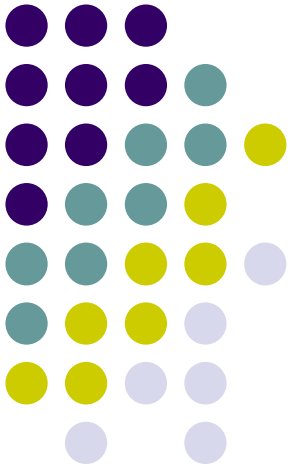
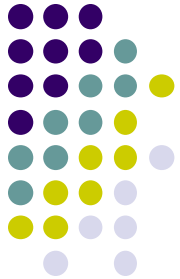
LOOGILINE PROGRAMMEERIMINE (*logic programming*)

Õppejõud: Jüri Vain

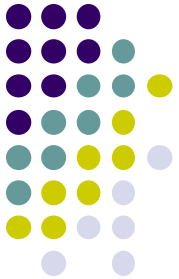


Kursusest üldiselt



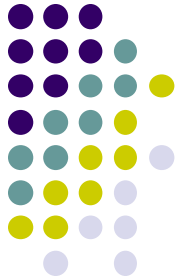
- Kood: ITI0211 6.0 4 2-2-0 H S
  - https://courses.cs.ttu.ee/pages/Loogiline_programmeerimine
  - Moodle
- Kontakt:
  - Konsultatsioon: N kl 16:00 – 17:00 (eelnevalt teatada)
  - E-post: juri.vain@ttu.ee
  - Telefon: 6204190
  - Ruum: ICT-418

Kursuse korraldus



- Loeng – prof. Jüri Vain
  - T 12:00-13:30 (NRG-131)
- Praktikum - Evelin Halling
  - E 08:00-09:30 (ICT-401)

Hindelise arvestuse nõuded



Praktikumi ülesannete (10 tk) esitamine (tähtajaliselt) – max 40%

+

Test 1 – loengute 1 – 6 materjalile – 25 %

+

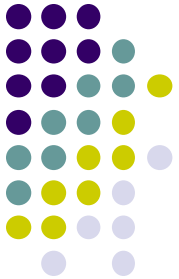
Test 2 – loengute 7– 12 materjalile – 25 %

+

Kodutöö: kabeprogramm (10%) + turniir

NB! Turniiri kohad I-III annavad koondhindele lisapalle vastavalt 3, 2 ja 1 palli.

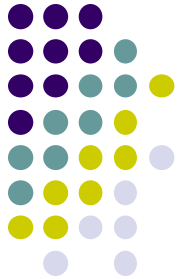
Õppeaine eesmärgid



- Teadmiste formaliseeritud esitamine *Horni lausetega*
- Järelduste tuletamine teadmusbaasist kasutades *resolutsiooni* meetodit ja *unifitseerimisreegleid*
- Põhilised otsingumeetodid ja nende programmeerimine keeles *Prolog*
- Tehisintellekti ülesannete lahendamine:
  - kitsendustega tegevuste/logistika/… planeerimine,
  - keerdülesannete (male, kabe, bridž,…) lahendamine
  - inimene-masin dialoogi süsteemid jne
- *Praktiline programmeerimisoskus*

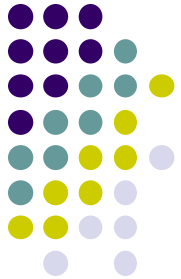
1. Sissejuhatus

1.1. Mis on loogiline programmeerimine?



Programmeerimise paradigma	Fookus:
Imperatiivne	KUIDAS ARVUTADA
OOP	KUIDAS STRUKTUREERIDA PROGRAMMI
Aspekt-orienteeritud	
Struktuurprogrammeerimine	
Loogiline (LP)	MIDA ARVUTADA (kuidas (spetsifitseerida probleemi)
Funktsionaalne (FP)	

