TeaMe

TTÜ informaatikainstituut

Objektorienteeritud modelleerimine

Objektorienteeritud (OO) lähenemisviis domineerib tarkvara, rakenduste ja infosüsteemide loomisel, arendamisel ja kasutamisel. Selle olemus seisneb reaalsete ja abstraktsete süsteemide ja objektide oleku ja käitumise modelleerimises arvutil tarkvara-objektide abil.

Sisu

Objektorienteeritud lähenemisviisi olemus ja põhimõisted

Objektorienteeritud (OO) lähenemisviis domineerib tarkvara, rakenduste ja infosüsteemide loomisel, arendamisel ja kasutamisel. Selle olemus seisneb reaalsete ja abstraktsete süsteemide ja objektide oleku ja käitumise modelleerimises arvutil **tarkvaraobjektide** abil.

Tarkvaraobjekt on omavahel seotud andmete ja programmide (protseduuride) kogum. Ka arvuti süsteemi- ja rakendustarkvara põhineb tarkvaraobjektidel, milleks on dokumendid, vormid, aknad, tööriistaribad, ohjurid jmt. Uuemad programmeerimiskeeled on objektorienteeritud ning võimaldavad kasutada ja luua tarkvaraobjekte.

OO-lähenemisviisi põhimõisted on järgmised: objekt, klass, objekti omadused ja tegevused, seosed objektide vahel, sündmused**.**

Objekt on piiritletud osa süsteemist ning teda võib vaadelda omaette tervikuna, milleks on **konkreetne** ese, seade, isik, hoone, asutus, fail, dokument, graafiline kujund jms. Objektid on näiteks Juku jalgratas, Pille jalgratas, Peetri arvuti, Juku Naaskel, Juku isa suvila, Juku pall, TTÜ, must kolmnurk, konkreetne dokument …

Klass on ühetüübiliste objektide hulk – abstraktsioon ehk üldmõiste:
jalgratas, arvuti, õpilane, suvila, hoone, ülikool, Excel'i tööleht, Word'i dokument ...

Konkreetne objekt on klassi eksemplar (ilming)

Klassi Jalgratas eksemplarid

Klassi Hulknurk eksemplarid

Omadused (atribuudid) on karakteristikud, mis identifitseerivad objekti ja iseloomustavad selle olekut, väljanägemist jms. Kasutatavate omaduste valik sõltub rakenduse liigist, eesmärkidest jms. Konkreetse ratta atribuudid: liik, mark, mõõtmed, värv, käikude arv, olek, ... Tarkvaraobjekti **ratas** atribuudid müügisüsteemis: samad, mis eelmisel rattal, lisaks veel näiteks tarnija, kauplus jms; arvutisimulatsioonis on atribuutideks (pildi) suurus, kiirus, asukoht jms, graafikaobjektil: laius, kõrgus, asukoht ekraanil, täitevärv, pindala, ümbermõõt …

Tegevused (meetodid, operatsioonid), mida täidab objekt ise või täidetakse objektiga. Valik sõltub süsteemi liigist, eesmärkidest jms. Rataste müük: lisamine, eemaldamine, andmete muutmine jms. Simulatsioon: kiiruse suurendamine, pidurdamine, peatamine jms. Graafikaobjekt: teisaldamine, kopeerimine, mõõtmete muutmine, täitevärvi muutmine jms.

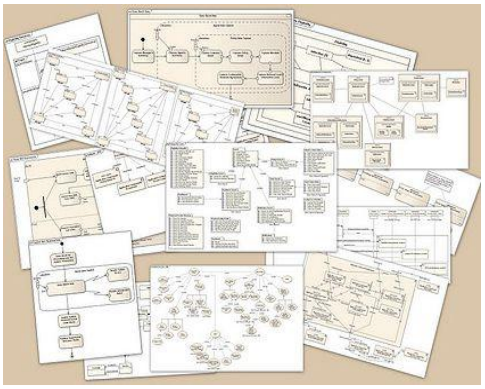
Sündmused. Objekt võib reageerida teatud välistele mõjudele: muudab olekut, käivitab teatud tegevuse jms. Konkreetne ratas: kokkupõrge, ... Simulatsioon: vajutus tühikule – alusta sõitu, hiireklõps – peatu jms.

Seosed (suhted, relatsioonid). Objektil võivad olla seosed teiste objektidega. Seosed on erineva olemuse ja tähendusega:

- omandisuhe: jalgratas kuulub Jukule, arvuti kuulub Peetrile
- töösuhe: Peeter Kask töötab SEB pangas,
- sisaldavuse suhe: protsessor on arvutis, tööleht leht kuulub töövihikusse

Unifitseeritud modelleerimiskeel UML

Objektorienteeritud graafiline keel on süsteemide visuaalseks analüüsiks, kavandamiseks, loomiseks ja dokumenteerimiseks. Keeles on kindel valik lihtsad graafilisi kujutisi (sümboleid) infosüsteemide ja



rakenduste erinevate komponentide, olemite ja tegevuste tähistamiseks ning elementide ühendamiseks. Sümbolite sisse paigutatakse kindla tähenduse ja struktuuriga tekstid. Loodavate diagrammide abil saab kirjeldada süsteemi struktuuri, olekuid, tegevusi jms ning on suhtlemisvahendiks informaatikute ja erinevate erialade spetsialistide vahel.

Kasutatavate diagrammide tüübid ning nende detailiseerimise aste sõltuvad süsteemi iseloomust ja arendusprotsessi faasist: ülesande püstitus, analüüs, disain, ...

Süsteemide struktuuri, oleku ja käitumise kirjeldamiseks on erinevat tüüpi **diagramme** (skeeme ehk mudeleid):

- klassi- ja objektidiagrammid,
- tegevusdiagrammid,
- kasutusjuhtude diagrammid,
- olekudiagrammid,
- …

Peamised mitteinformaatikute jaoks ülesande püstituse, analüüsi ja disaini faasis on **klassi**- ja **tegevusdiagrammid**.

Rahvusvaheline standard (ISO, OMG)

Klassid ja objektid

UML'i diagrammidel klass esitatakse ristkülikuna. Üldjuhul kolm sektsiooni:

- nimi,
- atribuudid (omadused)
- operatsioonid (tegevused, meetodid)

Kohustuslik on ainult nimi.

