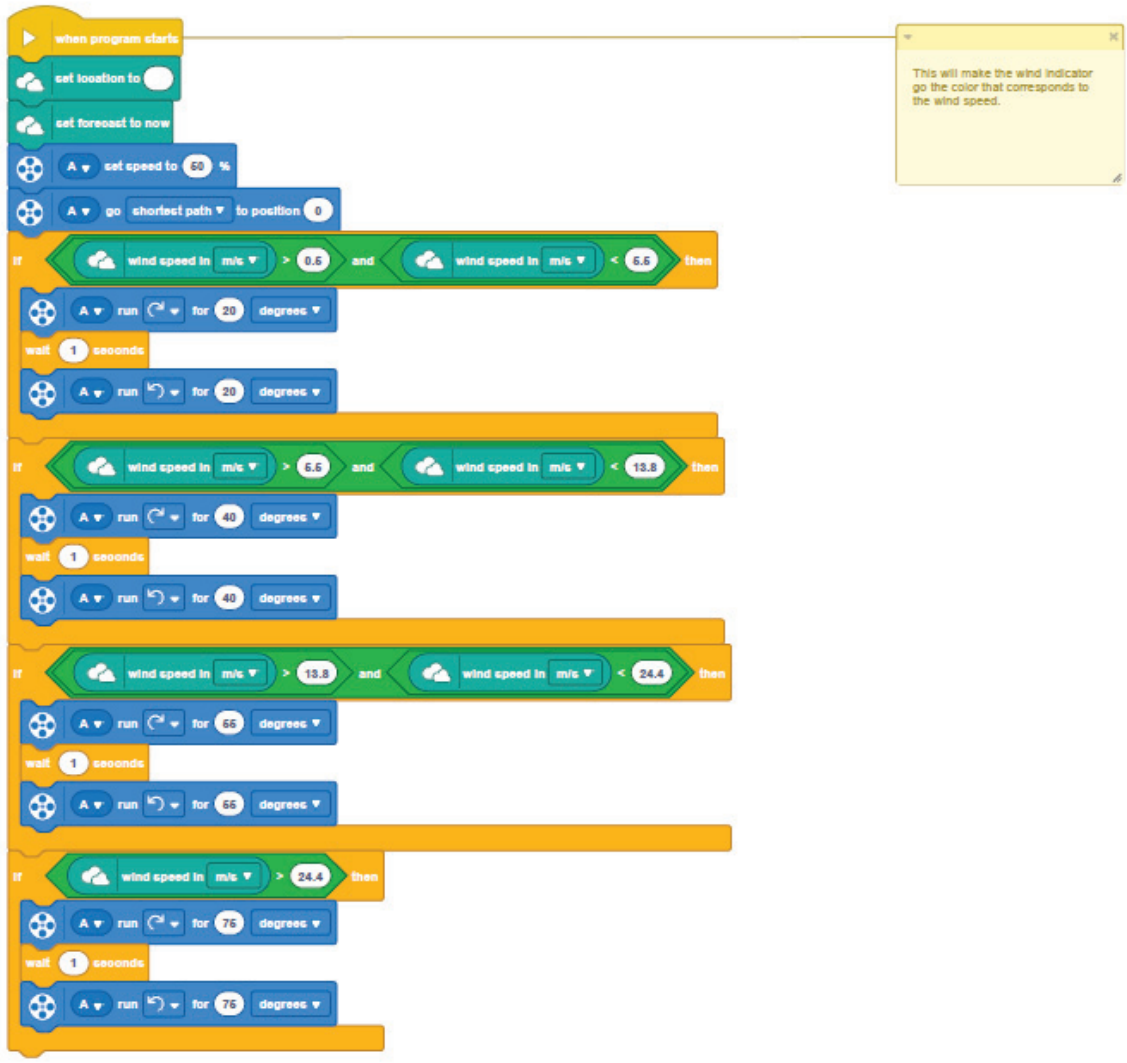
51 pav. „Spike“ roboto programa

Galimas sprendimas:



52 pav. „Spike robot“ programa

Kita programa:



53 pav. „Spike robot“ programa

Diferencijavimas:

Perkelkite šią pamoką į kitą lygį:

- Pakeiskite vėjo indikatorių, kad jis galėtų judėti 180 laipsnių ribose. Stebėkite, kaip greitai jūsų mokiniai gali iš naujo kalibruoti.
- Papašykite savo mokinių sukurti savo vėjo indikatorius.