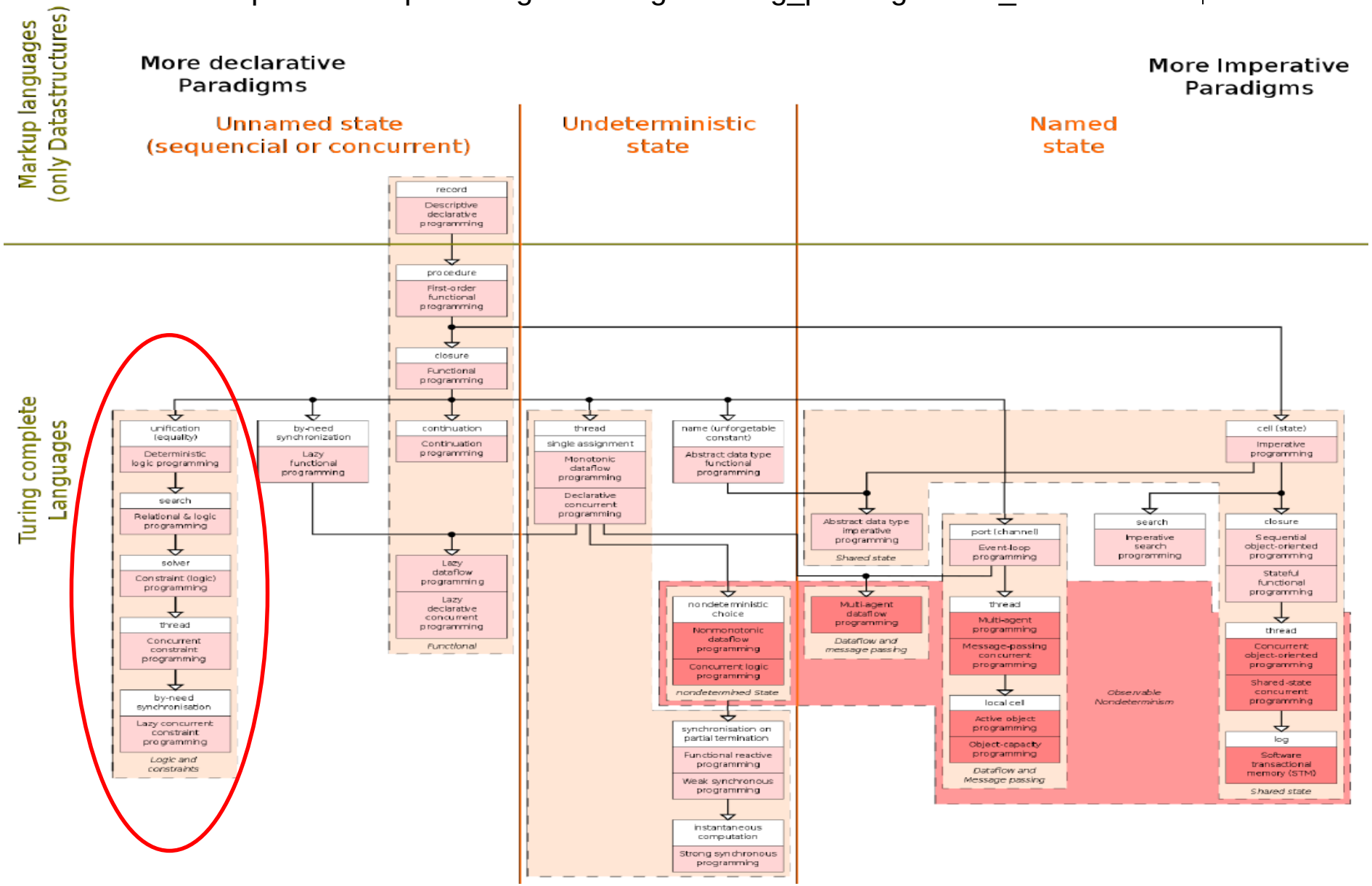
LP aspektid



- LP on *deklaratiivne* programmeerimine;
- Programm kirjutatakse loogika keeles
- Programmi täitmine põhineb
  - loogika printsiipidel ja
  - kasutab automaattõestamise protseduure - resolutsioon, unifitseerimine;
- LP keel on Prolog (PROgramming in LOGic) ,

kuid LP ≠ Prolog;