Näites on toodud kooli infosüsteemi klassi **Õpilane** lihtsustatud kirjeldus.

kood eesnimi perenimi sünniaeg aadress
lisamine() emialdamine() muutmine()

Klassi nimi. Nimi peab algama suurtähega

Atribuudid (**omadused** ehk **parameetrid**)

Igal atribuudil **nimi**. Võib olla näidatud ka **tüüp**: arv, tekst, kuupäev …
Konkreetsete objektide on omadusel kindel väärtus:
K20567, Juku, Naaskel, 26.10.2001, Spordi 5-9, 4B, …
K20573, Pille, Pilt, 23.06.2001, Teatri 7-13, 4A, …

Operatsioonid (**tegevused** ehk **meetodid**)
Esitatakse tavaliselt kujul **nimi()**.
Siin meetodite tähendus on järgmine. **lisamine()** – uue õpilase lisamine, **eemaldamine()** – õpilase eemaldamine, **muutmine()** – õpilase andmete muutmine.

Allpool on veel ühe klassi kirjelduse näide: klass **Arvuti** müügisüsteemis.

Arvuti
mark protsessor taktsagedus sisemälu_maht kõvaketta_maht hind
lisamine() emialdamine() muutmine()

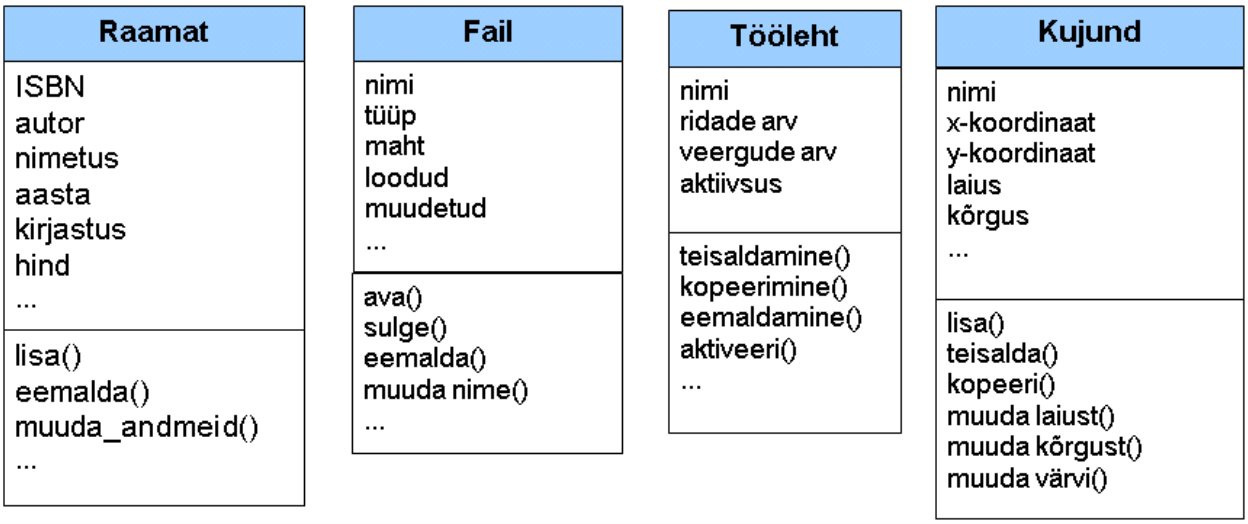
NB! Objektiks ei ole üksik arvuti, vaid nö kaubagrupp: ühesuguste parameetritega arvutid. Igal arvutil antud grupis on sama nimi.

Üheks omaduseks on olemasoleva arvutite arv antud grupis. Tegemist on nn muutuva ehk arvutatava atribuudi. Sama marki arvutite lisamisel või eemaldamisel, muudetakse omadust arv. Arvutatava omaduse tunnus on kaldkriips (/) omaduse nimetuse ees.

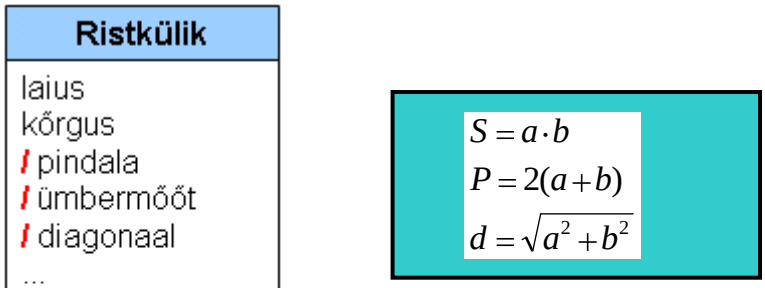
Allpool on toodud näiteks tabel, mis sisaldab andmeid arvutite kohta: arvutite omadusi

Mark	Protsessori tüüp	Taktsagedus	Sisemälu, MB	Kõvaketas, GB	Hind, €	Arv
Aragorn	AMD Athlon64	3,0	512	160	416	13
Balrog	Intel Celeron	2,8	256	80	349	21
Eldar	Intel Xeon	3,0	512	160	388	7
Elend	Intel Celeron	4,0	2048	400	1 083	0
Faram	AMD Athlon64	3,0	512	210	499	12
Frodo	AMD Athlon64	3,0	4072	500	1 108	25
Gandalf	Intel Xeon	3,2	512	160	708	2
Kalvar	Intel Xeon	3,0	512	160	583	7
Marvin	Intel Celeron	3,0	512	160	499	0
Rohan	AMD Athlon64	3,5	512	160	491	0
Sauron	Intel Xeon	2,5	256	180	283	5

Allpool on veel mõned klasside kirjeldamise näited:



Klass Ristkülik süsteemis Geomeetria



$$S = a \cdot b$$
$$P = 2(a+b)$$
$$d = \sqrt{a^2 + b^2}$$

NB! Sisestatavad andmed: laius (a) ja kõrgus (b) ning **tuletatavad** omadused: pindala (S), ümbermõõt (P) ja diagonaal (d), millest viimaste ees võib olla **/** (kaldkriips)

Klass Sprait Scratchis

Universaalne baasklass – superklass. Selle alusel saab luua alamklasse. Projekti loomise käigus, kasutades olemasolevaid spraite ja teisi objekte ning nende omadusi ja meetodeid, luuakse praktiliselt uued objektid. Nende nn tuletatud objektide omadused põhinevad baasobjektide (eeskätt spraitide) omadustel. Spraitide jaoks loodavad skriptid, milleks kasutatakse spraitide standardmeetodeid (käsuplokke) ja juhtimiskäske, kujutavad sisuliselt nende uusi meetodeid. Vt Scratchi põhiobjektid.