Apibendrinimas

Vertinimo galimybės. Mokytojo stebėjimo kontrolinis sąrašas: sukurkite skalę, atitinkančią jūsų poreikius, pavyzdžiui:

- Iš dalies atlikta
- Visiškai atlikta
- Daugiau nei reikia

Naudokite šiuos sėkmės kriterijus, kad įvertintumėte savo mokinių pažangą:

- Mokiniai gali naudoti sąlyginius teiginius kalibruoti savo skalę.
- Mokiniai gali kalibruoti savo skalę, nesvarbu, kokius duomenis jie naudoja (duomenų vienetus).
- Mokiniai gali modifikuoti savo programą, kad vienu metu rinktų ir naudotų dviejų skirtingų tipų debesų duomenis (vėjo greitį ir kryptį).

Įsivertinimas. Paprašykite kiekvieno mokinio pasirinkti kaladėlę, kuri, jo nuomone, geriausiai atspindi jų rezultatus.

- Mėlyna: suprogramavau savo modelį naudodamas debesų orų duomenis, kad rodyčiau du skirtingus vėjo greičius iš trijų skirtingų vietų.
- Geltona: užprogramavau savo modelį taip, kad iš trijų skirtingų vietų rodytų keturias skirtingas vėjo greičio kategorijas pagal Boforto skalę.
- Violetinė: pridėjau ir užprogramavau papildomą variklį su ratuku, kad būtų rodoma vėjo kryptis kiekvienoje vietoje.

Bendraamžių vertinimas. Skatinkite savo mokinius teikti atsiliepimus kitiems:

- Leiskite vienam mokiniui įvertinti kito pasirodymą, naudodamas aukščiau esančią spalvotą skalę.
- Paprašykite jų pateikti vieni kitiems konstruktyvių atsiliepimų, kad kitos pamokos metu jie galėtų pagerinti savo grupės veiklą.

Kalbos meno plėtinys. Norėdami įtraukti kalbos meno įgūdžių ugdymą:

- Paprašykite savo mokinių įrašyti ir paskelbti įspėjimus pagal jų vėjo greičio orų prognozes.
- Kaip savo prognozės dalį, paskatinkite juos paaiškinti, kaip veikia vėjas.

Matematikos plėtinys. Kai jūsų mokiniai programuoja variklio kampą, kad parodytų vėjo greitį:

- Paaiškinkite, kad jie interpretuoja teiginius apie santykinę dviejų skaičių padėtį.
- Paaiškinkite, kad jie rašo teiginius apie racionalius skaičius šiame realiame kontekste (pvz., jie rašo „vėjo greitis 13,8 > 24,4 m/s“, kad išreikštų faktą, kad vėjo greitis yra greitesnis). Jie parinks matavimams tinkamo dydžio vienetus ir užprogramuos variklį, kad kampas judėtų proporcingai.
- Paprašykite savo mokinių dirbti su skirtingais vienetais (pvz., mph, kmph, mazgai).

3. „LEGO Education EV3“ roboto konstravimo ir programavimo pavyzdinė pamoka – pavaros

Santrauka

Pavarų dėžės konstravimas, kad ištirtumėte skirtingų pavarų dėžių naudojimo poveikį.

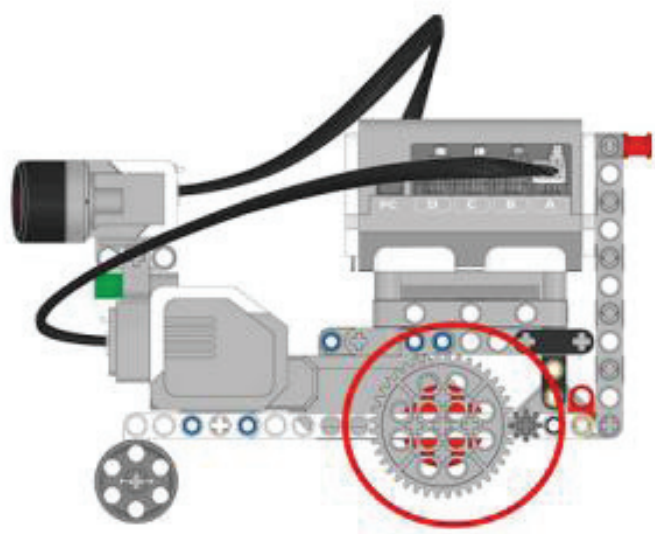
- Trukmė 45–90 min.
- Lygis – vidutinis (6–8 klasės).

