# Гарчиг

[Гарчиг 1](#_Toc326451550)

[Оршил 3](#_Toc326451551)

[Ажлын хураангуй 4](#_Toc326451552)

[Бүлэг 1. Блок схем, түүнийг хэрэгжүүлэх аргазүй 7](#_Toc326451553)

[1.1 Блок схем 7](#_Toc326451554)

[1.2 Блок схемийн стандартууд 9](#_Toc326451555)

[1.2.1 Монголд баримталдаг стандарт 9](#_Toc326451556)

[1.2.2 IBM байгууллагын баримталдаг стандарт 10](#_Toc326451557)

[1.3 Блок схем хэрэгжүүлэх аргазүй, дэвшүүлсэн санаа 12](#_Toc326451558)

[1.3.1 Програмчлалын хэл хэрэгжүүлэх аргазүй 13](#_Toc326451559)

[1.3.2 Програмчлалын хэлний өгөгдөл хадгалах зарчим 18](#_Toc326451560)

[Бүлэг 2. Блок схем хэрэгжүүлэгч, шинжилгээ, зохиомж 20](#_Toc326451561)

[2.1 Програм хангамжийн тодорхойлолт 20](#_Toc326451562)

[2.1.1 Хэрэглэгчийн шаардлагын шинжилгээ 23](#_Toc326451563)

[2.2 Хэрэглэгчийн интерфейс 23](#_Toc326451564)

[2.2.1 Хэрэглэгчийн интерфейсийн шаардлагын шинжилгээ 23](#_Toc326451565)

[2.2.2 Хэрэглэгчийн интерфейсийн зохиомж 24](#_Toc326451566)

[2.3 Блок схем зурагч модуль 26](#_Toc326451567)

[2.3.1 Блок схем зурагч модулийн шаардлагын шинжилгээ 26](#_Toc326451568)

[2.3.2 Блок схем зурагч модулийн use case диаграм 27](#_Toc326451569)

[2.3.3 Блок схем зурагч модулийн оролт, гаралтын зохиомж 29](#_Toc326451570)

[2.4 Хийсвэр код хэрэгжүүлэгч модуль 30](#_Toc326451571)

[2.4.1 Хийсвэр код хэрэгжүүлэгчийн шаардлагын шинжилгээ 30](#_Toc326451572)

[2.4.2 Хийсвэр код хэрэгжүүлэгч модулийн use case диаграм 31](#_Toc326451573)

[2.4.3 Хийсвэр код хөрвүүлэгч модулийн оролтын зохиомж 34](#_Toc326451574)

[Бүлэг 3. Блок схем хэрэгжүүлэгч програм хангамжийн хөгжүүлэлт 35](#_Toc326451575)

[3.1 Архитектурын сонголт 35](#_Toc326451576)

[3.1.1 Програмчлалын хэлний сонголт 35](#_Toc326451577)

[3.1.2 Хөгжүүлэлтийн орчны сонголт 35](#_Toc326451578)

[3.1.3 Үйлдлийн системийн сонголт 35](#_Toc326451579)

[3.2 Програм хангамжийн архитектур 35](#_Toc326451580)

[3.2.1 Модулиуд 36](#_Toc326451581)

[3.3 Туршилт 40](#_Toc326451582)

[3.3.1 Туршилтын өгөгдөл 1 40](#_Toc326451583)

[3.3.2 Туршилтын өгөгдөл 2 42](#_Toc326451584)

[3.3.3 Туршилтын үр дүн 42](#_Toc326451585)

[Дүгнэлт 43](#_Toc326451586)

[Ном зүй 44](#_Toc326451587)

# Оршил

Хүмүүс өдөр тутмын үйл ажиллагаандаа олон тооны алгоритмуудыг гүйцэтгэж байдаг боловч тэр бүр тэдгээр алгоритмуудыг мэдэрдэггүй. Тухайлбал тухайн өдрийн ажил төрөл болон тодорхой нэгэн ажлыг хийхдээ тогтсон дэс дарааллыг мөрддөг. Алгоритмын тухай ойлголт нь нэгэн төрлийн бодлогуудыг бодох болон ерөнхий арга олж тогтоох гэсэн оролдлоготой уялдан математикт анх үүссэн. Алгоритм хэмээх нэр томъёог арифметикийн дөрвөн үйлдлийг гүйцэтгэх дүрэм боловсруулсан Узбекийн математикч Муххамед Ибн Мусса Аль-Хорезмын нэрнээс гаралтай гэж үздэг. [1]

Харин математик болон Компьютерийн шинжлэх ухааны салбарт тооцооллыг хийхэд явагдах алхамуудын дарааллыг алгоритм гэдэг. Алгоритмыг илэрхийлэх олон арга байдгийн нэг нь “Блок схем” гэх дүрслэлийн арга юм. Алгоритм, програмчлалын талаарх анхны мэдэгдэхүүнийг өгч цааш гүнзгийрүүлэн суралцах суурийг тавьж өгдөг хамгийн энгийн түгээмэл арга нь блок схемийг сургалтанд ашиглах явдал байдаг. Уг дүрслэлийн аргыг ихэвчлэн алгоритм анхлан суралцагч нар хэрэглэж алгоритмын талаарх анхны мэдэгдэхүүнийг маш ойлгомжтойгоор олж авдаг.

Харин блок схемийг зурж байхад зурж буй талбараас хэтрэх, хэтрүүлэхгүйн тулд зайг тохируулах, зайг тохируулахын тулд зурах блок схемее бүтнээр төлөвлөх, тухайн зурагдсан блок схемийн алгоритм зөв ажилж буй эсэхийг шалгах зэрэг төвөгтэй асуудал олон байдаг. Уг асуудлуудыг шийдснээр суралцагч тулгарч буй асуудлыг тухай үе бүрт нь шийдэж, зурж байх явцдаа бодох зэрэг олон боломжуудыг хангаж байгаа нь алгоритмын хичээлийг илүү бодитой сурах боломжийг олгож байгаа юм. Мөн програмчлалын хэлний талаар анхны мэдэгдэхүүн ажиллах зарчмын талаар сайн ойлгож авах юм. Хүн болгон компьютертой болсон эдүгээ үед програм хангамжийн тусламжтайгаар алгоритм, мэдээлэлзүйн хичээлийг сурах нь илүү үр дүнтэй болно гэсэн үндсэн дээр энэхүү ажлыг хийх болсон юм.

# Ажлын хураангуй

#### Товч агуулга

Блок схемийг (Flow Chart) тодорхой стандартын дагуу компьютерийн тусламжтайгаар хялбараар зурж, зөв зурагдсан эсэхийг шалгах, түүнийг алхам алхмаар ажиллуулж хянах зориулалттай компьютерийн зохих мэдлэгтэй алгоритм суралцагч нар болон мэдээллийн технологийн чиглэлээр суралцаж буй оюутан сурагчидад зориулагдсан програм хангамж юм.

#### Ажлын зорилго

Уг ажлын үндсэн зорилго нь алгоритм суралцагч нар ашиглаж болохуйц блок схемийн тогтсон стандартаар блок схемийг зурж, хэрэгжүүлэх програм хангамжийг үйлдвэрлэх, цаашдийн зорилго нь уг програм хангамжийг сургуулиудын мэдээлэлзүйн хичээлийн програмд хэрэглэж болохуйц бүтээгдэхүүн болгох зорилготой.

#### Зорилтууд

Уг ажлын хүрээнд дараах зорилтуудыг тавьж ажиллаа:

* Ашиглаж болохуйц програм хангамжийн бүтээгдэхүүнийг бэлэн болгох
* Блок схемийн стандартууд, блок схемийг зурах аргазүйг судлах
* Програмчлалын хэлний архитектур, түүнийг хэрэгжүүлэгчийн ажиллагааг судлах
* Програм хангамжийн инженерчлэлийн стандартын дагуу бакалаврын судалгааны ажлыг хийж гүйцэтгэх
* Үр дүнгээс хамааран бүтээгдэхүүнээ борлуулах

#### Шинэлэг тал

Програмчлалын талаарх анхны мэдэгдэхүүнийг өгч цааш гүнзгийрүүлэн суралцах суурийг тавьж өгдөг хамгийн энгийн түгээмэл арга нь блок схем (Flowchart Diagram)-ийг сургалтанд ашиглах явдал байдаг. Блок схемийг ашиглахад алгоритмыг дүрслэлээр төлөөлүүлдэг нь ойлгоход хялбар боловч зөв алгоритм болсон эсэхийг шалгаж, үр дүнг нь харах, өөр өөр стандартыг ашиглах төвөгтэй байдаг. Уг төвөгтэй байдлыг програмчлалын дагуу шийдэж стандартуудыг өөр хооронд нь хөрвүүлж нэг програмаас удирдан, зураг хэлбэрээр гаргаж мөн файл хэлбэрээр хадгалж, түүнийгээ ажиллуулж байгаа нь уг ажлын шинэлэг тал юм.

#### Ажлын үр дүн

Уг бакалаврын төгсөлтийн ажлыг хийж гүйцэтгэснээр өмнө үзэж судалсан хичээлүүдээ бататган, мэдээллийн технологийн салбарын инженер бидний гол хэрэгсэл болох програмчлалын хэлний ажиллах зарчим, архитектурыг судалж мөн том хэмжээний програм хангамж хөгжүүлэх туршлагыг өөртөө хуримтлуулсан юм.