Sprait
nimi asukoht: X, Y suund suurus värvus nähtavus
liigu() pööra() muudaX() muudaY() vaheta kostüümi() muuda värvi() ütle() mängi heli()

Klassisümbolite erinevad esitusviisid

Antud juhul võib kasutada erinevaid variante: tühjad sektsioonid, ärajäetud sektsioonid jms.

Sageli kasutataksegi ainult nime ja seda eriti diagrammide juures.

Auto
number mark mudel aasta ...
lisa() muuda() eemalda() ...

Auto
number mark mudel aasta ...

Auto

Auto
lisa() muuda() eemalda() ...

Auto

Seosed ja klassidiagrammid

Süsteemi kuuluvatel objektidel on omavahel teatud seosed. Seos (relatsioon/suhe) näidatakse diagrammidel joone abil. Klassidiagramm näitab klasse ja võimalikke seoseid nende objektide vahel. Reeglina igas seoses on ainult kaks klassi. Seosele võib anda nime, mille võib näidata skeemil. Seose mõlema otsa juures võib esitada korduse. Näitab mitu antud klassi objekti võib olla seotud teise klassi objektiga. Korduste tüüpvariandid ja nende esitus UML'is on järgmised:

n – kindel arv, näit 1, 4, 256

n1, n2, n3, ... – loend, näit.: 2, 4, 6

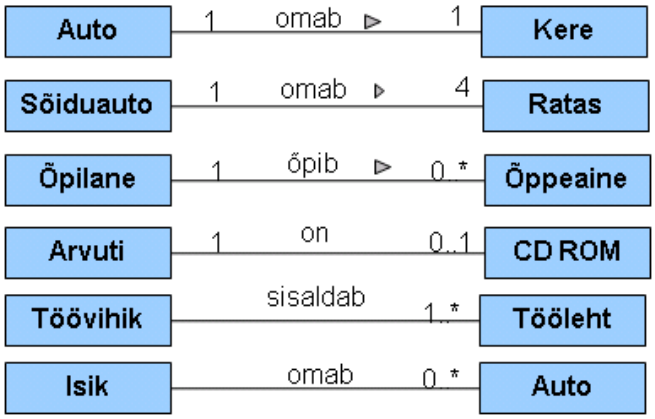
0..1 või 0, 1 – null või üks

* – suvaline hulk

1..* – üks või enam

0..* – null või enam

Seose nime ja korduse väärtuse 1 võib jätta ära



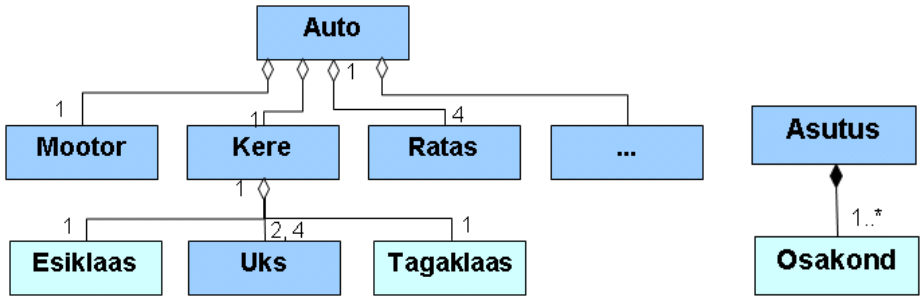
Eristatakse järgmisi seoste liike:

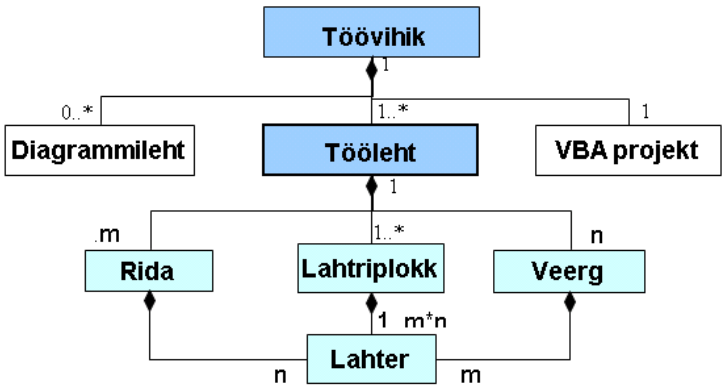
- agregatsioon
- assosatsioon,
- üldistus

Agregatsioon ja kompositsioon

Agregatsioon ehk sisalduvuse seos. Üks objekt sisaldab teisi. Tervik ja osad. Terviku poolel võib olla romb ◊, kuid ei pea tingimata olema.

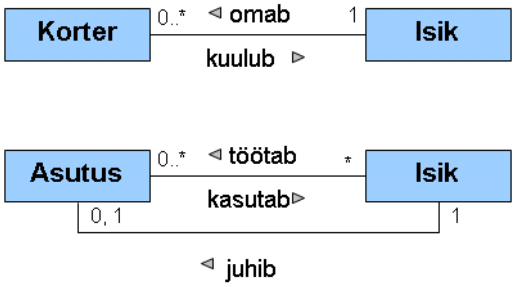
Kompositsioon – agregatsiooni erijuht – osa saab eksisteerida ainult tervikus. Tähistus – täidetud romb ♦





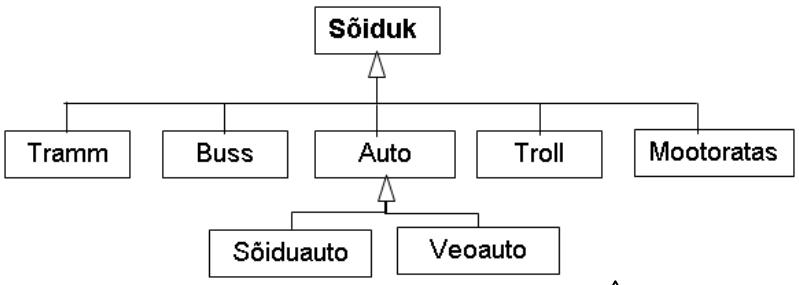
Assosatsioon

Assosatsioon – suvaline mõtet omav seos (suhe) objektide vahel – omandisuhe, töösuhe jms. on Assosatsiooni erijuht on agregatsioon.