Mokymosi tikslai. Mokiniai:

- Išmoks, kokiomis situacijomis reikia įjungti arba sumažinti pavarą;
- Sužinos, kad keičiant pavarų skaičių atsiranda skirtingi greičiai.

Priemonės:

- „LEGO MINDSTORMS Education EV3“ konstruktorius;
- „EV3 Classroom“ programa;
- Konstravimo instrukcijos: https://education.lego.com/v3/assets/blt293eea581807678a/blt0bd58a02e73bf015/5ec7ba64f11f7e3ed6d4d7e9/45.pdf



Įvadas

Pamokos planas:

- Pasiruoškite. Perskaitykite mokinio medžiagą programoje EV3. Surinkite šiek tiek informacijos apie pavarų dėžes, kuri padės jūsų mokiniams suprasti tokias sąvokas kaip pavaros didinimas ir sumažinimas.

- Sudominkite. Pasinaudokite idėjomis, pateiktomis toliau esančioje skiltyje „Pradėti diskusiją“, kad įtrauktumėte mokinius į diskusiją, susijusią su šia pamoka. Suskirstykite mokinius į poras.
- Tyrinėkite. Leiskite kiekvienai mokinių porai sukurti pavarų dėžę. Duokite jiems šiek tiek laiko atlikti bandomąjį paleidimą, kad įsitikintumėte, jog modelis sukurtas tinkamai ir veikia taip, kaip tikėtasi.
- Paaiškinkite, kad kiekviena komanda atliktų eksperimentą bent tris kartus su kiekviena iš siūlomų pavarų dėžių ir užrašykite jų rezultatus. Įsitikinkite, kad jie sukūrė savo bandymų lenteles.
- Išspręskite. Tegul mokiniai išanalizuoja, kaip pavaros santykis yra susijęs su nuvažiuotu atstumu ir transporto priemonės greičiu. Paprašykite kiekvienos komandos trumpai apibendrinti savo eksperimentų rezultatus.
- Įvertinkite. Pateikite atsiliepimų apie kiekvieno mokinio veiklą. Norėdami supaprastinti procesą, galite naudoti pateiktas vertinimo rubrikas.

Pradėkite diskusiją. Gera pavarų sistema yra naudinga norint reguliuoti galią, reikalingą judėti. Važiuodami lyguma naudojate žemą pavarą, o norėdami važiuoti didesniu greičiu, perjungiate didesnę pavarą. Pradėkite diskusiją apie pavarų perjungimą užduodami atitinkamus klausimus, pvz.:

Ką reiškia terminai transmisija ir pavarų mažinimas?

Kodėl reduktoriai dažnai yra uždengti?

Koks ryšys tarp pavaros ir nuvažiuoto atstumo?

Kokiais atvejais pageidautina sumažinti pavarą? Kokiose situacijose taip nėra?

Konstravimas

54 pav. Pavaros

Bandomasis paleidimas. Padėkite transporto priemonę bent 10 cm atstumu nuo sienos, kad ultragarsinis jutiklis būtų statmenas sienai. Paleiskite programą ir palaukite, kol ekrane pasirodys EV3 piktograma. Norėdami pradėti bandomąjį paleidimą, paspauskite EV3 plytelės centrinį mygtuką. Pradinis atstumas iki sienos išmatuojamas naudojant ultragarsinį jutiklį, o laikmatis nustatomas iš naujo. Transporto priemonė nutols nuo sienos vieną apsisukimą, paleidusi didelį variklį. Dar kartą išmatuojamas atstumas iki sienos, o praėjęs laikas išmatuojamas nuvažiuotam atstumui, greičiui ir sukimosi greičiui apskaičiuoti. Apskaičiuotos vertės rodomos ekrane, kol dar kartą paspaudžiamas centrinis mygtukas, kad būtų pasiruošta kitam bandymui.

Ultragarsinio jutiklio naudojimas. Ultragarsinis jutiklis generuoja garso impulsus, kurie sudaro garso kūgį, kuriame galima aptikti objektus. Pradėdami eksperimentą, nestovėkite prieš transporto priemonę arba ultragarsinio jutiklio jutimo kūgio viduje; geriausia vieta yra už ultragarsinio jutiklio.