#### Өргөтгөх боломж

Програмчлалыг уян хатан хийснээр тухайн програмчлал дээр тулгуурлан ижил төрлийн диаграм зурах, хэрэгслүүд буюу plug-in дэмжих зориулалттайгаар хийгдэх тул шинэ төрлийн бүтээгдэхүүн үүсгэх зэрэг өргөн боломжтой. Мөн java хэл дээр бичигдсэн тул нээлттэй эх код бүхий програм хангамж болох юм. Тиймээс үүнийг олон нийтээр хөгжүүлэн өргөтгөх боложмтой юм.

#### Ач холбогдол, хэрэглээ

Хэрвээ уг програм хангамжийг сургалтанд ашиглавал суралцагч нар алгоритмын талаар маш сайн ойлголт авах бөгөөд цаашлаад мэдээллийн технологийн салбарын чадварлаг инженерүүдийг бэлтгэх гол эхлэлийг тавьж өгж болох юм. Мөн блок схемийн тодорхой стандартыг баримталж олон нийтэд түгээснээр хэрэглэгчид тухайн блок схемийг зурах стандартыг хэвшмэл болгож нийтээрээ нэг стандартыг баримтлах ач холбогдолтой.

#### Ашигласан технологи

Уг бүтээгдэхүүнийг дунд сургууль болон их сургуулийн лабораторийн орчинд ашиглах тул хэрэглэгчид компьютерт суулгац байдлаар ашиглагдах юм. Тиймээс Java Standard Edition-г сонгож програм хангамжийг хийж гүйцэтгэлээ. Уг хэлийг ашигласнаар дараах давуу талуудтай.

* Үйлдлийн систем хамаарахгүй ажиллана.
* Бүтээгдэхүүний хэмжээ бага болно.
* Системын нөөцийг бага шаардана.
* Open Source төрлийн програм хангамж болох тул цаашид өөр олон хэрэглэгчид хөгжүүлэх боломжтой.

Кодчилолыг хийхдээ **Netbeans 7 IDE** програм хангамжийг, шинжилгээ, зохиомжийг хийхдээ **Enterprise Architect** програм хангамжийг, Version Control System буюу “Хувилбар Удирдах Систем”-ээр **Dropbox** хэмээх програм хангамжийг ашиглалаа. Үүний тусламжтайгаар эрсдэлгүй програм хангамжийн хөгжүүлэлт хийсэн юм.

1. Блок схем, түүнийг хэрэгжүүлэх аргазүй
   1. Блок схем

Блок схем нь алгоритмыг илэрхийлэх нэг арга юм. Хүмүүс өдөр тутмын үйл ажиллагаандаа олон тооны алгоритмуудыг гүйцэтгэж байдаг боловч тэр бүр тэдгээр алгоритмуудыг мэдэрдэггүй. Тухайлбал тухайн өдрийн ажил төрөл болон тодорхой нэгэн ажлыг хийхдээ тогтсон дэс дарааллыг мөрддөг. Алгоритмын тухай ойлголт нь нэгэн төрлийн бодлогуудыг бодох болон ерөнхий арга олж тогтоох гэсэн оролдлоготой уялдан математикт анх үүссэн. Алгоритм хэмээх нэр томъёог арифметикийн дөрвөн үйлдлийг гүйцэтгэх дүрэм боловсруулсан Узбекийн математикч Муххамед Ибн Мусса Аль-Хорезмын нэрнээс гаралтай гэж үздэг. [1]

Алгоритмыг дараах хэлбэрүүдээр дүрсэлдэг байна.

1. Хүмүүсийн харилцааны ерөнхий хэл
2. Блок-схем
3. Хийсвэр-код (Псевдо код)
4. Програмчлалын хэл

|  |
| --- |
| C:\Users\LMO\Dropbox\Diplom\Useful files\500px-LampFlowchart.svg.png |

Зураг

Гэрэл асахгүй байх үед хийх үйл ажиллагааны дарааллыг блок схемээр харуулав.

Дээрх аргуудаас Блок схем(flowchart)-ээр дүрслэх арга нь алгоритм болон үйл ажиллагааны алхамуудыг өөр өөр төрлийн тусгай тэмдэглэгээнүүдээр дүрсэлж тэдгээрийг сумаар холбон илэрхийлдэг диаграм зургийн нэг төрөл [2] бөгөөд алгоритмын бүтцийг нүдэнд харагдахуйц байдлаар харуулан алгоритмыг графикаар дүрслэх хэлбэр юм. Блок схемд алгоритмын алхам бүрийг геометрийн дүрсээр тэмдэглэх бөгөөд уг дүрс дотор тухайн алхамын товч утгыг томъёогоор бичнэ. Ийм дүрсийг блок ч гэж нэрлэдэг.

Харин блок схемийг дүрслэх олон арга стандартууд байдаг ба эдгээрт нийтлэг шинж нь:

* Үйл ажиллагааны алхам нь тэгш өнцөгтөөр дүрслэгддэг.
* Шийдвэрлэх нөхцөл нь ромбо дүрсээр дүрслэгддэг.

Жишээг -т харуулав.

* 1. Блок схемийн стандартууд

Блок схемийн стандартууд манай орны хувьд бусад орнуудын барьдаг стандарттай харьцуулахад дүрслэлүүд нь ондоо боловч бусад зүйлс нь ижил байдаг байна.

Блок схемд хэд хэдэн үндсэн ойлголт, элементүүд байдаг.

* **Эхлэл төгсгөл**  
  *Алгоритмын эхлэл төгсгөлийг заадаг*
* **Сум***Өөр хоорондоо огтлолцохгүй ба доош чиглэлтэй байдаг. Хоёр элементийг холбож алгоритм хэрэгжих замыг заадаг. Хэрэв буцаж заасан буюу дээш чиглэлтэй сум байвал энэ нь давталтыг илэрхийлэх үед л хэрэглэгддэг.*
* **Энгийн үйлдэл***Алгоритмын энгийн үйлдэл. Үүнд ямар нэг хувьсагчын утганд өөрчлөлт оруулах, утгыг авах, илэрхийлэл бодогдох зэрэг үйлдэл багтана.*
* **Оролт, гаралт**  
  *Хүний гараас утгыг оруулж өгөх болон ямар нэг утгыг дэлгэцэнд харуулах үйлдэл*
* **Бэлтгэл үйлдэл**  
  *Функц зарлах, хувьсагч зарлах, массим зарлах зэрэг үйлдлүүд багтана*
* **Шийдвэрийн нөхцөл**  
  *2 гаралттай ба нөхцөлөөс хамаарч алгоритм аль 1 гаралтаар шийдэгдэн хэрэгждэг.*
* **Холбогч**  
  Зурагдаж буй талбараас илүү гарах, сум урт зурагдах болон сумнууд огтлолцохоос сэргийлсэн шилжилтийг заахад хэрэглэгддэг.
  + 1. Монголд баримталдаг стандарт

-т МУИС-ийн Математик Компьютерийн сургуульд баримталдаг блок схемийн стандартыг харуулав. [1]

Хүснэгт

МУИС–ийн математик компьютерийн сургуульд баримталдаг стандарт

|  |  |  |  |
| --- | --- | --- | --- |
| № | Алхамын нэр | Монгол стандарт [1] | Тайлбар |
| 1 | Эхлэл, төгсгөл | Эхлэл | Эхлэл төгсгөлийг заана |
| 2 | Утга олгох | A=10 | Хэмжигдэхүүний утгыг өөрчилнө |
| 3 | Нөхцөл шалгах | A=10  бол | Ямар нэг нөхцөлөөс хамааран алгоритм биелэх замыг сонгоно. |
| 4 | Оруулах | A | Өгөгдөл хэмжигдэхүүнүүдийн утгыг оруулна. |
| 5 | Гаргах | A | Өгөгдөл, завсрын хэмжигдэхүүнүүдийн утга болон үр дүнг гаргана |
| 6 | Холбоос |  | Алхамуудын шилжилтүүдийг заахад хэрэглэнэ. |
| 7 | Тайлбар |  | Ямар нэг алхамыг тайлбарлахад хэрэглэнэ. |

* + 1. IBM байгууллагын баримталдаг стандарт

“Олон улсын стандартчилалын байгууллага”-аас ISO-1028 “Information Processing – Flowchart Symbols” нэртэйгээр блок схемийн стандартыг дараах байдлаар гаргасан байдаг байна.[3] Үүнд тэмдэглэгээнүүдийг

1. Үндсэн тэмдэглэгээнүүд
2. Програмчлалын тэмдэглэгээнүүд
3. Оролт/гаралт болон холболт

гэж 3 хэсэгт хуваасан байна. Үүний дэлгэрэнгүйг -т үзүүлэв.