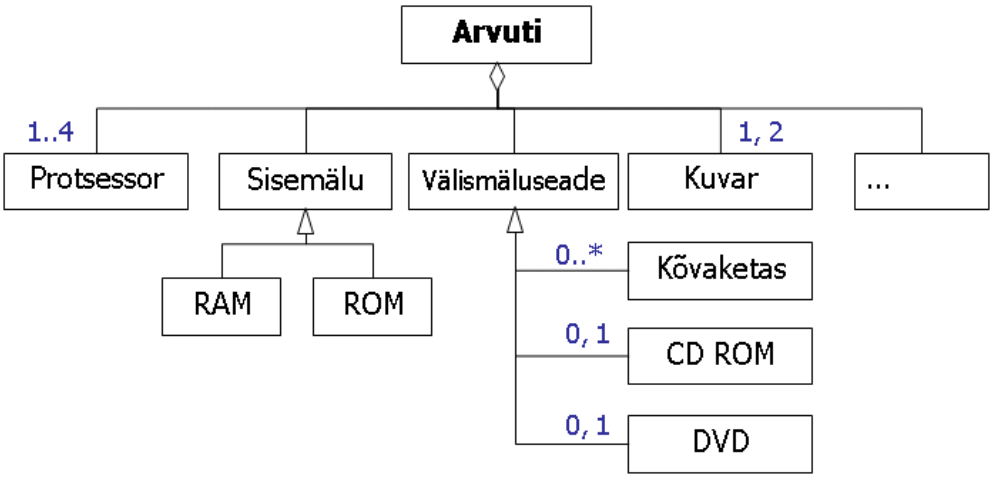
Üldistus (*generalization*)

Üldistus – esivanema (superklassi) ja järglaste (alamklasside) seos. Järglased pärivad esivanema omadused

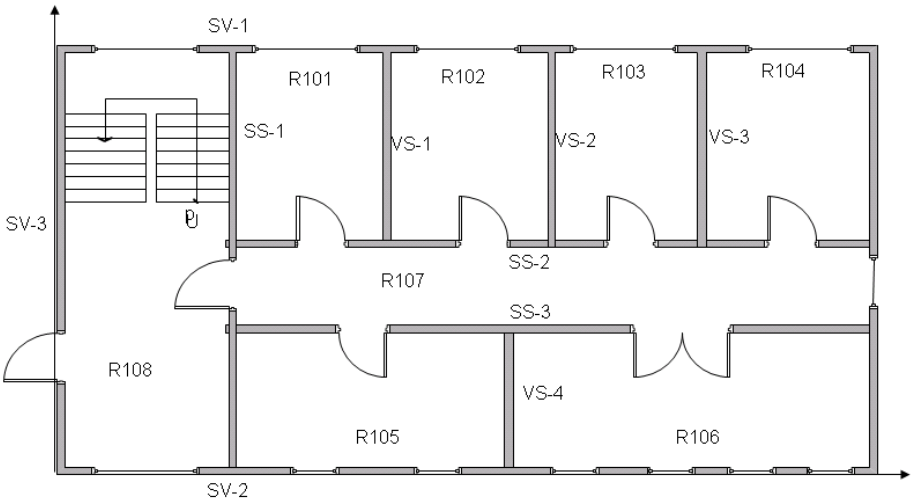


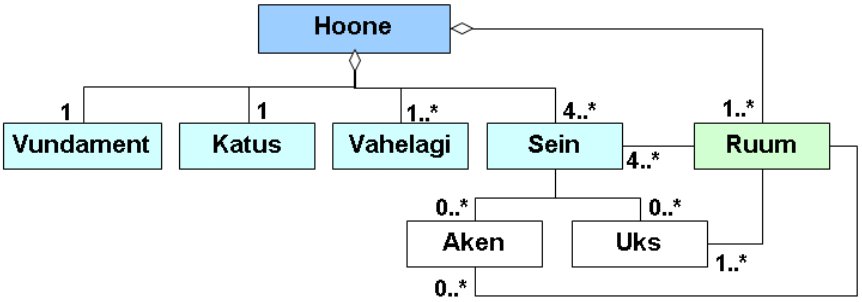
ning võivad omada täiendavaid omadusi. Kujund △ – esivanema poolel.

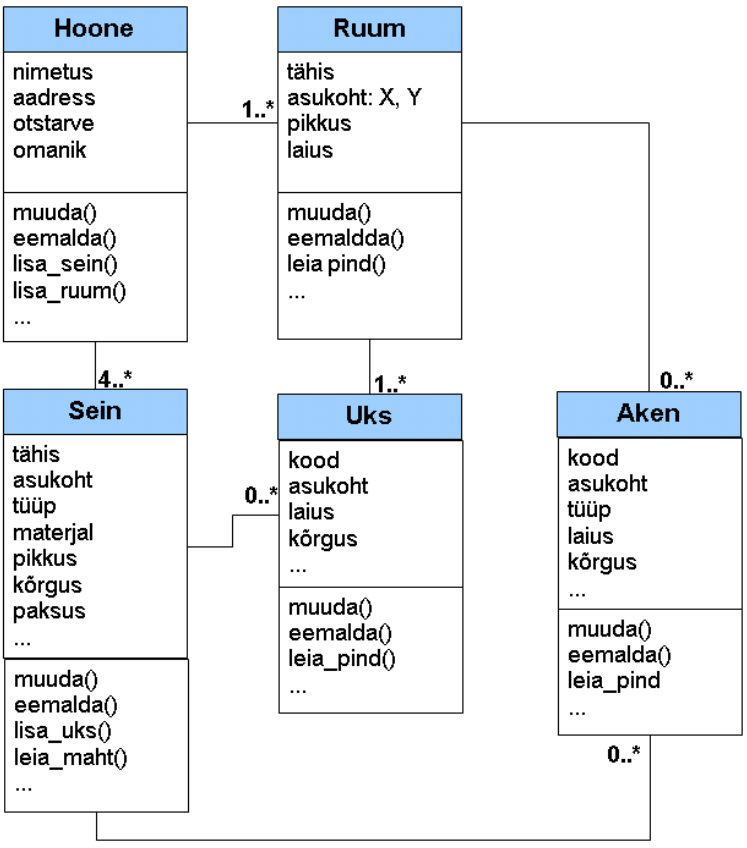
Veel mõned klassidiagrammide näited:



