**Лекция № 8.** Жанама адрестеуі бар есептеу моделі. Функционалды есептеу моделдері.

**МАҚСАТЫ:** Жанама адрестеуі бар есептеу моделі (Равнодоступная адресная машина , *РАМ – машина*), шектеусіз регистрлі машина (МНР), алгоритмдік шешімі болмау, шешімі болмайтын проблемалар, алгоритмдік келтіру ұғымдарымен таныстыру.

**СҰРАҚТАР**:

1. Жанама адрестеуі бар есептеу моделі (Равнодоступная адресная машина, *РАМ – машина*)
2. Шектеусіз регистрлі машина (МНР)Алгоритмдік шешімі болмау,
3. Шешімі болмайтын проблемалар.
4. Алгоритмдік келтіру.

**НЕГІЗГІ ҰҒЫМДАР:** Шектеусіз регистрлі машина (МНР).Жартылай шешімнің болуы. Алгоритмдік шешімі болу және болмау ұғымы

**Жанама адрестеуі бар есептеу моделі** (ТЕҢҚОЛЖЕТІМДІ АДРЕСТІК МАШИНА)

Жанама адрестеуі бар есептеу моделі (Теңқолжетімді адрестік машина РАМ) – есептеу қондырғысының сандық моделі. Бұл модель реалды есептеуіш машиналарына ең жақын модель болып табылады және нақты есептерге алгоритмдер күрделілігін теориялық бағалауда барынша нақты қолдануға мүмкіндік береді.

Машинаның жады регистрлерден (ұяшықтардан) тұрады. Әрбір ұяшықтың адрестері бар және туынды сандардан тұрады. 0 нөмірі бар регистр сумматор деп аталады.

***Бағдарлама*** – нөмірленген командалар тізбегі. Команда келесі түрде болады

<код операции> <операнд>

Операциялардың кодтары:

Load, Store, Add, Sub, Mult, Div, Read, Write, Jump, JgtZ, Jzero, Halt.

***Операнд*** төмендегі үш түрдің бірінде болады

= *i*, *i*, \**i*,

мұндағы *i* – натурал сан.

 i нөмірі бар регистрдің құрамын c (i) арқылы белгілеймиз. v(a) мәні операнда a оның түріне байланысты келеісі түрде анықталады

*v*(*a*) – *i саны*, егер *a* түрі =*i*,

*v*(*a*) – *c*(*i*) саны, егер *a* түрі *i*,

*v*(*a*) – *c*(*c*(*i*)) саны, егер *a* түррі \**i*.

Алгоритмдерддің күрделілігін бағалаған кезде жағдайға байланысты әр түрлі командаларды қолданады. Аз сандармен және аз регистрлермен жұмыс жасаған кезде әрбір команда бірлік уақытты талап ететін теңөлшемді салмақ шындыққа сәйкес келеді. Улкен сандармен жұмыс жасаған кезде команданың жұмыс жасау уақыты операнданың уақытынан сонымен бірге оның адресінің мәнінен тәуелді логарифмдік салмақ қолданылады. Команданың салмағын анықтағанда *L*: *Z* → *Z*, функциясы қолданылады, ол сандық жазбаның ұзындығын көрсетеді

⎩⎨⎧=≠+=.01,01)log()(припри*iiiiL*

Асимптотикалық бағаларды алған кезде логарифмнің негізіне көп мән берілмейді.

Салмақ t(a) операнда a оның түріне байланысты келесідей анықталады

*t*(*a*) = *L*(*i*), егер *a* түрі =*i*,

*t*(*a*) = *L*(*i*) + *c*(*i*), егер *a* түрі *i*,

*t*(*a*) = *L*(*i*) + *L*(*c*(*i*)) + *L*(*c*(*c*(*i*))), егер *a* түрі \**i*.

Load(*a*) *с*(0) := *v*(*a*) *t*(*a*)

Store(*i*) *c*(*i*) := *c*(0) *L*(*c*(0)) + *L*(*i*)

Store(\**i*) *c*(*c*(*i*)):=*c*(0) *L*(*c*(0)) + *L*(*i*) + *L*(*c*(*i*))

Add(*a*) *с*(0) := *c*(0) + *v*(*a*) *L*(*c*(0)) + *t*(*a*)

Sub(*a*) *с*(0) := *c*(0) − *v*(*a*) *L*(*c*(0)) + *t*(*a*)

Mult(*a*) *с*(0) := *c*(0)·*v*(*a*) *L*(*c*(0)) + *t*(*a*)