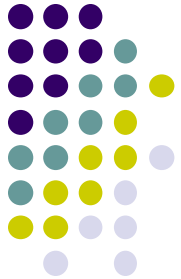
https://en.wikipedia.org/wiki/Programming_paradigm#cite_note-4



1.1. Miks loogiline programmeerimine?

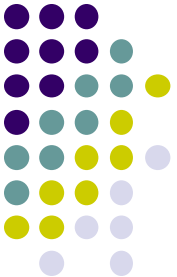
- LP sobib *tehisintellekti rakenduste* programmeerimiseks:
  - loomuliku keele analüüs (DCG grammatikareeglid)
  - ekspertsüsteemid (otsingu- ja järeldusreeglid)
  - kujundituvastus (tuvastusreeglid)
  - kitsendustega planeerimine (logistika, marsruudi otsimine)
  - rekursiivsete funktsioonide püsipunkti arvutus
  - jne

Milleks ei sobi loogiline programmeerimine?



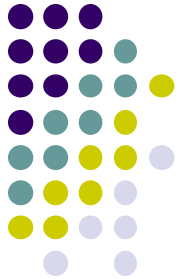
- Ei ole programmeerimiskeelt, mis sobiks ühtviisi hästi kõikide ülesannete lahendamiseks!
- LP ei sobi
  - kiireteks numbrilisteks arvutusteks n. maatriksarvutused, võrrandite lahendamine, numbriline optimeerimine
  - OOP, kuigi OO on toetatud mõnes prologis
  - graafiliste kasutajaliideste programmeerimiseks
  - masingraafikaks ja animatsioonideks

1.1. Miks peaks õppima loogilist programmeerimist?



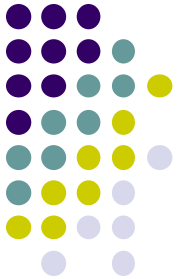
- LPõ petab mõtlema *probleemikeskselt* ja esitama lahenduskäigu *abstraktsel* kujul, konkreetse lahenduse leiab Prologi tuletusmootor
  - Programmi põhifunktsioonid:
    - probleemide *esitamine* abstrakses vormis
    - abstraktsioonide teisendamine ja sidumine omavahel
    - seoste põhjal arvutamine/otsuste tegemine
  - *Programeerimiskeel* peab võimaldama
    - kirjeldada ja analüüsida abstraktsioone nii inimesele, kui *arvutile* sobival kujul
  - *Deklaratiivsed* keeled sobivad enam kui imperatiivsed
    - abstraktsete objektide ja nende seoste kirjeldamiseks
    - *väldivad protseduurseid detaile, kompaktne programm*

1.1. Mis on loogiline programmeerimine?



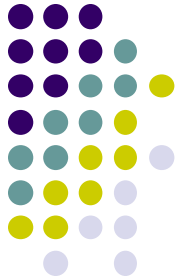
- Universaalne keel omaduste/seoste abstraktseks kirjeldamiseks on loogika.
- → LP on programmeerimine loogika keeles!
- Prolog – *programming in logic*
- LP ≠ Prolog

1.2 LP ajalugu



- Prolog (1972)
  - Alain Colmerauer, Phillipe Roussel;
- Edinburgh Prolog (1980 algus)
  - David Warren;
- 1980 - Jaapani 5. põlvkonna arvuti loomise projekt
- 1980 – tänapäev – LP laiendamine teiste programmeerimise paradigmadega:
  - paralleelsus, OO, andmetüübid jm
  - palju Prologi dialekte
  - OWL

1.3 LP meetod



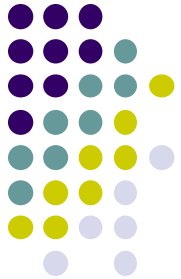
- Piiritleda valdkond:
  - reaalse maailma modelleeritav situatsioon (*domain, use-cases*)
  - määratleda sellega seotud põhimõisted/olemid
  - defineerida mõisteid iseloomustavad atribuudid ja nende omadused
  - defineerida seosed olemite ja nende atribuutide vahel
- Formaliseerida valdkonna kirjeldus Horni lausetena:
  - tulemusena tekib hulk Horni lauseid (faktid, reeglid ja päringud)

    Fakt:
    - **raamatukogu** asub 2. korrusel, % fakt
    - Sokrates on kreeklane % fakt

    Reegel:
    - kui keegi on kreeklane, siis on ta inimene % reegel
- Formuleerida päringud teadmiste struktuuril

  Päring:
  - Kas Sokrates on inimene?