Хүснэгт

ISO-1028 “Information Processing – Flowchart Symbols”[3]

| **№** | **Тэмдэглэгээ** | **Тайлбар** |
| --- | --- | --- |
| Үндсэн тэмдэглэгээ | | |
| 1 | http://bpascal.ru/doc/image/blok-shema2.jpg | Утга олгох буюу энгийн процесст хэрэглэгдэнэ |
| 2 | http://bpascal.ru/doc/image/blok-shema24.jpg | Тайлбар хийхэд хэрэглэгдэнэ |
| Програмчлалын тэмдэглэгээнүүд | | |
| 3 |  | Нөхцөлийг шалгаж алгоритм аль замаар биелэхийг шийднэ. |
| 4 | http://bpascal.ru/doc/image/blok-shema4.jpg | Ямар нэг давталт явагдахаас өмнө индекс регистерийг тохируулах зэрэг машины тохиргоог хийдэг. Энд ямар нэг хувьсагч өөрчлөгддөггүй. |
| 5 | http://bpascal.ru/doc/image/blok-shema5.jpg | Өмнө зарлагдсан функц, модуль болон процедурыг дуудаж ашиглахад хэрэглэдэг. |
| 6 | http://bpascal.ru/doc/image/blok-shema1.jpg | Алгоритмын эхлэл төгсгөлийг заахад хэрэглэдэг. Дотор нь програмын нэр, модулийн нэр эсвэл эхлэл эсвэл төгсгөл гэж бичдэг. |
| Оролт/гаралт болон холболт | | |
| 7 | http://bpascal.ru/doc/image/blok-shema6.jpg | Стандарт оролт гаралтанд ашиглагддаг. Хэрэглэхдээ оруулах эсвэл гаргах гэсэн үгийг хувьсагчийн хамт бичнэ. |
| 8 | http://bpascal.ru/doc/image/blok-shema22.jpg | Нэг блок схемийг өөр блок схемтэй холбоход хэрэглэдэг. Хэрэглэхдээ холбогдож байгаа 2 холбоос дотор ижил тэмдэглэгээгээр тэмдэглэнэ. |

* 1. Блок схем хэрэгжүүлэх аргазүй, дэвшүүлсэн санаа

Блок схем нь блокуудын холбоост дараалал билээ. Блок схем хэрэгжих зарчим нь дээрээс доош чиглэлтэй байдаг. Өөрөөр хэлбэл 1 элементрүү 1 сум орж элементээс 1 сум гардаг гэж хэлж болно. Гэвч энэ шинж чанарыг зөрчдөг нэг блок нь “шийдвэрийн нөхцөл”-ийн блок билээ. Гэвч “шийдвэрийн нөхцөл”-ийг аль нэг нь хэрэгжих хоёр ширхэг “блок схем”-ийг агуулсан нэг элемент гэж үзэж болох юм. Мөн сум буцаж явсан буюу дээш чиглэлтэй тохиолдолд давталт үүсдэг. Энэ тохиолдолд давталт зогсох нөхцөл заавал байх ёстой бөгөөд хэрэв үгүй бол мөнх ажиллах алгоритм болно гэсэн үг. Иймээс давталтыг зогсоох шийдвэрийн нөхцөл зайлшгүй байх хэрэгтэй. Зогсох шийдвэрийн нөхцлийг шалгах хүртэл ажиллах “блок схем”, зогсоогүй үед давталтын эхлэлд очих хүртэл ажиллах “блок схем” гэсэн 2 блок схем агуулна. Үүнийг мөн л өөртөө хоёр ширхэг “блок схем” агуулсан нэг блок гэж үзэж болох юм. Үүнийг тайлбарлаж –т харуулав.

Блок схем зурагдахад ашиглагдах блокуудыг блок бүр дээрээс доош чиглэлд ажиллах дараах 7-н төрлийн блокт хувааж үзэж болно. Үүнд:

* Эхлэл блок
* Төгсгөл блок
* Шийдвэрийн блок
* Давталтын блок
* Оролтын блок
* Гаралтын блок
* Энгийн үйлдлийн блок.

Иймээс эдгээр тус бүрийг нэг комманд гэж үзвэл энэ нь програмчлалын хэл хэрэгжүүлэхтэй ижил болох юм. Блок бүрийг коммандруу хөрвүүлээд түүнийгээ хэрэгжүүлэх нь бидний гол зорилгын нэг юм. Өөрөөр хэлбэл дээрх үйлдлүүдийг багтаасан програмчлалын хэл зохионо гэсэн юм. Үүнийг хийснээр коммандуудыг хэрэгжүүлж байх явцдаа аль коммандыг хэрэгжүүлж байгаа, аль коммандыг хэрэгжүүлэх гэж байгаа, алдааны мэдээлэл зэргийг мэдэх боломжтой болж байгаа юм.

|  |
| --- |
|  |

Зураг

A – Шийдвэрийн нөхцлийг 1 блок болгосон хэлбэр

Б – Давталтыг 1 блок болгосон хэлбэр

* + 1. Програмчлалын хэл хэрэгжүүлэх аргазүй

Ажилд шаардлагатай програмчлалын хэл энгийн бөгөөд алгоритмд хэрэглэгдэх үндсэн элементүүд болох

* гараас унших,
* дэлгэцрүү бичих,
* хувьсагч зарлах,
* утга олгох,
* нөхцөл шалгах,
* давталт хийх
* функц дуудах

үйлдлүүдийг агуулсан байх юм. Коммандуудыг бичиг хэлбэрээр дүрслэх нь алгоритм илэрхийлж буй нийтлэг компьютерийн онцлогт тохирсон арга юм. Иймээс бичиг хэлбэрээр дүрслэхийн тулд түлхүүр үгүүд байх ёстой ба түлхүүр үгүүдээс хамааран аль элемент болохыг таньж хэрэгжүүлэх юм.

Програмчлалын хэлэнд үндсэн 3 ойлголт байдаг. [4] Үүнд:

* **Syntax:** Өгүүлбэрийн харагдац болон бүтэц, өгүүлбэрзүй.
* **Semantics:** Програмистууд өгүүлбэрийг ойлгох ойлголт. Математикчид тоо, тэмдэглэл функцыг, хөгжимчид нот ойлгодогтой ижил.
* **Pragmatics:** Хэлний хэрэглээ, хэлний боломжуудыг багтаасан хэрэглэлт хэрэгжүүлэлтийг хөнгөвчилсөн хэлний зорилгод нийцсэн ойлголт юм.

Бүх програмчлалын хэл дээрх зүйлсийг багтаасан байдаг. Програмчлалын хэлний код хөрвүүлэлт нь үндсэн 2 гол хэсгээр дамжин хэрэгждэг. Эхний хэсэг нь эх кодны алдааг хянагч модуль (*parser*) буюу эх кодны syntax-ийг зөв бичигдсэн эсэхийг шалгадаг модуль. Хоёрдох хэсэг нь тооцоолох модуль буюу эх кодны үр дүнг гаргах оролтын semantic-ийг тодорхойлдог. Ингэснээр систем эх кодны pragmatics-ийг хэрэгжүүлэх юм. Өөрөөр хэлбэл хэрэгжүүлэгч юм. Хэрэгжүүлэх аргазүйг энгийн зургаар ***-***т харуулав.

|  |
| --- |
|  |

Зураг

Програмчлалын хэлний эх кодыг хэрэгжүүлэгч архитектур

Дээрх аргазүйг энгийн жишээ дээр тайлбарлая. *Жишээ нь:*

Арифметик илэрхийллийг хэрэгжүүлэгч програм хангамжийг авч үзье. Илэрхийлэл нь +, -, \*, / гэсэн бинар операторууд ба 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 гэсэн цифрүүдээс бүрдэнэ. Мөн цифрүүд ганцаараа болон нэгдэж тоо үүсгэнэ. Энэ нь хэлний **syntax** юм. Тоонуудын хоорондоо зөвхөн оператаар холбогдох ба операторуудын 2 талд заавал тоо орсон байх ёстой. Энэ нь хэлний **semantics** юм. [4]

**Parser** модуль илэрхийллийг мод бүтэцтэй болгох ба дараах байдлаар **semantics**-ийг үүсгэе.

**Хүснэгт** 3-т харуулав.

Хүснэгт

Илэрхийллийг мод бүтэцтэй болгосон байдал

|  |  |
| --- | --- |
| **Мод** | **Бүтэц** |
| **Цифр** | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 |
| **Оператор** | +, -, \*, / |
| **Тоо** | Цифр, (Тоо, Цифр) |
| **Илэрхийлэл** | Тоо |
| (Илэрхийлэл, Оператор, Илэрхийлэл) |

Дээрх дүрмээр (4+24)-1 илэрхийллийг мод бүтцэд салгаж –т харуулав.

|  |
| --- |
|  |

Зураг

(4+24)-1 илэрхийллийг мод бүтэцтэй болгов.

Илэрхийллийг мод бүтэцрүү хөрвүүлэхдээ операторуудын хүч ихтэй буюу эхэнд хийгдэх операторуудыг модны гүнд байхаар ижил хүчтэй операторуудын баруун талын оператор илүү гүнд байхаар зохион байгуулна. Хэрэгжүүлэгч модульруу уг мод дамжихад модны орой бүрийг мөчирүүдээс хамааруулан бодолт хийгдэх ба дэд илэрхийлэлүүд дээр операторын үйлдлийг хийсний дараа тухайн илэрхийллийн үр дүн гарна. Утгуудын хооронд үйлдэл хийж байгаа нь **pragmatics** гэх ойлголт юм. Ийнхүү илэрхийллийг хэрэгжүүлэгч програм хангамжийн үндсэн 3-н ойлголтыг жишээн дээр тайлбарлалаа.

Харин програмчлалын хэл хэрэгжүүлэгч нь илүү олон төрлийн мод элементүүдээс тогтоно. Үүнийг -т харуулав.