Hoone põhiklassid



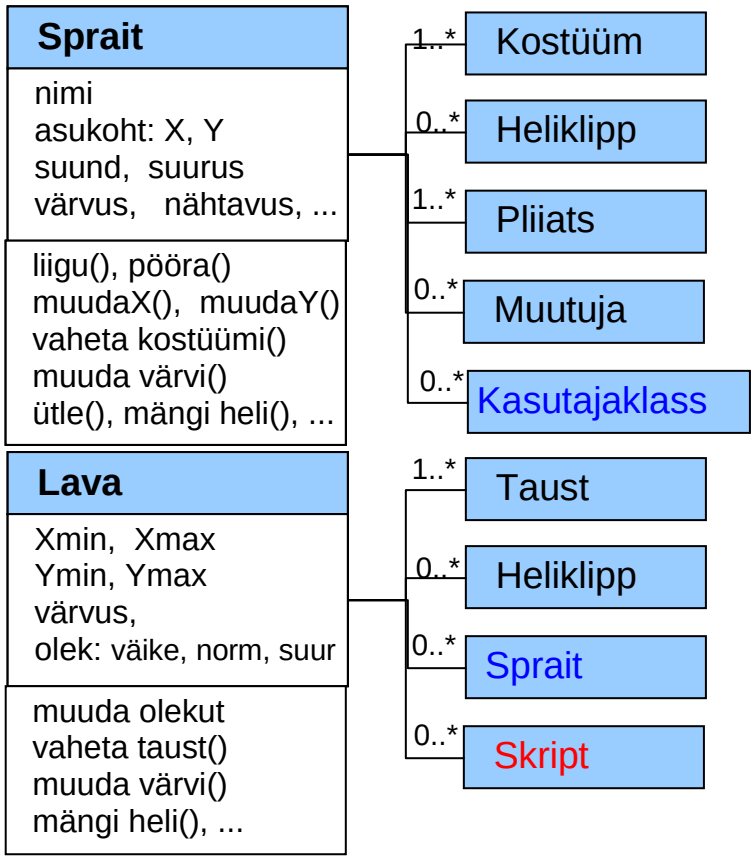


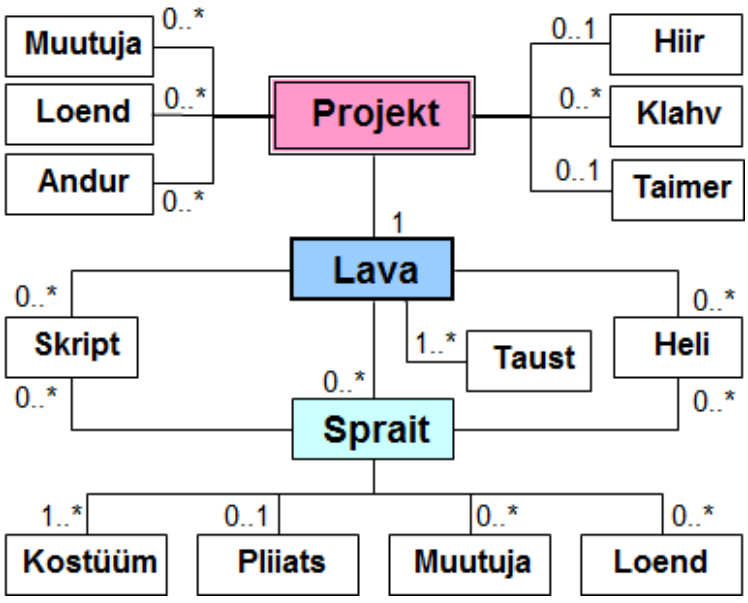


Objektid ja klassid süsteemis Scratch

Universaalne graafikaobjektide klass on **sprait** (***sprite***). Spraitidel on kindel valik **omadusi**: nimi, asukoht, suurus, jms. Spraitidega saab määrata erinevaid tegevusi: liigutada, pöörata, muuta suurust jms. Seda saab teha kindlate **käskude** ehk **plokkide** (**meetodite**) abil. Käskudest saab moodustada programmiüksusi – **skripte.** Klassil sprait on olemas mõned **alamklassid** (kostüüm, pliiats). Saab luua ka uusi alamklasse. **Lava** on ekraani piirkond, kus toimuvad spraitide tegevused. Vt ka Scratchi põhiobjektid