Div(*a*) *с*(0) := *c*(0) div *v*(*a*) *L*(*c*(0)) + *t*(*a*)

Read(*i*) *c*(*i*) := келесі сан *L*(*i*) + *L*(*c*(*i*))

Read(\**i*) *c*(*c*(*i*)):= келесі сан *L*(*i*) + *L*(*c*(*i*)) +

*L*(*c*(*c*(*i*))

Write(*a*) теру *v*(*a*) *t*(*a*)

Jump(*a*) a 1 нөмірі бар командаға өту

JgtZ(*a*) *a* нөмірі бар командаға өту, егер *c*(0)>0 *L*(*c*(0))

Jzero(*a*) *a* нөмірі бар командаға өту, егер *c*(0)=0 *L*(*c*(0))

Halt(*a*) есептеуді аяқтау 1

**Функциональды есептеу моделі** —[дискретті математик](http://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%81%D0%BA%D1%80%D0%B5%D1%82%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)а бөлімі және программалау [парадигмасы](http://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) , мұнда есептеу процесі математикалық тұрғыда функция мәнін есептеу ретінде қарастырылады.

[Императивті программалау](http://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) парадигмасында , мысалы автоматтар теориясында, есептеу процесін күйлердің тізбекті өзгеруі ретінде қарастырады. Функциональды программалауда есептеу процесін барлық күйлерінің жиынтығы нақты түрде тізім ретінде қарастырылады. Ол программа күйін сақтамайды , функция мәнін бастапқы деректер мен басқа функциялардың мәндері бойынша есептейді.шығыстық деректер тек кірістікке тәуелді. Бірдей аргументі бар функцияны шақыру әрқашан бір нәтиже береді.

Танымал функциональды программалау тілдері:

* [LISP](http://ru.wikipedia.org/wiki/LISP) - ([Джон МакКарти](http://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%BA%D0%9A%D0%B0%D1%80%D1%82%D0%B8%2C_%D0%94%D0%B6%D0%BE%D0%BD), [1958](http://ru.wikipedia.org/wiki/1958)) және оның диалекттері:
* [Scheme](http://ru.wikipedia.org/wiki/Scheme)
* [Clojure](http://ru.wikipedia.org/wiki/Clojure)
* [Common Lisp](http://ru.wikipedia.org/wiki/Common_Lisp)
* [F#](http://ru.wikipedia.org/wiki/F_Sharp) — [.NET](http://ru.wikipedia.org/wiki/.NET_Framework) платформасы үшін
* [Haskell](http://ru.wikipedia.org/wiki/Haskell) — [таза](http://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D1%82%D0%BE%D1%82%D0%B0_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)  функциональды тіл. [Хаскелла Карри](http://ru.wikipedia.org/wiki/%D0%9A%D0%B0%D1%80%D1%80%D0%B8%2C_%D0%A5%D0%B0%D1%81%D0%BA%D0%B5%D0%BB%D0%BB) құрметіне аталды
* [Erlang](http://ru.wikipedia.org/wiki/Erlang) — ([Joe Armstrong](http://ru.wikipedia.org/w/index.php?title=Joe_Armstrong&action=edit&redlink=1" \o "Joe Armstrong (страница отсутствует)), [1986](http://ru.wikipedia.org/wiki/1986)) процестерді қолдайтын функциональды тіл
* [APL](http://ru.wikipedia.org/wiki/APL) — [MATLAB](http://ru.wikipedia.org/wiki/MATLAB)-тың алдыңғысы.
* [ML](http://ru.wikipedia.org/wiki/ML) ([Робин Милнер](http://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BB%D0%BD%D0%B5%D1%80%2C_%D0%A0%D0%BE%D0%B1%D0%B8%D0%BD), [1979](http://ru.wikipedia.org/wiki/1979), қазіргі диалекттері [Standard ML](http://ru.wikipedia.org/wiki/SML) и [Objective CAML](http://ru.wikipedia.org/wiki/OCaml%22%20%5Co%20%22OCaml)).
* [Scala](http://ru.wikipedia.org/wiki/Scala_%28%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F%29)
* [Miranda](http://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D1%80%D0%B0%D0%BD%D0%B4%D0%B0_%28%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F%29) ([Дэвид Тёрнер](http://ru.wikipedia.org/wiki/%D0%A2%D1%91%D1%80%D0%BD%D0%B5%D1%80%2C_%D0%94%D1%8D%D0%B2%D0%B8%D0%B4), [1985](http://ru.wikipedia.org/wiki/1985), Haskell тілін дамытты).
* [Nemerle](http://ru.wikipedia.org/wiki/Nemerle) — қоспа функциональды/императивті тіл.
* [XQuery](http://ru.wikipedia.org/wiki/XQuery)