Хүснэгт

Програмчлалын хэлний мод бүтэц

|  |  |  |
| --- | --- | --- |
| **Мод** | **Модны төрөл** | **Модны бүтэц** |
| Програмын мод | Програмын мод | Коммандын модууд |
| Коммандын мод | Утга олгох коммандын мод, | Хувьсагчын мод  Илэрхийллийн мод |
| Давталтын мод | Илэрхийллийн мод  Коммандын модууд |
| Нөхцөлт шийдвэрын мод | Илэрхийллийн мод  1-р коммандын модууд  2-р коммандын модууд |
| Хэвлэх коммандын мод | Илэрхийллийн мод |
| Унших коммандын мод | Хувьсагчын мод |
| Илэрхийллийн мод | Илэрхийлэл | **Оператор**  1-р илэрхийллийн мод  2-р илэрхийллийн мод |
| Утгын мод | **Утга** |
| Хувьсагчын мод | Хувьсагчын мод | **Хувьсагчийн нэр** |

Эх кодыг мод бүтэцрүү хөрвүүлсний дараа үүнийг хэрэгжүүлэх шаардлагатай. Хэрэгжүүлэлт хийхэд модны үндсээс эхлэн хэрэгжүүлэх бөгөөд програмын мод нь уг эх кодны үндэс нь юм. Програмын мод нь Коммандуудын модуудаас бүтэж байна. Эдгээр коммандуудын хэрэгжүүлэхэд комманд хэрэгжүүлэгч функц хэрэгтэй юм. Комманд хэрэгжүүлэгч функц нь ямар төрлийн мод байхаас хамаараад харилцан өөр өөр ажиллана. Гэх мэтчилэн мод бүрт хэрэгжүүлэгч функц байх бөгөөд харин хувьсагчийн утгыг авах, хувьсагчид утга оноох зэрэг комманданд програмчлалын хэлний өгөгдөл хадгалах зарчмыг ашигладаг.

* + 1. Програмчлалын хэлний өгөгдөл хадгалах зарчим

Програмчлалын хэлний өгөгдөл хадгалалт нь хувьсагч бүрт харгалзан 1 утга байх харгалзаа байдаг. Хэрэв тухайн утга жагсаалт эсвэл бүтэц төрлийг илэрхийлж байвал, өөрөөр хэлбэл 1 зэрэг олон утгыг зааж байвал тэдгээрийг хамааруулах шинэ харгалзааг үүсгэж уг харгалзаанд хаяг харгалзуулж үүнийг тухайн хувьсагчийн утга болгоно. Ингэснээр тухайн хувьсагч 2 хаягаар илэрхийлэгдэнэ гэсэн үг юм. Жишээг **Хүснэгт 5**-т үзүүлэв.

Хүснэгт

С-Төст хэлний санах ойн загвар

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Хаяг | | Харгалзаа | | | | | |
| САНАХ ОЙ | Түлхүүр | A | B | C | D | MASSIV | STRUCT |
| Утга | 10 | 20 | 30 | 40 | ХАЯГ1 | ХАЯГ2 |
| ХАЯГ1 | Түлхүүр | 1 | 2 | 3 | 4 | 5 | 6 |
| Утга | 100 | 200 | 300 |  |  |  |
| ХАЯГ2 | Түлхүүр | A | B |  |  |  |  |
| Утга | 20 | 30 |  |  |  |  |

* “A” хувьсагчид хандах бол САНАХ ОЙ-оос “А” түлхүүртэй элементийг авна. Өөрөөр хэлбэл САНАХ ОЙ -> A гэж илэрхийлэгдэнэ.
* “B” хувьсагчид хандах бол САНАХ ОЙ-оос “B” түлхүүртэй элементийг авна. Өөрөөр хэлбэл САНАХ ОЙ -> B гэж илэрхийлэгдэнэ.
* “MASSIV->1” хувьсагчид хандах бол эхлээд САНАХ ОЙ-оос “MASSIV” түлхүүртэй элементийг авна. Өөрөөр хэлбэл САНАХ ОЙ->MASSIV гэж илэрхийлэгдэнэ. Гэвч энэ нь ХАЯГ1 утгатай байгаа тул ХАЯГ1->1 элементийг авна. Өөрөөр хэлбэл ХАЯГ1 -> 1 гэж илэрхийлэнэ.
* “STRUCT->A” хувьсагчид хандах бол эхлээд САНАХ ОЙ-оос “STRUCT” хаягтай элементийг авна. Өөрөөр хэлбэл “САНАХ ОЙ->STRUCT” гэж илэрхийлэгдэнэ. Гэвч энэ нь ХАЯГ2 утгатай байгаа тул ХАЯГ2->A элементийн авна. Өөрөөр хэлбэл ХАЯГ2->A гэж илэрхийлэнэ.

**САНАХ ОЙ** нь *global* буюу ерөнхий хандалттай хувьсагчуудыг агуулах юм. Өөрөөр хэлбэл програмын кодны аль ч хэсгээс хандаж утгыг авч, утгыг сольж болох юм.

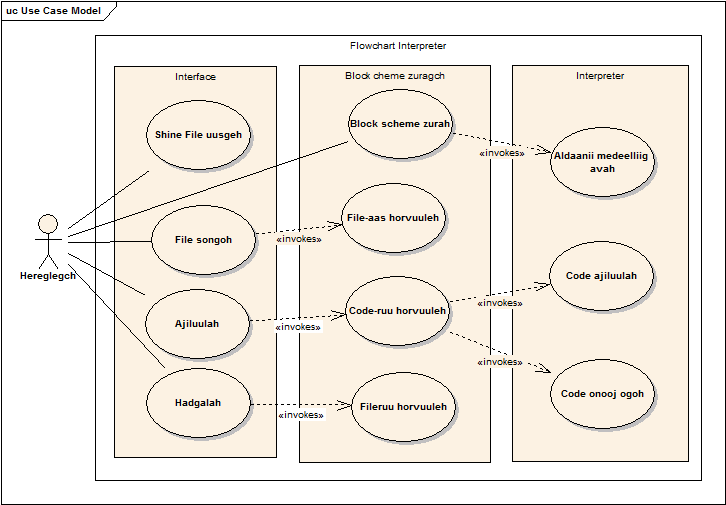
Дээрх байдлаар өгөгдөлд хандаж ажилладаг юм. Иймээс бид өгөгдөл бүрийг 2 утгаар илэрхийлэх тул өгөгдөлд утга оноож өгөгдлийн утгыг авах боломжтой болж байна.

1. Блок схем хэрэгжүүлэгч, шинжилгээ, зохиомж
   1. Програм хангамжийн тодорхойлолт

Блок схем хэрэгжүүлэгч програм хангамж нь блок схемийг зурах, түүнийгээ хэрэгжүүлэх зориулалттай програм хангамж юм. Иймээс дараах 3 модулиас бүтнэ. Үүнд:

* Хэрэглэгчийн интерфейс
* Блок схем зурагч
* Хийсвэр код хэрэгжүүлэгч

Эдгээрийн холбоо хамаарлыг use case диаграмаар -т харуулав



Зураг

Модиулиудын холбоо хамаарлын use case диаграм

Use case диаграмын тайлбарыг хүснэгт -д хийв.

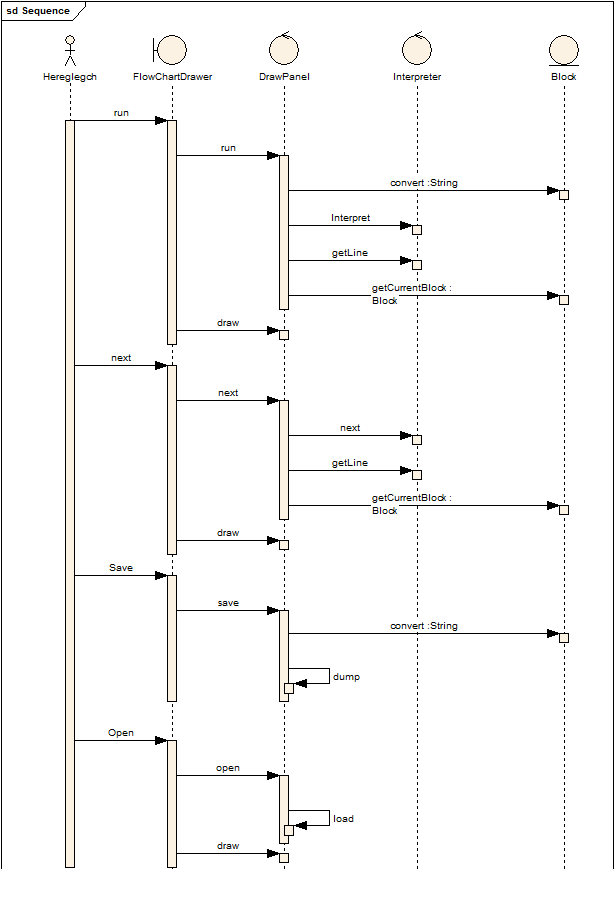
Хүснэгт

Блок схем хэрэгжүүлэгчийн use case диаграм

| **Use Case** | **Тайлбар** |
| --- | --- |
| **User Interface** | |
| Файл сонгох | Зөвхөн блок схем хэрэгжүүлэгч програм хангамж дэмжих файлыг сонгох use case |
| Ажиллуулах | Зурагдсан блок схемийг ажиллуулах use case |
| Шинэ файл үүсгэх | Шинээр блок схем зурах файл үүсгэх use case |
| Хадгалах | Зурагдсан блок схемыг зөвхөн блок схем хэрэгжүүлэгч програм хангамж дэмжих файлыг сонгож хадгалах use case |
| **Блок схем зурагч** | |
| Блок схем зурах | Блок схемийг зурах үйлдлүүдийг багтаасан use case |
| Файлаас хөрвүүлэх | Хадгалагдсан блок схемийг файлаас уншиж зурах use case |
| Файлруу хөрвүүлэх | Зурагдсан блок схемийг файлруу хадгалах use case |
| Хийсвэр кодруу хөрвүүлэх | Зурагдсан блок схемийг файлруу хөрвүүлэх, хөрвүүлэгдсэн мэдээллийг хадгалах use case |
| **Interpreter** | |
| Алдааны мэдээлэл авах use case | Блок схем хэрэгжих явцад гарсан алдааны мэдээллийг авах use case |
| Код ажиллуулах | Оноогдож өгсөн кодыг ажиллуулах use case |
| Код оноож өгөх | Блок схемээс хөрвүүлэгдсэн кодыг оноож өгөх use case |

Эдгээр модулиудыг тус бүрт шинжилгээ зохиомжийг дараах байдлаар хийв.