Eksperimentas. Priminkite savo mokiniams prieš vykdant eksperimentą:

Nuvažiuotas atstumas (cm), greitis (m/s) ir sukimosi greitis (apsukomis per sekundę) bus rodomas ekrane.

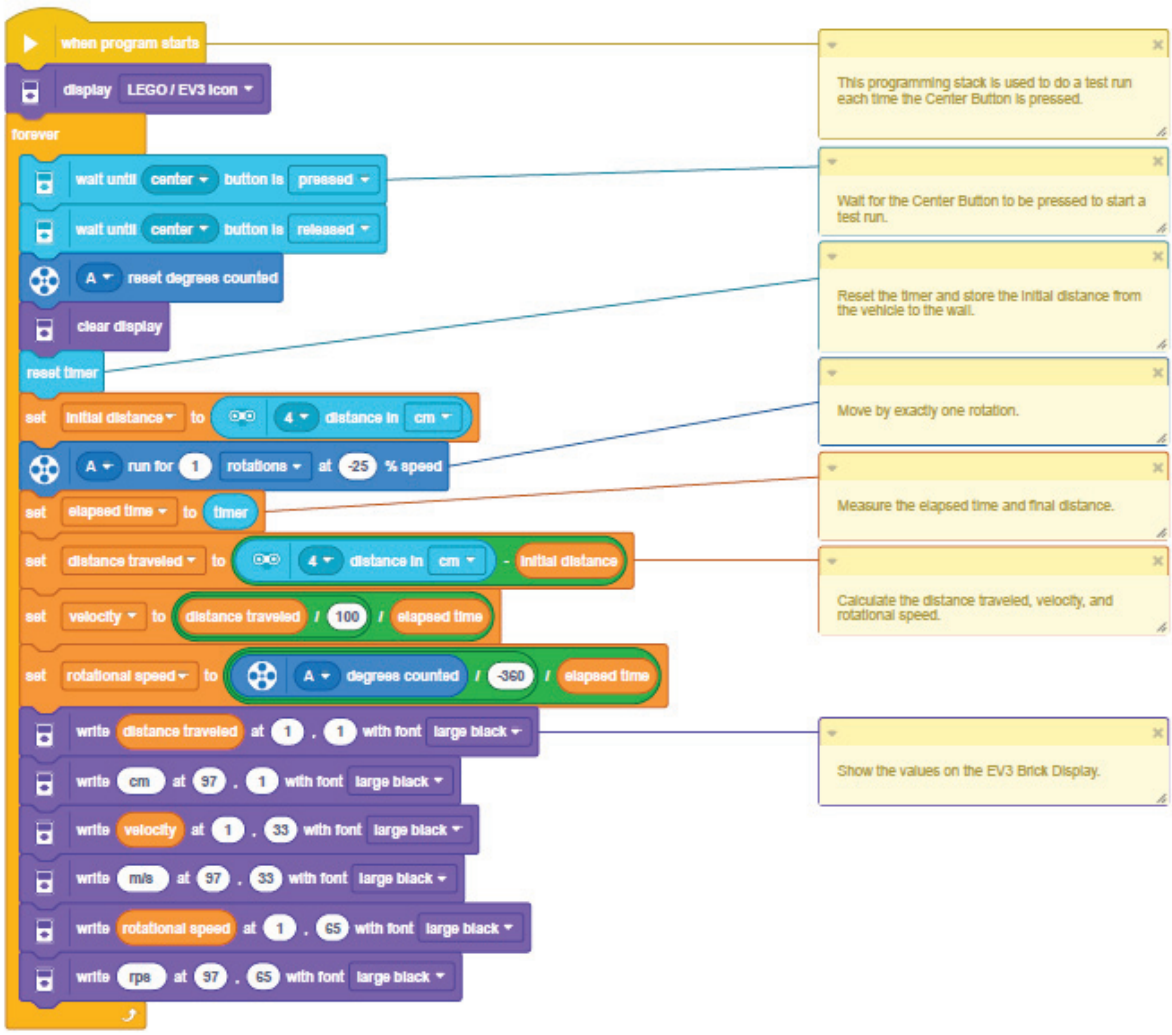
Į bandymų lentelę įrašykite eksperimento numerį, pavaros santykį, savo prognozes, nuvažiuotą atstumą ir greitį. Būtinai palikite pakankamai vietos kitiems stebėjimams įrašyti.

Atlikite eksperimentą bent tris kartus kiekvienai pavarų dėžei ir naudokite vidutines vertes, kad užtikrintumėte patikimiausius rezultatus.