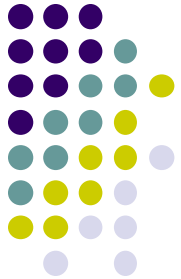
LP “õrnad” kohad



- Teadmiste esitamise ja programmi efektiivsus oleneb reeglite *kujust:*
  - päringu tulemus oleneb otsingureeglist ja faktide järjestusest teadmusbaasis
  - tagurdamismehhanismist (*backtracking*) arusaamine nõuab otsingumootori head tundmist
- Keeruline on saavutada “puhast deklaratiivsust”
- Praktilises programmeerimises vaja ka “madala taseme” käske:
  - kasutajaliidese juhtimine,
  - failisüsteemi käsud,
  - stringioperatsioonid

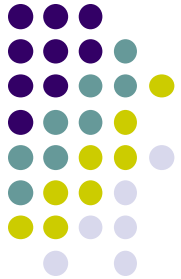
  jms.

LP edasiarendused



- Laiendamine teiste programmikeelte paradigmadega
  - Functional logic programming
    - http://www.informatik.uni-kiel.de/~mh/FLP/
    - keeled Curry and Mercury.
- Efektiivsuse suurendamine
  - Concurrent prolog
    - Curry, ToonTalk, Janus, Alice
- Probleem-orienteerituse suurendamine
  - Constraint Logic Programming
    - http://en.wikipedia.org/wiki/Constraint_logic_programming
  - Semantiline veeb
    - http://hcs.science.uva.nl/projects/SWI-Prolog/articles/mn9c.pdf

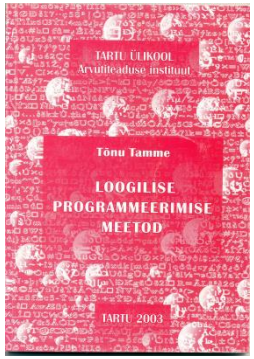
Kursuse sisu (mitte loengute järjestuses)

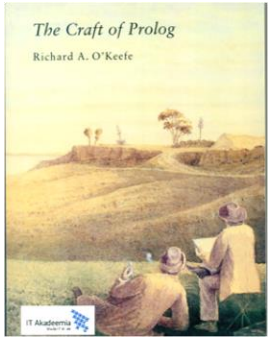


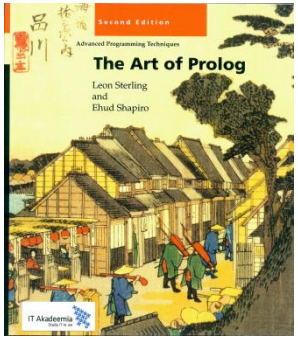
- Alusmõisteid loogikast
  - Loogikasüsteem (faktid ja reeglid)
  - Termid, unifitseerimine ja võrdlemine
  - Tõestusmeetod - resolutsioon
- LP andmestruktuurid (listid, semantilised võrgud, freimid)
- LP denotatsiooniline ja operatsiooniline semantika
- Prologi süntaks ja operaatorid
- Prologi otsingumootor, otsingu juhtimine
- Arendused: kitsenduste süsteemi kirjeldamine ja lahendamine
- Rakendusnäiteid:
  - teadmusbaasid ja reisiplaani koostamine
  - loomulike keelte analüüs (lausete parsimine)
  - kujundituvastus ja keerdülesannete lahendamine (sudoku, krüptoülesanded)
- Näpunäiteid praktiliseks programmeerimiseks: integreerimine Java ja C++ga.