Sequence диаграмыг -д үзүүлэв.



Зураг

Блок схем хэрэгжүүлэгчийн sequence диаграм

* + 1. Хэрэглэгчийн шаардлагын шинжилгээ

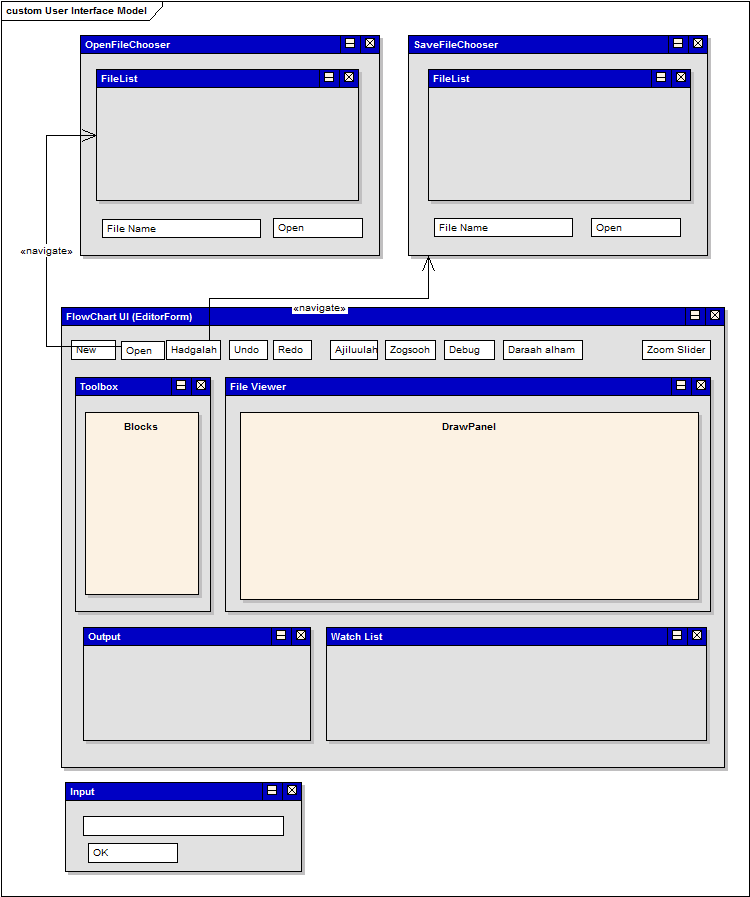
Уг програм хангамж нь алгоритм сонирхогч, анхлан суралцагч нарт зориулагдсан тул хэрэлэгчид компьютерийн зохих мэдлэгтэй хүн байх юм. Мөн програм хангамжийг ашиглахын тулд дараах шаардлагууд тавигдана. Үүнд:

* Блок схемийн талаар тодорхой мэдлэгтэй байх
* Хэрэглэгч блок схемийг програм хангамж дэмжиж буй стандартын дагуу зурах
* Блок схемийг алдаатай зурсан бол алдаагүй болгох
  1. Хэрэглэгчийн интерфейс
     1. Хэрэглэгчийн интерфейсийн шаардлагын шинжилгээ

Хэрэглэгчийн интерфейс нь дараах элементүүдийг агуулсан байх шаардлагатай. Үүнд:

* Товчлуур
  + Шинээр файл үүсгэх
  + Одоо байгаа файлыг хадгалах
  + Файл нээх
  + Хийсэн үйлдлээ буцаах
  + Хийсэн үйлдлээ үргэлжлүүлэх
  + Ажиллуулах
  + Зогсоох
  + Алхам алхамаар ажиллуулах
  + Дараах алхамыг хэрэгжүүлэх
  + Зурах талбарын харагдацын тохируулга
* Талбарууд
  + Toolbox буюу зөөж тавих блокуудын жагсаалт
  + Файлыг нээж үзүүлэх талбар
  + Алдаа болон Гаралтыг харуулах талбар
  + Хувьсагчуудыг харуулах талбар
* Цонх
  + Файл нээх цонх
  + Програм хангамжийн цонх
    1. Хэрэглэгчийн интерфейсийн зохиомж

Шаардлагын шинжилгээнд тавигдсан элементүүдийг –д зохиомжлов.



Зураг

Хэрэглэгчийн интерфейсийн зохиомж

Хэрэглэгчийн интерфейсийн зохиомжийг **Хүснэгт 7**–д тайлбарлав

Хүснэгт

|  |  |
| --- | --- |
| Элемент | Тайлбар |
| New | Шинэ блок схем зурах талбарыг FileViewer-т нээнэ |
| Open | Блок схемийг файлаас нээнэ |
| Хадгалах товчлуур | FileViewer-т идэвхтэй харагдаж буй блок схемийг хадгална. |
| Undo | FileViewer-т идэвхтэй харагдаж буй блок схемд хийсэн сүүлийн өөрчлөлтийг буцаана |
| Redo | FileViewer-т идэвхтэй харагдаж буй блок схемд хийсэн сүүлийн өөрчлөлтийг үргэлжлүүлнэ |
| Ажилуулах | FileViewer-т идэвхтэй харагдаж буй блок схемийг хэрэгжүүлнэ |
| Зогсоох | FileViewer-т идэвхтэй харагдаж буй блок схемийг хэрэгжүүлэлтийг зогсооно |
| Debug | FileViewer-т идэвхтэй харагдаж буй блок схемийг алхам алхмаар ажиллуулна |
| Дараах алхам | FileViewer-т идэвхтэй харагдаж буй блок схемийг алхам алхмаар ажиллуулах явцад дараах алхамруу шилжүүлнэ |
| Zoom slider | FileViewer-т идэвхтэй харагдаж буй блок схемийн харагдацыг томруулж жижгэрүүлэлэх тохируулгыг удирдана |
| Toolbox | Зурж болохуйц блокуудын жагсаалт байна |
| File Viewer | Зурж буй блокыг харуулах талбар юм. |
| Output | Блок схем хэрэгжих явцад хэвлэгдэх гаралтын талбар юм. |
| Watch List | Блок схем хэрэгжилтэд хянаж буй хувьсагчуудыг харуулах талбар |
| Flowchart UI | Програмын цонх |
| OpenFileChooser | Файл нээх товчлуурыг дарахад гарч ирэх цонх |
| SaveFileChooser | Файл хадгалах товчлуурыг дарахад гарч ирэх цонх |
| Input | Блок схем хэрэгжилтэд оролт оруулах шаардлага гарахад гарч ирэх цонх |

* 1. Блок схем зурагч модуль

Өмнөх бүлэгт блок схемийг блокуудын холбоост дараалал гэж үзэж болох санааг дэвшүүлсэн билээ. Энэ нь блокруу нэг сум орж, блокоос нэг сум гарсан гэдгийг тодорхойлж өгж байгаагаараа онцлог байгаа юм. Энэ шинж чанарыг нь ашиглаж блок схем зурж болно. Өөрөөр хэлбэл бид холбоост жагсаалыг таслаад, дунд нь шинэ блок нэмж, тасалсан хоёр блоктой шинэ блокыг холбочихвол блокуудын холбоост жагсаалтанд шинэ блок нэмж чадна. Харин блокын холбоосыг салгаж, холбоостой байсан хоёр блокыг хооронд нь холбовол блокыг холбоост жагсаалаас устгаж чадаж байгаа юм. Үүний жишээг -д харуулав. Тиймээс блок бүрийг хооронд холбох сум, мөн блок гэсэн 2 элементээс бүрдэх блок схемийг байгуулж болно. Харин эхлэлийн блокыг ямар ч блок заахгүй ба төгсгөлийн блок ямар ч блокыг заахгүй гэдгийг анхаарах хэрэгтэй.

|  |
| --- |
|  |

Зураг

Блок зурагч модулийн блок зурах зарчим: нэмэх, устгах

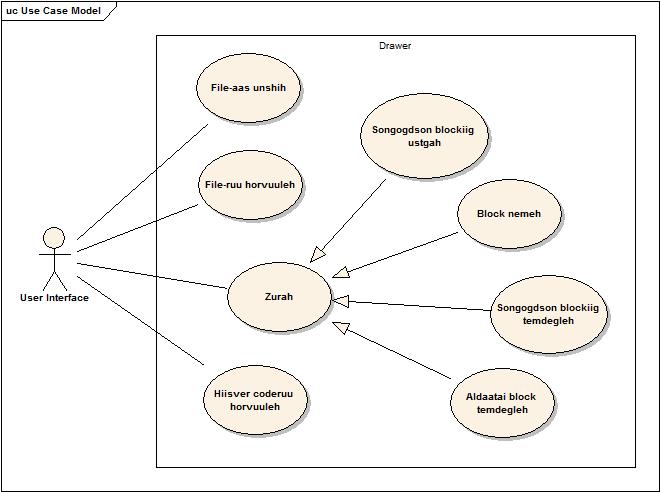
* + 1. Блок схем зурагч модулийн шаардлагын шинжилгээ

Жагсаалтад тухайн элементийн эцэг (parent) элемент гэж тухайн блокыг зааж байга элемэнтийг хэлдэг ба хүү (child) элемент гэж гэж тухайн блок зааж буй элемэнтийг хэлдэг. Блокуудын жагсаалт анхнаасаа эхлэл төгсгөлийн блоктой байх ба дунд нь элемэнт нэмж устгах замаар блок схем зурах юм. Иймээс блок схем нь жагсаалт бүтэцтэй гэсэн үг юм. Аль нэг блокруу хандахын тулд түүний эцэг блок гэх блокоор дамжина.