Norėdami sužinoti, kaip pakeisti pavarų dėžes, žiūrėkite paveikslėlius „Užuominoje“.

Programavimas

Programavimo patarimai:



55 pav. „EV3 robot“ programa

Diferencijavimas:

Supaprastinkite šią pamoką:

- Dirbdami su mokiniais, padėkite jiems analizuoti, kaip pavaros santykis yra susijęs su nuvažiuotu atstumu ir transporto priemonės greičiu.
- Sumažinkite pavarų, kurias mokiniai turi ištirti, skaičių.
- Perkelkite šią pamoką į kitą lygį:
- Paaiškinkite vidutinių verčių, pvz., aritmetinio vidurkio ir medianos, jautrumą nuokrypiams, kurie gali būti naudojami matavimo paklaidoms išlyginti eksperimentų serijos metu.
- Paskatinkite mokinius apibrėžti funkciją, leidžiančią numatyti nuvažiuotą atstumo ir pavaros santykį.
- Paprašykite mokinių sugalvoti būdų, kaip pagerinti savo eksperimentų tikslumą.

Apibendrinimas

Vertinimo galimybės. Mokytojo stebėjimo kontrolinis sąrašas: sukurkite skalę, atitinkančią jūsų poreikius, pavyzdžiui:

- Iš dalies atlikta
- Visiškai atlikta
- Daugiau nei reikia

Naudokite šiuos sėkmės kriterijus, kad įvertintumėte savo mokinių pažangą:

- Mokiniai paaiškino pavaros santykio reikšmę, remdamiesi eksperimento rezultatais.
- Mokiniai taikė matematines sąvokas ir (arba) procesus, kad nustatytų ryšį tarp pavarų skaičiaus ir nuvažiuoto atstumo.
- Mokiniai įvertino eksperimentų procedūras ir nustatė nepriklausomus, priklausomus ir kontrolinius kintamuosius.

Įsivertinimas: paprašykite kiekvieno mokinio pasirinkti lygį, kuris, jo nuomone, geriausiai atspindi jo pasiekimus.

- Bronza: Atlikau eksperimentus, bet neaprašiau pavaros santykio vaidmens numatant eksperimento rezultatus.
- Sidabras: Su tam tikra pagalba aprašiau pavaros santykio vaidmenį numatant eksperimento rezultatus.
- Auksas: Eksperimentuose pasinaudojau savo supratimu apie pavaros santykio vaidmenį, kad nuspėčiau, koks perdavimo skaičius turėtų būti naudojamas transporto priemonėje, tinkančioje sunkiems kroviniams vežti.
- Platina: Eksperimente pasinaudojau savo supratimu apie pavaros santykį, kad nuspėčiau, koks perdavimo skaičius turėtų būti naudojamas transporto priemonėje, tinkančioje sunkiems kroviniams vežti. Taip pat numačiau, koks pavarų skaičius turėtų būti naudojamas gaminant greitąjį automobilį.

4. „SumoBoy“ roboto konstravimo ir programavimo pavyzdinė pamoka – programinis variklių greičio reguliavimas

Santrauka

Pamokoje supažindinama, kaip naudoti PWM reguliuoti variklių greitį.

- Trukmė 45–60 min.
- Lygis – paženguisiems (9–12 klasės).

Mokymosi tikslai:

- Išmokti naudoti PWM variklio greičiui reguliuoti.

Priemonės:

- Rezistorius R1 (10 kΩ) – 1 vnt.
- Rezistorius R2 (1 kΩ) – 1 vnt.
- Potenciometras R3 (50 kΩ) – 1 vnt.
- DC variklis (3–6 V) – 1 vnt.
- Diodas IN5819 – 1 vnt.
- NPN tranzistorius BC517 – 1 vnt.
- Tvirtinimo laidas – pagal poreikį.
- Arduino programa *https://www.arduino.cc/*