Klass **Sprait** ja selle alamklassid

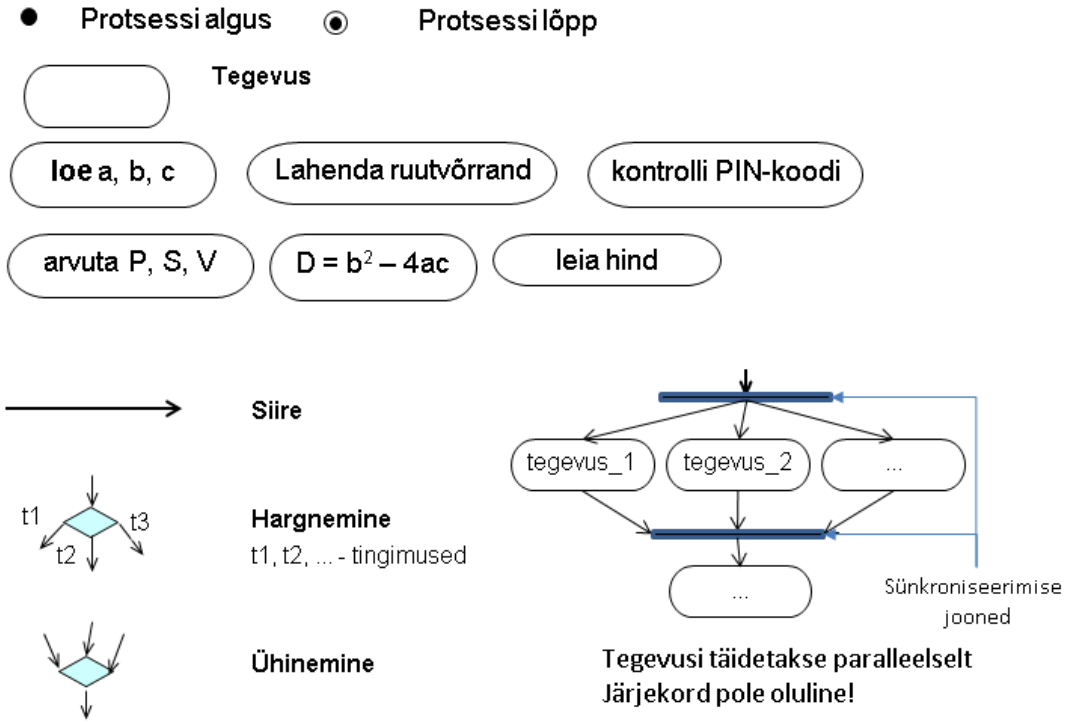




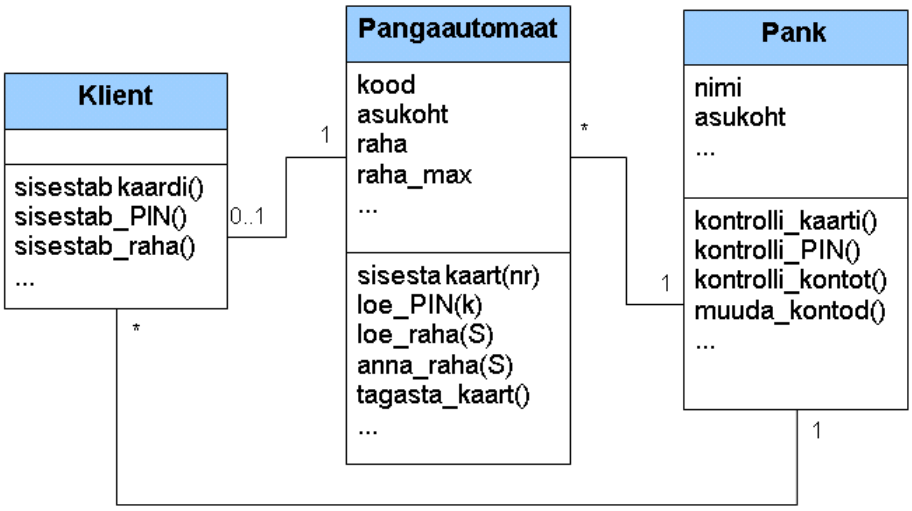
UMLi tegevusskeemid

Tegevusskeem ehk tegevusdiagramm (*Activity Diagram*) esitab graafiliselt antud protsessi või töö täitmiseks vajalikud tegevused ja nende järjekorra. Kasutatakse äri- ja tööprotsesside ning algoritmide kirjeldamiseks

Standardkujundid (sümbolid)