Иймээс тухайн блок дээр хийх ямар нэг үйлдэл бүр эцэг блокоор дамжих тул блок схемийн блок бүрийн хувьд дараах функцууд байх шаардлагууд тавигдана. Үүнд:

* Блокыг өгөгдсөн координатад зурах
  + Тухайн блокын арын блокыг түүний араас зурах
  + Хүү блокын хувьд зурагдах өндрийг тооцоолох
  + Хүү блокын баруун тийш зурагдах өргөнийг тооцоолох
  + Хүү блокын зүүн тийш зурагдах өргөнийг тооцоолох
* Блок схем сонгогдсон эсэхийг шалгах
  + Хүү блок сонгогдсон эсэхийг шалгах
* Блокийг хийсвэр хэлрүү хөрвүүлэх
  + Хүү блокыг хийсвэр хэлрүү хөрвүүлэх
  + Блок бүр хийсвэр кодны аль хэсэгт хөрвүүлэгдсэнийг тэмдэглэх
* Хүү блокыг устгах
* Хүү блок нэмэх
  + 1. Блок схем зурагч модулийн use case диаграм

Use case диаграмыг **Зураг 9**-д харуулав.



Зураг

Блок схем зурагч модулийн use case диаграм

Хүснэгт

Блок схем зурагч модулийн use case диаграмын тайлбар

| **Use Case** | **Тайлбар** |
| --- | --- |
| Файлаас унших | Файлаас блок схем уншиж блок схемийг зурах use case |
| Файлруу хөрвүүлэх | Зурагдсан блок схемийг файлруу хөрвүүлэх use case |
| Зурах | Хэрэглэгчийн интерфейсээс ирсэн компьютерийн гар, хулганы үйлдэлийн мэдээллүүдийг боловсруулж блок схем зурах use case |
| Хийсвэр кодруу хөрвүүлэх | Зурагдсан блок схемийг хийсвэр кодруу хөрвүүлэх use case |
| Сонгогдсон блокыг тэмдэглэх | Хэрэглэгчийн интерфейсээс ирсэн хулганы үйлдэлийн мэдээллийг хэрэгжүүлж сонгогдсон блокыг тэмдэглэх use case |
| Блок нэмэх | Блок схемд шинэ блок нэмэх use case |
| Сонгогдсон блокыг устгах | Сонгогдсон блокыг устгах use case |
| Алдаатай блокыг тэмдэглэх | Хэрэгжүүлэх явцад алдаа гарвал тухайн алдаа гарсан блокыг тэмдэглэх use case |

* + 1. Блок схем зурагч модулийн оролт, гаралтын зохиомж
       1. Блок схемийг илэрхийлэх файлын зохиомж

Блок схемийг зурагч модулийн тусламжтайгаар “Блок схем хэрэгжүүлэгч” програм хангамж маань блок схемийг файлаас уншиж, файлд хадгалах юм. Учир нь блок схем зурагч нь блок схемийг холбоост жагсаалтаар илэрхийлж байгаа билээ. Иймээс блок схем обьект гэсэн үг юм. Объектыг файлд нэг утгатай харгалзуулж хадгалах олон арга байдаг ба миний хувьд YAML файл форматыг сонголоо. Энэ файл формат нь C, Perl, Python гэж програмчлалын хэлүүдэд өргөнөөр ашиглагддаг бөгөөд XML, JSON форматтай төстэй юм. Өөрөөр хэлбэл обьектыг файлруу хөрвүүлэхээс гадна файлаас обьектрүү хөрвүүлэгдэх боломжтой файлын төрөл юм. Иймээс оролт гаралт ижил файл формат байх юм. YAML файл форматын жишээг үзүүлэв.

|  |
| --- |
| name: John Smith  age: 37  spouse:  name: Jane Smith  age: 25  children:  - name: Jimmy Smith  age: 15  - name: Jenny Smith  age: 12 |

* 1. Хийсвэр код хэрэгжүүлэгч модуль
     1. Хийсвэр код хэрэгжүүлэгчийн шаардлагын шинжилгээ

Блок схем хэрэгжүүлэгчийн хэрэгжүүлэх модуль буюу блок схем зурагч дээр блок схем зурсны дараах үйл ажиллагааг удирдах хэсэг тул дараах шаардлагуудыг хангах ёстой. Үүнд:

* Хийсвэр кодны бүтцийн шаардлагууд:
  + Хувьсагч зарлах шаардлагүйгээр шууд ашигладах
  + Төрлийг илэрхийлэх түлхүүр үг байхгүй байх
  + Давталт, нөхцөл, хэвлэх, гараас утга авах зэрэг коммандуудад түлхүүр үг байх
  + Хувьсагчид зүүн гар талаас утга оноодог байх
  + Илэрхийллийг тоон утгатай хувьсагч болон тоонуудын хооронд хийдэг байх
  + Нэмэх үйлдлийг тэмдэгт мөрүүд болон бусад төрлийн хувьд хийдэг байх
  + Синус, косинус зэрэг хэрэгцээт функцуудыг ашиглаж болдог байх
  + Эх код функц хэлбэрээр бичигддэг байх
  + Файлын нэрээр функцын нэрийг тодорхойлж дууддаг байх
  + Хувьсагчийн нэрний үсэгний том жижиг хамаардаггүй байх
  + Хувьсагчид утга оноох байдлаар массив зарлах
* Хэрэгжүүлэгч модулийн ажиллагааны шаардлагууд:
  + Алдаа гарсан хэсгийг олж, ямар алдаа гарсныг мэдэгддэг байх
  + Ажилж байхад зогсоож болдог байх
  + Алхам алхмаар хянаж, аль хэсгийг хэрэгжүүлж байгааг мэдэж болдог байх
  + Ажилж байх явцад хувьсагчуудын утгыг авж болдог байх
  + Тоонуудын арифметик үйлдлийн хариу зүүн гар талын төрлөө баримталдаг байх.
  + Рекурсив функц ажилладаг байх

Дээрх шаардлагуудыг блок схемийг зурах зарчимаас үндэслэн үүсэгсэн юм. Жишээлбэл блокийн дотор хувьсагч зарлахгүйгээр шууд ашиглаад явах боломжтой юм.

* + 1. Хийсвэр код хэрэгжүүлэгч модулийн use case диаграм

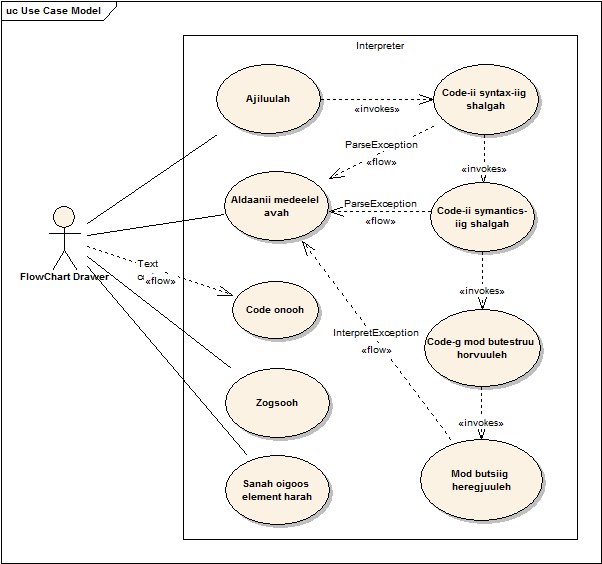
Хийсвэр код хэрэгжүүлэгч нь 2 модулиас бүтэх бөгөөд эдгээр нь өмнөх бүлэгт үзсэн програмчлалын хэл хэрэгжүүлэх аргазүйтэй холбоотойгоор гарч ирсэн юм. Үүнд:

* Хөрвүүлэгч (Parser)
* Хэрэгжүүлэгч (Interpreter)

Хөрвүүлэгч модуль нь хийсвэр код зөв бичигдсэн эсэхийг шалгаад зөв байвал түүнийг “програмын мод”-руу хөрвүүлэх юм.

Хэрэгжүүлэгч модуль нь “програмын мод”-ыг хүлээн авч түүнийг хэрэгжүүлэх үүрэгтэй юм.

Хийсвэр код хэрэгжүүлэгчийн use case диаграмыг -т харуулав.



Зураг

Хийсвэр код хэрэгжүүлэгч болох блок схем зурагч модулийн use case диаграм

Use Case-үүдийн тайлбарыг -т хийв.