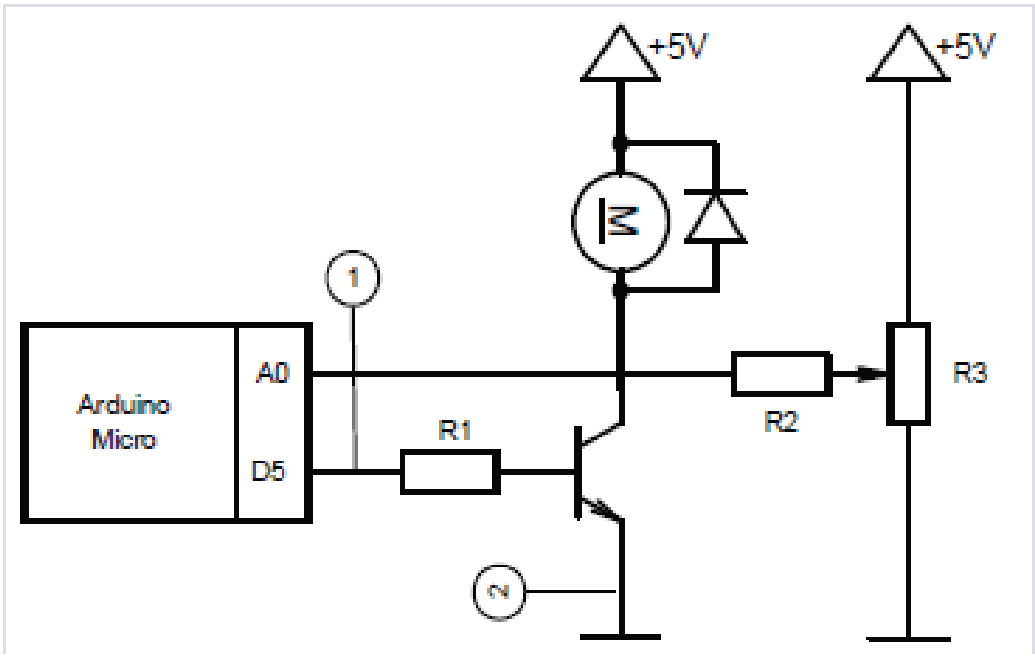
56 pav. „SumoBoy“ robotas

Konstravimas

Darbo eiga:

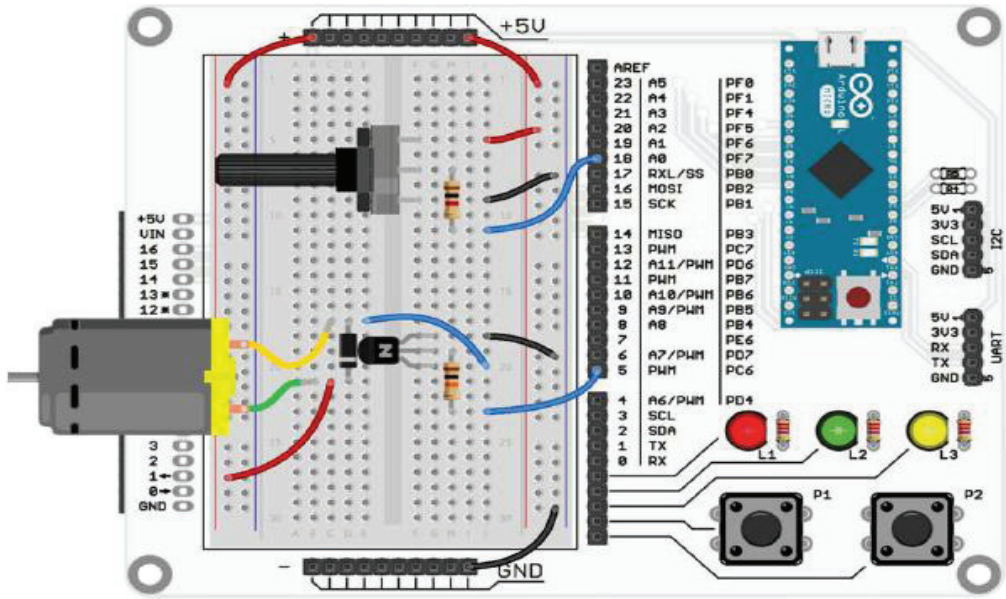
- Sujunkite grandinę, parodytą paveikslėlyje.
- Parašykite pateiktą programą.
- Prijunkite grandinę prie įtampos šaltinio.
- Paleiskite pateiktą programą.
- Pasukite potenciometro svirtį.
- Patikrinkite, ar pasukus svirtį nepasikeičia variklių greitis.
- Atidarykite serijos prievado monitorių: Įrankiai → Serija prievado monitorius.
- Pažiūrėkite, kaip keičiasi variklio greitis priklausomai nuo svirties pasukimo kampo.
- Pasukite svirtį, kad prievado monitorius rodytų apie 200.
- Išmatuokite įtampą tarp 1 ir 2 taškų.
- Pasukite svirtį, kad prievado monitorius rodytų apie 50.
- Išmatuokite įtampą tarp 1 ir 2 taškų.
- Apskaičiuokite, kokia įtampa atitinka kokias reikšmes serijos prievado monitorius.