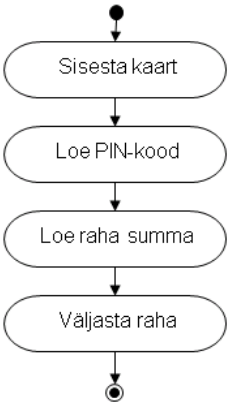


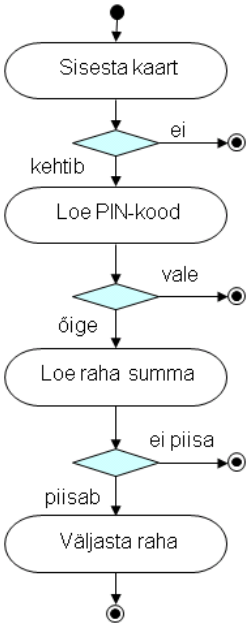
Pangaautomaatide süsteem

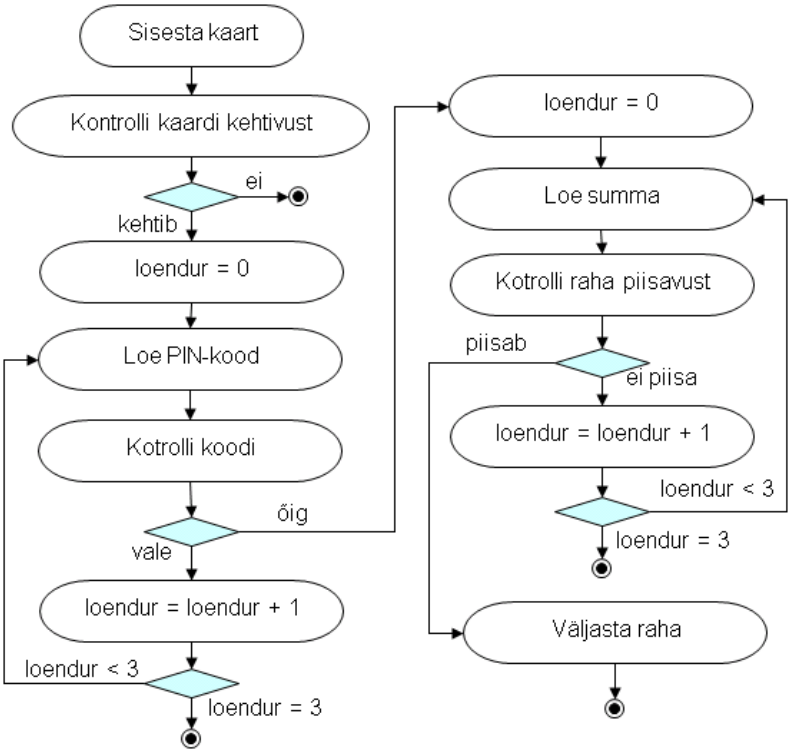


Pangaautomaat

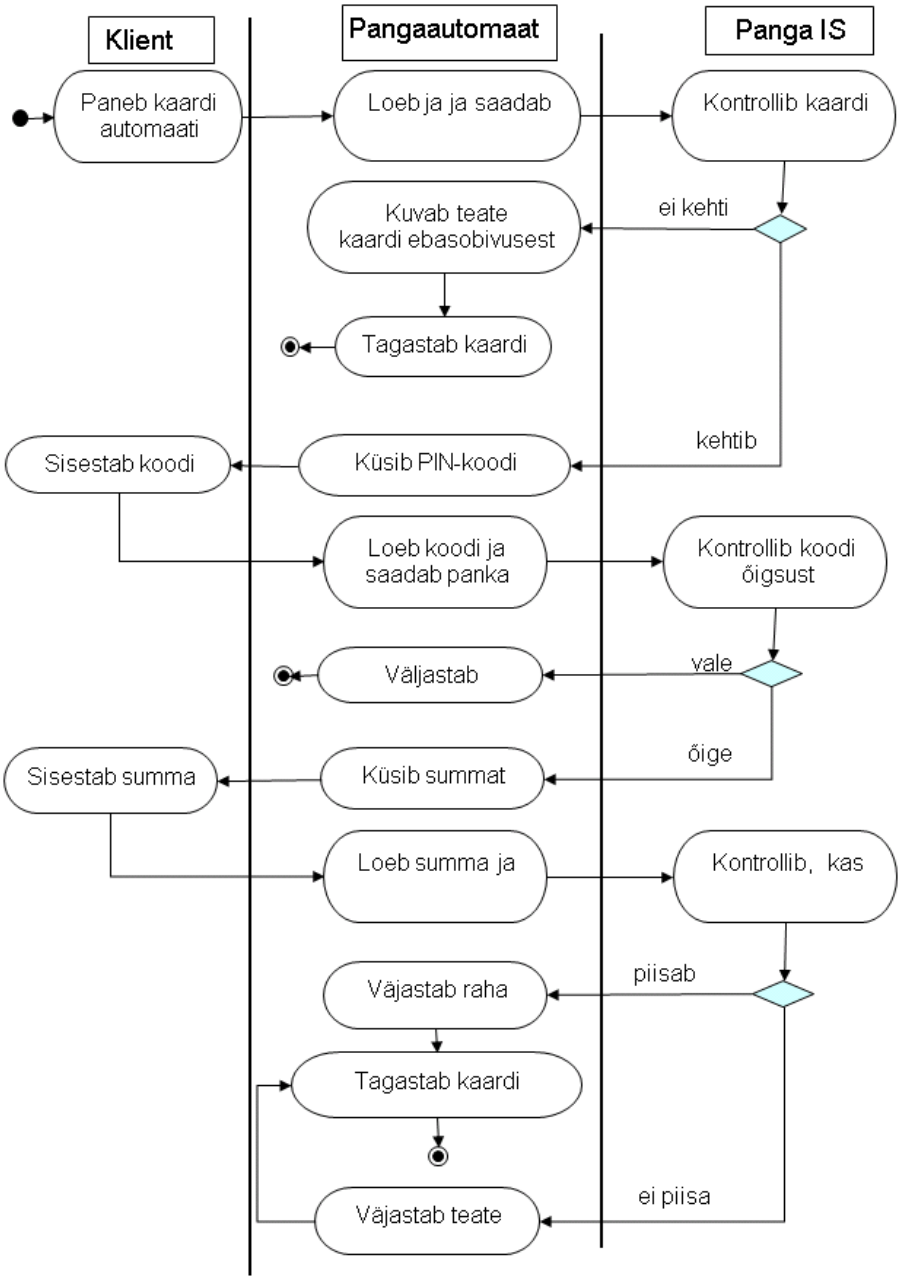
Klass ja erineva täpsusega tegevusskeemid





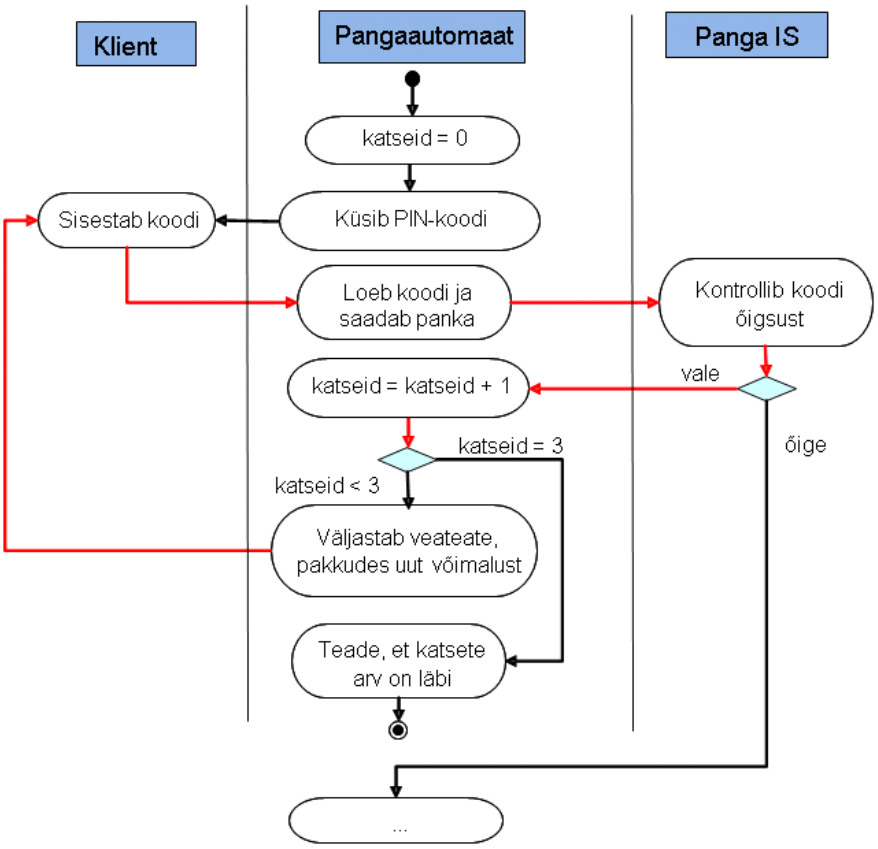


Tegevuste jaotus objektide vahel



Pangaautomaat. PIN koodi kontroll

Lubatud on kolm katset

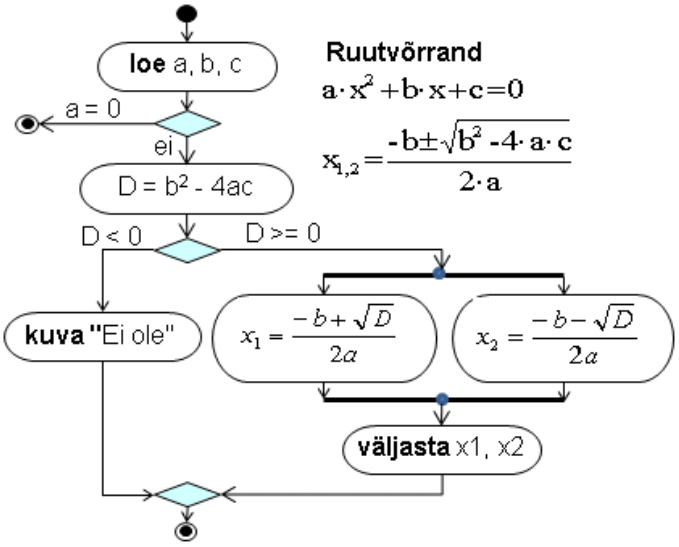


Tegevusskeemid ja algoritmid

UML'i tegevusskeeme sageli kasutatakse algoritmide esitamiseks. Allpool on toodud mõned näited