Хүснэгт

Хийсвэр код хэрэгжүүлэгч модулийн use case диаграмын тайлбар

| **Use case** | **Тайлбар** |
| --- | --- |
| Код оноох | Хэрэгжүүлэх хийсвэр кодыг тэмдэгт мөр хэлбэрээр оноох use case |
| Ажиллуулах | Хийсвэр кодыг хэрэгжүүлэх use case |
| Алдааны мэдээлэл авах | Хэрэгжүүлэх явцад алдаа гарсан эсэхийг шалгаж гарсан бол алдааны мэдээллийг авах use case |
| Зогсоох | Ажилж байх явцад зогсоох use case |
| Санах ойгоос элемэнт харах | Санах ойгоос элемэнтийн утгыг авах use case |
| Кодны syntax-ийг шалгах | Кодны синтаксыг шалгаж алдаатай байвал алдааны мэдээлэл шалгах use case-рүү алдааны мэдээллийг илгээх use case |
| Кодны symantics-ийг шалгах | Хийсвэр код хэрэгжүүлэх боломжтой эсэхийг шалгаж алдаатай байвал алдааны мэдээлэл шалгах use case-рүү алдааны мэдээллийг илгээх use case |
| Кодыг мод бүтэцрүү хөрвүүлэх | Хийсэр кодыг хэрэгжүүлэх “програмын мод” -руу хөрвүүлэх use case |
| Мод бүтцийг хэрэгжүүлэх | Мод бүтцийг хэрэгжүүлэх use case |
| Зогсоох | Мод бүтцийг хэрэгжүүлэх явцыг зогсоох use case |
| Санах ойтой харилцах | Санах ойтой харилцаж хувьсагчийн утгыг харах use case |

* + 1. Хийсвэр код хөрвүүлэгч модулийн оролтын зохиомж

Хийсвэр код нь коммандуудын дараалал байх бөгөөд дараах төрлийн коммандуудын дараалал байх юм. Үүнд:

Хүснэгт

|  |  |
| --- | --- |
| Комманд | бүтэц |
| Утга олгох | хувьсагч = илэрхийлэл |
| Нөхцөл | if ( илэрхийлэл ) {  коммандууд } else {  коммандууд } |
| Давталт | while ( илэрхийлэл ) {  коммандууд  } |
| Унших | read хувьсагч |
| Хэвлэх | print илэрхийлэл |
| Давталт зогсоох | break |
| Массив зарлалт | хувьсагч = array[ бүхэл хариутай илэрхийлэл ] |

Дээрх коммандуудаар илэрхийлэгдэх хийсвэр код нь дараах бүтэцтэй байх юм.

|  |
| --- |
| function ( хувьсагчууд ) {  коммандууд  } буцах\_утга |

Хийсвэр код хэрэгжүүлэгчийн оролтын жишээг хүснэгт 11-т харуулав.

1. Блок схем хэрэгжүүлэгч програм хангамжийн хөгжүүлэлт
   1. Архитектурын сонголт
      1. Програмчлалын хэлний сонголт

Програм хангамжийг хөгжүүлэх програмчлалын хэлээр “Жава” програмчлалын хэлийг сонгосон юм. Уг програмчлалын хэл дээр бичсэн програм хангамж нь бүх төрлийн Java Virtual Machine дээр ажиллах боломжтой ба Java Virtual Machine олон төрлийн үйлдлийн систем дээр сууж ажиллах боломжтой юм. Уг хэлийг ашигласнаар дараах давуу талуудай.

* Үйлдлийн систем хамаарахгүй ажиллана.
* Бүтээгдэхүүний хэмжээ бага болно.
* Системын нөөцийг бага шаардана.
* Open Source төрлийн програм хангамж болох тул цаашид өөр олон хэрэглэгчид хөгжүүлэх боломжтой.
  + 1. Хөгжүүлэлтийн орчны сонголт

Хөгжүүлэлтийн орчноор Netbeans IDE 7-г сонгосон юм. Netbeans нь Java програмчлалын хэл дээр програм бичихэд зориулагдсан хүчирхэг IDE бөгөөд олон төрлийн хэрэгслүүдээр тоноглож болдог билээ. Мөн төсөл үүсгэж төслийн баримтжуулалт зэргийг хялбараар хийж болдог. Энэ нь Java Standard Edition (SDK 7) – г ашиглаж Java хэлний кодыг хэрэгжүүлж бүтээгдэхүүнийг бэлэн болгодог юм.

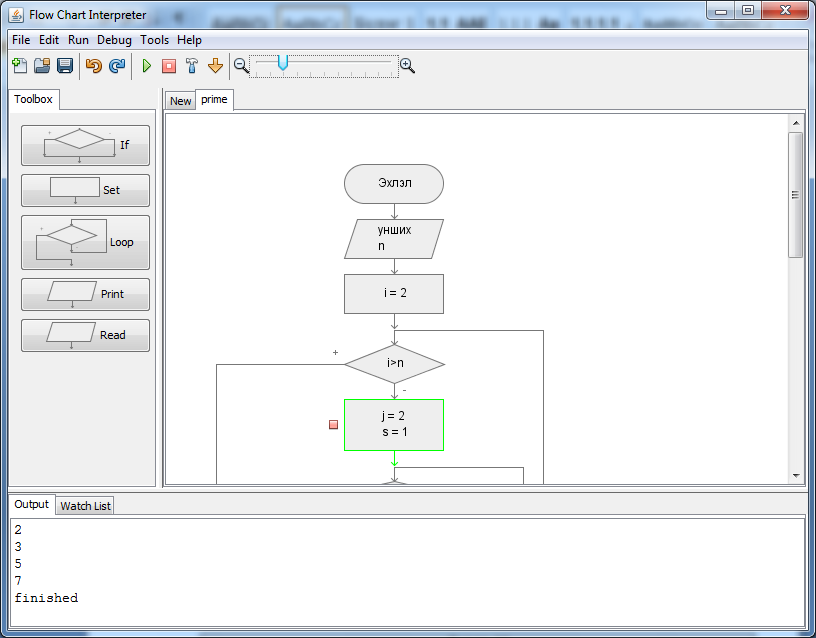
* + 1. Үйлдлийн системийн сонголт

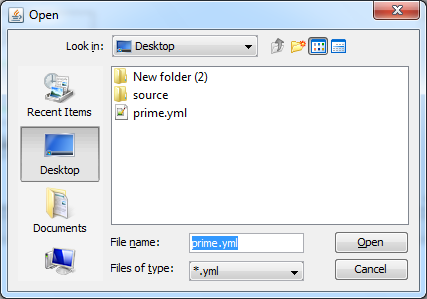
Програм хангамж бэлэн болсны дараа дурын Java Virtual Machine дээр ажиллах ба өөрөөр хэлбэл Java Runtime Environment суусан бүх үйлдлийн систем дээр ажиллах боломжтой юм. Иймээс програм хангамж үйлдлийн системээс хамаарахгүй ажиллана.

* 1. Програм хангамжийн архитектур
     1. Модулиуд

Өмнөх бүлгийн шинжилгээ зохиомжоор 3 модуль байх ёстойг гаргаж ирсэн. Эдгээр модулиудыг дараах байдлаар хэрэгжүүллээ.

* + - 1. Хэрэглэгчийн интерфейс



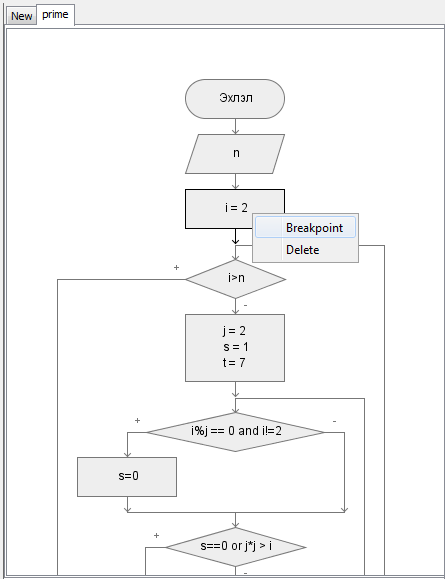


Зураг

Хэрэглэгчийн интерфейсийн хэрэгжилтийн зураг

Хэрэглэгчийн интерфейс нь олон блок схем зурах боломжтой бөгөөд тус бүрд хэрэгжүүлэх боломжтой юм.

* + - 1. Блок схем зурагч модуль



Зураг

Блок схемийн зурагч модулийн хэрэгжилтийн зураг

Блок схем зурагч нь дараах боломжуудтай хөгжүүлэгдлээ.

* Сонгогдсон байгаа блокыг хулганы баруун товчлуур дээр дараад устгах сонголтоор устгана
* Сонгогдсон сум дээр дарж хэрэглэгчийн интерфейсийн toolbox хэсгээс сонгогдсон блокыг нэмнэ.
* Сонгогдсон байгаа блокыг хулганы баруун товчлуур дээр дараад зогсох блог болгон тэмдэглэнэ. Ингэснээр блок схем хэрэгжих явцад уг блокыг хэрэгжүүлэхээс өмнө хэрэгжүүлэлт түр зогсох юм.
* Блок дээр хоёр дарж тухайн блокын өгөгдлийг засна.
  + - 1. Хийсвэр код хэрэгжүүлэгч модуль

Уг хэрэгжүүлэгч модуль нь хийсвэр кодыг хэрэгжүүлдэг бөгөөд дараах боломжуудтай хөгжүүлэгдлээ. Үүнд:

* Хийсвэр кодын файлын нэрээр харгалзуулж функц хэлбэрийн хийсвэр кодыг хэрэгжүүлдэг.
* Хэрэгжүүлэх явцад гарсан алдааны мэдээлэл, алдаа гарсан мөрийн дугаарыг авна.
* Хэрэгжүүлэх явцад тухайн функцын нэрийг авна.  
  Энэ нь рекурс функц ажилж байх явцад аль функц-д хэрэгжиж байгааг мэдэж болно.
* Хэрэгжүүлэх явцад тухайн функцын хэрэгжиж буй мөрийг авна.
* Хэрэгжих явцад хувьсагчдын утгыг харж болно.
* Хэрэгжүүлэлтийг блок бүр дээр түр зогсоох боломжтой ба мөн тухайн сонгогдсон мөрийг хэрэгжихээс өмнө түр зогсоох боломжтой  
  Ингэснээр блок схем хэрэгжүүлэх явцад түр зогсоож хувьсагчын утгыг харах боломжийг олгож байгаа юм.