Kuriama principinė schema:



57 pav. Principinė schema

Komponentų sujungimo paveikslėlis:



58 pav. Komponentų sujungimo schema

Programavimas

Programos pavyzdys:

```
int motPin = 5;  //motor management output
int potPin = A0;
    //potentiometers readings input
void setup ()
{
   pinMode(motPin,OUTPUT);
     //set up motor management leg as an
     //output
     /*analog input reads values from 0 to
     1023, but motors management happens in
     the range of 0 to 255, that is why the
     read value is divided by 4*/
   #define potReading analogRead(potPin)/4
      //definition of potentiometers reading
      //comand
   #define motDrive analogWrite(motPin,
   potReading);
      //definition of managing the motor
   Serial.begin(9600);
   //turn on communication with the computer
}
void loop ()
{
   motDrive;
   //turn on the motor according to the
   //potentiometers readings
     Serial.println(potReading);
   //information about the speed of the motor
}
```

59 pav. „SumoBoy“ roboto programos pavyzdys

Literatūros sąrašas

Harel, I. and Papert S. (Eds.) Constructionism, Ablex Publishing Corporation, Norwood, NJ: 1991

Harel, I. (Ed.) Constructionist Learning, M.I.T. Media Laboratory, Cambridge, MA: 1990

Papert, S. Mindstorms, Basic Books Inc., New York: 1993

Piaget, J. The Essential Piaget, Grubert, H. and Voneche, J. (Eds.), Basic Books Inc., New York: 1997

Alimisis, D., & Kynigos, C. Constructionism and Robotics in Education, 2009

Gary S. Stager, Ph.D., A Constructionist Approach to Teaching with Robotics, Pepperdine University 2010

LEGO Learning Institute, A System for Learning, Published by LEGO® Education 2014

Bruner, Jerome: The culture of education, Harvard University Press, 1996

Czikszentmihalyi, Mihalyi: Flow: The psychology of optimal experience, Harper & Row, 1991

Drotner, Kirsten: 'Leisure is hard work: Digital practices and future competencies', in Buckingham, David (ed.), Building the field of digital media and learning, The MacArthur Foundation Digital Media and Learning Initiative, 2008

Duschl, Richard, Schweingruber, Heidi, and Shouse, Andrew (eds): Taking science to school: Learning and teaching science in grades K-8, The National Academies Press, 2007

Dweck, Carol: Mindset: The new psychology of success, Random House, 2006

Edwards, David: Artscience – Creativity in the post-Google generation, Harvard University Press, 2008

Forman, G, and Fleet, H: Constructive play: Applying Piaget in the preschool, Addison-Wesley Publishing Co., 1984

Gauntlett, David: Creative explorations: New approaches to identities and audiences, Routledge, 2007

Gee, James Paul: 'Learning and Games', in Salen, Katie (ed.), The ecology of games: Connecting youth, games and learning, MIT Press, 2008

Jenkins, Henry: Confronting the challenges of participatory culture: Media education for the 21st century, MIT Press, 2009

OECD: The creative society of the 21st century, 2000

Papert, Seymour: Mindstorms: Children, computers, and powerful ideas, Basic Books, 1980

Pink, Daniel: A whole new mind, Marshall Cavendish, 2008

Rogers, Carl (1967): 'The interpersonal relationship in the facilitation of learning', in Kirschenbaum, H. and Henderson, V. L. (eds), The Carl Rogers Reader, Houghton Miflin, 1989

ROBONEST leidinys: Learn the basics of Robotics "SumoBoy" v 2.0

Vygotskij, Lev S: Mind in society: The development of higher psychological processes, (eds) Michael Cole et al. [transl. from Russian], Harvard University Press, 1978

Robotikos vadovėlis popamokinėms veikloms

Parengė
Artūras Gaulia

2022-10-18
Apimtis 6 apsk. sp. l. Užs. Nr. 2335.
Tiražas 400 egz.
Išleido ir spausdino UAB „Utenos Indra“
Maironio g. 12, LT-28143 Utena
www.indra.lt