Ruutvõrrandi lahendamine

Ruutvõrrandi reaalarvuliste juurte leidmine



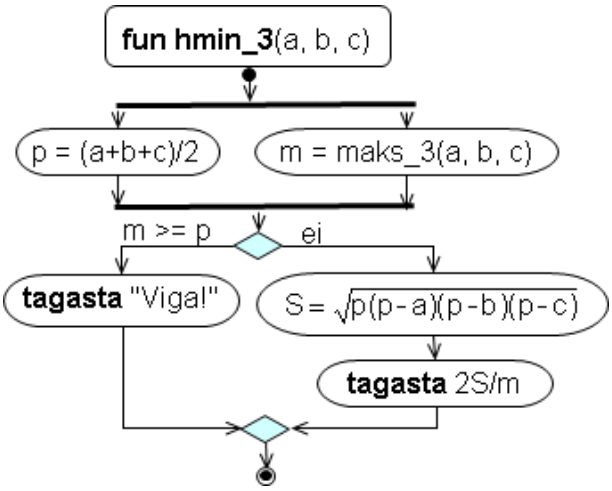

```
loe a, b, c
kui a = 0 siis
   kuva „a ei tohi olla 0!“
   stopp
lõpp kui
D = b2 – 4ac
kui D < 0 siis
   kuva „Lahendeid ei ole!
muidu
   x1 = (-b + sqrt(D)) / (2a)
   x2 = (-b - sqrt(D)) / (2a)
   kuva x1, x2
lõpp kui
```

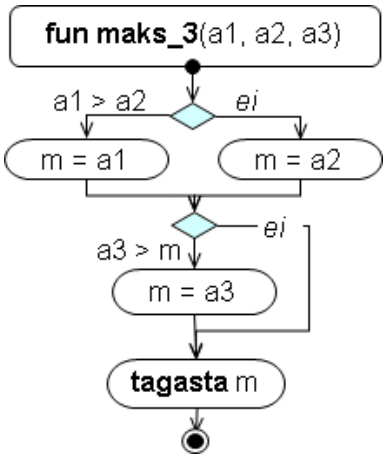
Kolmnurga minimaalne kõrgus

Algoritm kolmnurga minimaalse kõrguse h_{min} leidmiseks, külgede pikkuste a_1, a_2, a_3 alusel.

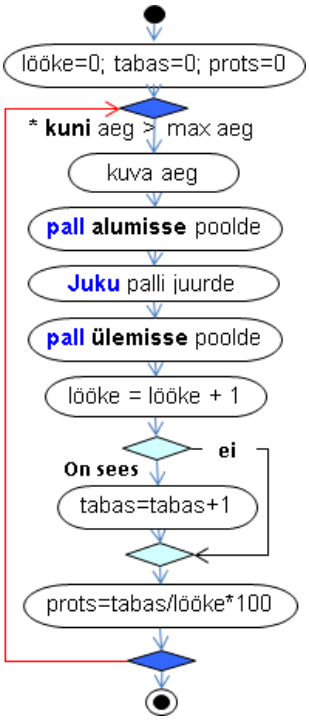
Minimaalne kõrgus: $h_{\min} = 2S / a_{\max}$, kus S on kolmnurga pindala, a_{max} on maksimaalne külg.

Kasutusel on kaks protseduuri (funktsiooni). Funktsioon **maks_3** leiab kolmest arvust suurima. Funktsioon **hnin_3** leiab minimaalse kõrguse, kasutades funktsiooni **maks_3**. Funktsioon kontrollib kolmnurga tingimust. Kui kolmurka moodustada ei saa, tagastab teate „Viga“.

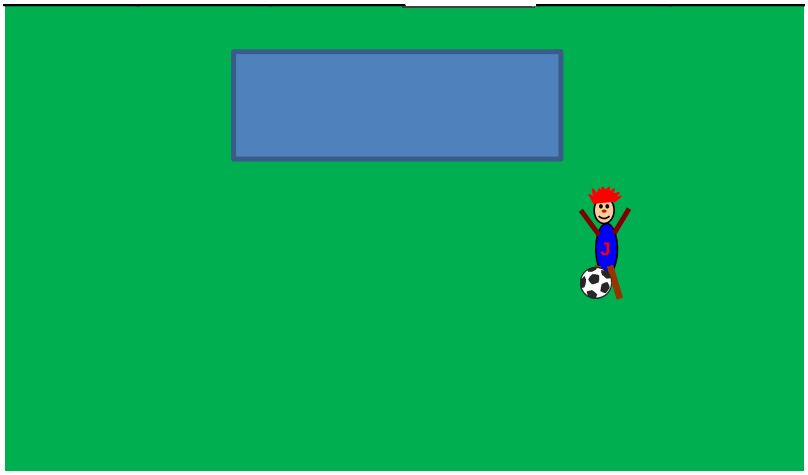




Jalka

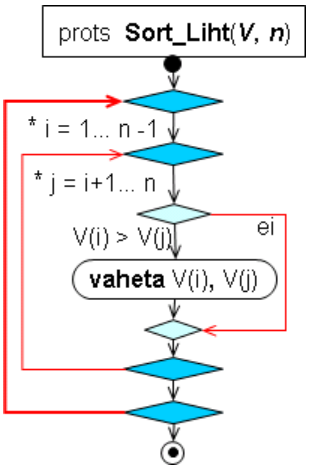


Juku harjutab jalgpalli. Palli asukohta muudetakse juhuslike arvude abil. Tegevusi korratakse kuni aeg saab suuremaks mängu ajast. Pall viiakse platsi alumisse poolde ja Juku liigub palli juurde. Pall viiakse ülemisse poolde, imiteerides lööki. Kui pall sattub väravasse, suurendatakse tabamuste arvu. Pidevalt leitakse protsent.

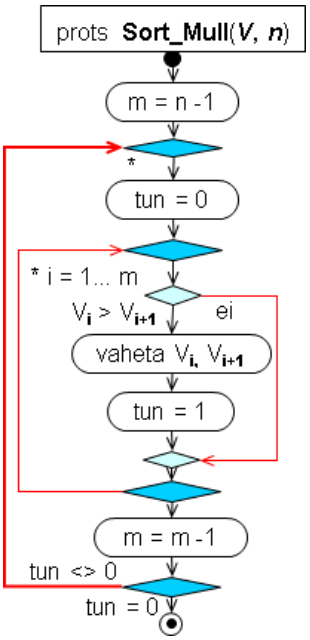


Sortimised

Lihtsortimine



Mullsortimine



Valiksortimine