Õppematerjal

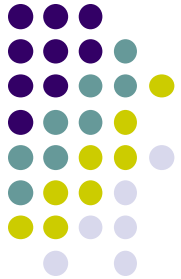
- Õpikud TTÜ raamatukogus:
  - Tõnu Tamme. Loogilise programmeerimise meetod. Tartu Ülikool 2003. (algajatele)
  - R.A.O'Keefe The Craft of Prolog, MIT Press (sissejuhatav)
  - L. Sterling, E. Shapiro, The Art of Prolog. (edasijõudnutele)
  - I. Bratko,"Prolog Programming for Artificial Intelligence", Addison–Wesley Ltd. (rakendusprogrammerijatele)







Lisamaterjal

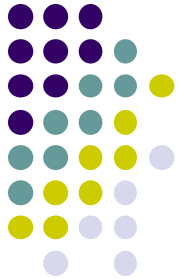


- Ajakirjad:
  - The Journal of Logic and Algebraic Programming
    - (http://www.informatik.uni-trier.de/~ley/db/journals/jlp/jlap.html)
  - Theory and Practice of Logic Programming
    - (http://www.cwi.nl/projects/alp/TPLP/)
- SWI prologi help

Veel kasulikke linke

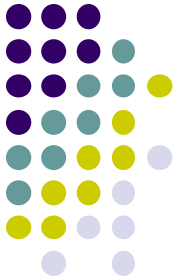
- Peter Hancox. Prolog and Logic Programming. School of Computer Science in the University of Birmingham, UK.
  - http://www.cs.bham.ac.uk/~pjh/prolog_course/sem242.html
- Ischislenie Vyskazyvaniy I Logicheskoe Programmirovanie
  - 2012/5/2 Vladimir Kulakov **Vene keeles!**
- The World Wide Web Virtual Library: Logic Programming
- Guide to Prolog Programming
  - http://kti.mff.cuni.cz/~bartak/prolog/implementations.html
- Object-Oriented Prolog
  - http://www.cetus-links.org/oo_prolog.html
- Jonathan Bowen Logic Programming page
  - http://formalmethods.wikia.com/wiki/Logic_programming

Kuidas hankida oma Prolog?



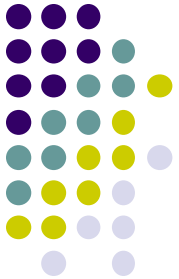
- Unix, Windows, Linux:
  - ALS (Applied Logic Systems, Inc.) Prolog compiler
  - BinProlog, BinNet Corp. See also Jinni (Java INference Engine and Networked Interactor).
  - GNU Prolog compiler - free Prolog compiler with constraint solving over finite domains.
  - IF/Prolog system. IF Computer. Unix, Windows 95/98/NT.
  - IT ProLog. IT Masters. (Unix and Windows NT).
  - LPA WIN-PROLOG, MacProlog32 and Prolog++. Logic Programming Associates Ltd.
  - Quintus Prolog. For Unix and MS Windows.
  - SICStus Prolog (commercial, portable) Unix machines, Windows.
  - **SWI-Prolog**. Unix and MS Windows. Portable.
- PC Prologid:
  - YAP Prolog System (Yet Another Prolog) – kiire Prologi kompilaator, Linux/Solaris/Windows NT, 95, 98. Akadeemiline litsents vaba.
  - Amzi Prolog + Logic Server. (Commercial). Windows 3.x, 95, WFW, NT 3.5x, DOS, Extended-DOS. Allows embedding of Prolog components in C/C++, Visual Basic, Delphi, Access, etc. http://www.amzi.com/download/freedist.htm
  - ADA Prolog (aeglane) ja ESL Prolog (hea, kiire).
  - LPA WIN-Prolog. Windows 3.1, Macintosh ja MS-DOS.
  - Qu-Prolog. Support symbolic computation for mathematical notations and languages such as Z.
  - Visual Prolog from the Prolog Development Center. DOS, Windows 3.1/95/98, NT, Linux.

Mis prologi kasutada praktikumides?

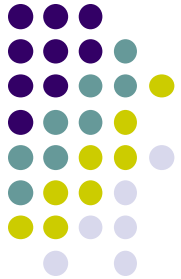


- Praktikumides ainult SWI prolog!!
  - http://www.swi-prolog.org/

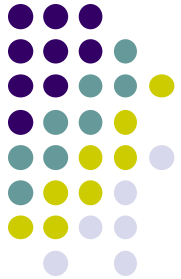
Küsimused?