Жишээ хийсвэр кодын файлыг -д харуулав.

Хүснэгт

Хийсвэр код хэрэгжүүлэгч модулийн дэмжих хийсвэр кодны жишээ.

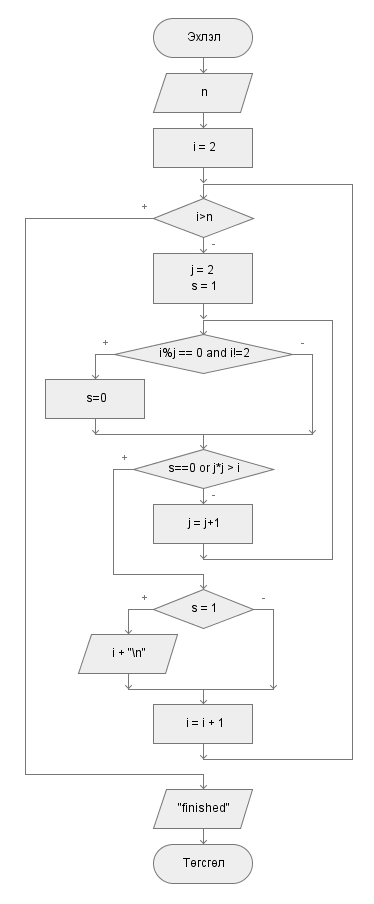
|  |
| --- |
| prime.txt |
| function ( ) {  print !true;  read n;  i = 2;  while ( true ) {  if ( i>n ) { break; }  j=2;  s=1;  while ( true ) {  if ( i%j == 0 && i!=2 ) {  s=0;  } else {  }  if ( s==0 || j\*j > i ) { break; }  j = j+1;  }  if ( s = 1 ) {  print i + "\n";  } else {  }  i = i + 1;  }  } |

Хийсвэр код хэрэгжүүлэгч блок схем зурагчтай холбогдож тухайн блок схемийг хийсвэр кодруу хөрвүүлэхдээ хэддүгээр мөр болж хөрвүүлэгдсэнийг блок бүрд тэмдэглэж аваад уг модулийн хэрэгжүүлэх функцээр дамжин хэрэгжинэ. Хийсвэр код хэрэгжиж байх явцад аль мөрөнд хэрэгжиж буйг аваад блок бүрээр дамжихдаа тухайн блокын хөрвүүлэгдсэн кодны мөр байвал тухайн блокын өнгийг өөрчилж тэмдэглэх замаар хэрэглэгчид харуулна.

* 1. Туршилт
     1. Туршилтын өгөгдөл 1

Блок схем хэрэгжүүлэгч програм хангамжаар өгөгдсөн тоо хүртэлх бүх анхны тоог олох алгоритмыг зохиож туршилтыг өгөгдлийг бэлдэв. Үүнийг **Зураг 13**-т үзүүлэв. Туршилтын алгоритмын ажиллагаанд гараас оруулах утгыг дараах байдлаар өгч -д үр дүнг гаргаж харуулав.

Зураг



Хүснэгт

|  |
| --- |
| Оролт:  50 |
| Гаралт:  2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 |

Дээрх блок схемийг файл хэлбэрээр хадгалж туршсан ба

**Хүснэгт 13**-т текст файл хэлбэрээр хадгалагдсаныг үзүүлсэв.

Хүснэгт

|  |
| --- |
| Prime.yml |
| --- !drawer.entity.block.BeginBlock  nextBlock: !drawer.entity.block.ReadBlock  nextBlock: !drawer.entity.block.SetBlock  nextBlock: !drawer.entity.block.LoopBlock  backBlock: !drawer.entity.block.SetBlock  nextBlock: !drawer.entity.block.LoopBlock  backBlock: !drawer.entity.block.SetBlock  text: "j = j+1 "  initBlock: !drawer.entity.block.IfBlock  leftBlock: !drawer.entity.block.SetBlock  text: "s=0 "  text: "i%j == 0 and i!=2 "  nextBlock: !drawer.entity.block.IfBlock  leftBlock: !drawer.entity.block.PrintBlock  text: "i + \"\\n\" "  nextBlock: !drawer.entity.block.SetBlock  text: "i = i + 1 "  text: "s = 1 "  text: "s==0 or j\*j > i "  text: |  j = 2  s = 1  nextBlock: !drawer.entity.block.PrintBlock  nextBlock: !drawer.entity.block.EndBlock {}  text: "\"finished\" "  text: "i>n "  text: "i = 2 "  text: "n " |

* + 1. Туршилтын өгөгдөл 2

Уг туршилтын өгөгдөл нь рекурсив функцын ажиллагааг шалгаж үзэх зорилготой өгөгдөл юм. Туршилтын өгөгдлийн блок схемийг **Зураг 14**-т харуулав.

|  |  |
| --- | --- |
| Recursive.yml | Factorial.yml |
| C:\Users\LMO\Dropbox\Diplom\Images\Recrusive.png | C:\Users\LMO\Dropbox\Diplom\Images\factorial.png |

Зураг

Дээрх алгоритмын оролт гаралтын өгөгдлийг **хүснэгт 14**-т харуулав.

Хүснэгт

|  |
| --- |
| Оролт:  10 |
| Гаралт:  factorialiin 10-r gishuun = 89 |

Туршилтын өгөгдлүүд зөв үр дүн буцаасан ба рекурсив функцын ажиллагаа факториалыг тооцоолохдоо удаан байсан юм. Учир нь 1 рекурсив функц бүр 2 рекурсийн функцыг дуудаж байгаа. Эдгээр дуудаж байгаа үйлдэл бүр ашиггүй үйлдэл болох өмнө ажиллуулж хариу олсон функцээ ахин ажиллуулж байгаа тул удаан ажиллах эх үүсвэр болж байгаа юм.

* + 1. Туршилтын үр дүн

Туршилтуудын үр дүнд бидний зорилгод нийцсэн үр дүн гарсан юм. Энэ нь програм хангамжийг зөв ажилж буйг гэрчилж байгаа юм.

# Дүгнэлт

Бакалаврын төгсөлтийн ажлаараа “Блок схем хэрэгжүүлэгч” програм хангамж сэдвийг сонгон гүйцэтгэлээ. Уг ажлыг хийж гүйцэтгэснээр програмчлалын хэлний бүтэц, ажиллах зарчмын талаар өргөн хэмжээний ойлголт, түүнийг хэрэгжүүлэх дадлага туршлагатай болсон нь миний хувьд том дэвшил байлаа. Энэ ажилтай ижил төстэй програм хангамжууд байдаг боловч монгол орны хувьд блок схемийн стандарт бусад орныхтой зөрдөгөөс шалтгаалан тэдгээрийг манай оронд ашиглахад хүндрэлтэй юм. Энэ нь манай ажилд давуу тал байгааг харуулж байгаа юм. Мөн програмчлалын хэлний хийсвэр код хэрэгжүүлэгч модуль нь миний хувьд том хэмжээний судалгааны ажил болсон ба дипломын ажлын гүйцэтгэлийн ихэнх кодчилол, судалгааны хэсгийг эзэлсэн юм. Уг ажлын кодчилолыг жава хэлний хүчирхэг хэрэгслүүд болох удамшил, интерфейс, австракт ангиудыг ашиглан уян хатан чанартай бичсэн юм. Тиймээс цаашид хөгжүүлэх бүрэн боломжтой юм. Эдгээрийг дурьдвал:

* Програмын хийсвэр хэлийг хэрэглэгч үүсгэх
* Блок схемийг дизайнжуулах дизайны plug-in-г хөгжүүлэх
* Блок схемийн бүтцийг ашиглан өөр диаграм зурагч програм хангамж хөгжүүлэх
* Олон орны хэлийг дэмжиж ажиллах

Энэ ажлын үр дүнд анх тавьсан зорилгод хүрсэн програм хангамж бий болоод байгаа билээ. Logo, програм хангамжийн нэр зэргийг патентжуулснаар зах зээлд борлуулж худалдаалах боломжтой болох юм.

Мөн open source хэлбэрээр Git, Google Project зэрэг вэб сайтууд дээр төсөл үүсгэн модулиудыг тавьснаар дэлхий даяар хөгжүүлэх боломж нээлттэй болох билээ.

# Ном зүй

1. *Д. Гармаа., “Алгоритмчлал ба Програмчлал - 2”, Улаанбаатар хот, 2009 он*
2. [*http://en.wikipedia.org/wiki/Flowchart*](http://en.wikipedia.org/wiki/Flowchart) *- Flowchart*
3. [*http://upload.wikimedia.org/wikipedia/en/4/4d/Flowchart-template.jpg*](http://upload.wikimedia.org/wikipedia/en/4/4d/Flowchart-template.jpg) *- Flowcharting Template*
4. *David A. Schmidt, “Denotational Semantic: A Methodology For Language Development”, Department of Computing and Information Sciences, 234, Nichols Hall, Kansas State University, Manhattan, KS 66506.* [*schmidt@cis.ksu.edu*](schmidt@cis.ksu.edu)