Функциональды программалау тілдері ғылыми тұрғыдан қызықтырады, алайда Erlang, [OCaml](http://ru.wikipedia.org/wiki/OCaml%22%20%5Co%20%22OCaml), Haskell, Scheme (1986 кейін) [R](http://ru.wikipedia.org/wiki/R_%28%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F%29) (статистика), [Mathematica](http://ru.wikipedia.org/wiki/Mathematica%22%20%5Co%20%22Mathematica) (символьная математика), J мен K (финанстық анализ), және [XSLT](http://ru.wikipedia.org/wiki/XSLT) ([XML](http://ru.wikipedia.org/wiki/XML)) коммерциялық программалау индустриясында кеңінен қолданылады.

λ-есептеу функцияның және оны есептеудің теориялық негізін қалады.  Функциональды тілдердің негізі болды. [Комбинаторлық логика](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%B1%D0%B8%D0%BD%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F_%D0%BB%D0%BE%D0%B3%D0%B8%D0%BA%D0%B0) бұған ұқсас, бірақ ол λ-есептеуге қарағанда неғұрлым абстрактты, әрі ертерек ашылды . [Комбинаторлық логика](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%B1%D0%B8%D0%BD%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F_%D0%BB%D0%BE%D0%B3%D0%B8%D0%BA%D0%B0) [Unlambda](http://ru.wikipedia.org/wiki/Unlambda) тілінде қолданылады. Екеуі де математиканығ принциптерін нақты дәл сипаттау үшін жасалды.

**Алғашқы** функциональды **тіл** [Джон МакКарти](http://ru.wikipedia.org/wiki/%D0%94%D0%B6%D0%BE%D0%BD_%D0%9C%D0%B0%D0%BA%D0%9A%D0%B0%D1%80%D1%82%D0%B8)  жасаған [Lisp](http://ru.wikipedia.org/wiki/Lisp) тілі.

Функциональды программалауға тән кейбір **концепциялар мен парадигмалар** императивті программалауға керағар болады ([объектті-бағдарланған программалауды қоса](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%22%20%5Co%20%22%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5))..

Функциональды тілдерде [цикл](http://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D0%BA%D0%BB_%28%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%29) әдетте рекурсия түрінде іске асады.

Функциональды тілдерді функция аргументтерінің өңдеу процесінде қалай өзгеретініне байланысты жіктеуге болады. Мысалы:

print length([2+1, 3\*2, 1/0, 5-4]), нәтижеде 3-ші элементте қате шығады, нольге бөлуге болмайды. Қатаң түрде қарастырмаса, өрнек мәні 4 болады.

Қатаң емес есептеу [Miranda](http://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D1%80%D0%B0%D0%BD%D0%B4%D0%B0_%28%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F%29),[Clean](http://ru.wikipedia.org/wiki/Clean) ,[Haskell](http://ru.wikipedia.org/wiki/Haskell%22%20%5Co%20%22Haskell) функциолналды тілдерінде іске асқан.

**ҚОРЫТЫНДЫ:** Теңқолжетімді адрестік машина (РАМ) – есептеу қондырғысының сандық моделі. Бұл модель реалды есептеуіш машиналарына ең жақын модель болып табылады және нақты есептерге алгоритмдер күрделілігін теориялық бағалауда барынша нақты қолдануға мүмкіндік береді. Машинаның жады регистрлерден (ұяшықтардан) тұрады. Әрбір ұяшықтың адрестері бар және туынды сандардан тұрады. 0 нөмірі бар регистр сумматор деп аталады. МНР әрқайсысы R1, R2,R3,..., деп белгіленетін шексіз регистрлердің санынан тұрады

кейбір командалардың комегімен регистрлердің құрамын өзгерте алады. Функциональды есептеу моделі —[дискретті математик](http://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%81%D0%BA%D1%80%D0%B5%D1%82%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0%22%20%5Co%20%22%D0%94%D0%B8%D1%81%D0%BA%D1%80%D0%B5%D1%82%D0%BD%D0%B0%D1%8F%20%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)а бөлімі және программалау [парадигмасы](http://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F%22%20%5Co%20%22%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) , мұнда есептеу процесі математикалық тұрғыда функция мәнін есептеу ретінде қарастырылады