Loogilise programmeerimise põhimõisted

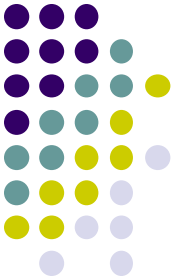


Mis on loogika keel?



- 3 komponenti:
  1. Süntaks: reeglid valemite kirjutamiseks
  2. Semantika: valemitele tähenduse andmine
  3. Tõestusaparaat: reeglid olemasolevatest valemitest (aksioomid ja tuletatud reeglid) loogiliselt korrektsete järelduste tegemiseks.

Kuidas rakendada loogika reegleid (1)?

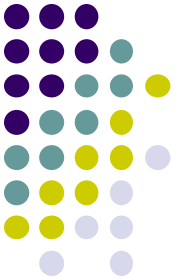


- Näite

  Olgu antud hulk fakte

  - Juhan on Liia isa
  - Kati on Liia ema
  - Liia on Juku ema
  - Ken on Karini isa
- Kes on Juku vanavanemad?
- Kes on Karini vanavanemad?

Kuidas rakendada loogika reegleid (2)?



- Faktid Prologis:

  isa(juhan, liia).

  ema(kati, liia).

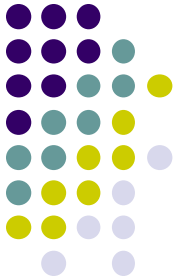
  ema(liia, juku).

  isa(ken, karini).

kus

- ‘isa’ ja ‘ema’ on *predikaatide nimed*
- ‘juhan’, ‘liia’, ‘kati’, ‘ken’ on *termid*
- ‘ema(liia, juku).’ on atomaarne valem, mis väljendab postuleeritud fakti.

Programm kui loogika valem



- Esitame seose ‘vanavanem’

```
vanavanem(X, Z) :- vanem(X, Y), vanem(Y, Z).
vanem(X, Y) :- isa(X, Y).
vanem(X, Y):-  ema(X, Y).
```

See formaliseerib tingimusliku lause:

„**Mistahes** X,Y,Z korral,

kui X on Y vanem **ja** Y on Z vanem,

siis X on Z vanavanem“.

Programm kui loogika valem

- Konjugeerides faktid ja reeglid, saame loogilise programmi:

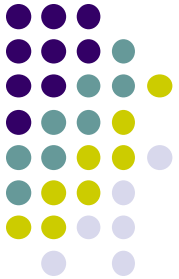
```
isa( juhan, liia).
ema( kati, liia).
ema( liia, juku).
isa( ken, karini).
vanavanem(X, Z) :- vanem(X, Y), vanem(Y, Z).
vanem(X, Y) :- isa(X, Y) ; ema(X, Y).
```

implikatsioon

konjunktsioon

disjunktsioon

Prologi päring



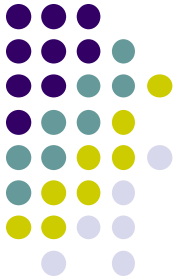
- Kes on Juhani lapselapsed

?- vanavanem(juhan, Lapselaps).

- Prolog tagastab vastuse

  Lapselaps = juku

Prolog programmi struktuur



- Prologi programm koosneb lausetest kujul

  A :- B1, B2, …, Bn.

  - kus A ja B-d on atomaarvalemid
  - :- tähistab loogilist implikatsiooni.
  - Kui lause tõesus ei sõltu eeldustest, on tegemist faktiga mis kirjutatakse kujul

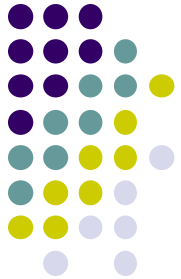
    A.

  Näiteks:

  `ilus_ilm` on 0-kohaline fakt, sest tal puuduvad argumendid.

  `ilus_ilm(päike, soe)` on 2-argumendiline fakt

Kokkuvõte



- Loogiline programm koosneb kindlat süntaktilist kuju omavatest lausetest – Horni lausetest.
- Programmi täitmine on Horni lausetest uute lausete tuletamine.

Uurimis- ja lõputöö teemad

swiss space center

- https://www.spacecenter.ch/igluna/