SE302 – ПХ БҮТЭЭЛТ
Лекц 6

Routine

- Routine гэж юу вэ?
  - Нэгэн зорилготой, бие даасан функц эсвэл процедур
    - Funciton - C++
    - Method - Java
    - Function, sub procedure - Visual Basic
- Сайн чанартай routine гэж юу юэ?
  - Хариулахад хэцүү
  - Сайн биш гэдгийг нь хариулахад хялбар
  - Жишээ нь

```
void HandleStuff( CORP_DATA & inputRec, int crntQtr, EMP_DATA empRec,
   double & estimRevenue, double ytdRevenue, int screenX, int screenY,
   COLOR_TYPE & newColor, COLOR_TYPE & prevColor, StatusType & status,
   int expenseType )
{
int i;
for ( i = 0; i < 100; i++ ) {
   inputRec.revenue[i] = 0;
   inputRec.expense[i] = corpExpense[ crntQtr ][ i ];
   }
UpdateCorpDatabase( empRec );
estimRevenue = ytdRevenue * 4.0 / (double) crntQtr;
newColor = prevColor;
status = SUCCESS;
if ( expenseType == 1 ) {
     for ( i = 0; i < 12; i++ )
           profit[i] = revenue[i] - expense.type1[i];
     }
else if ( expenseType == 2 )  {
          profit[i] = revenue[i] - expense.type2[i];
          }
else if ( expenseType == 3 )
          profit[i] = revenue[i] - expense.type3[i];
          }
```

Example

- Функцын нэр муу
  - HandleStuff()
- Функц нь баримтжуулаагүй
  - Код өөрийгөө баримтжуулах тухай
- Функц нь муу зохион байгуулагдсан
- Оролтын утга өөрчлөгдсөн
  - inputRec
  - Оролтын утгыг өөрчлөхгүй байх хэрэгтэй
  - Үнэхээр өөрчлөх шаардлагатай бол
    - outputRec гэж нэрлэх

Example

- Функц нь глобал хувьсагч уншиж, бичих
  - corpExpense унших, Profit бичих
  - Өөр функцууд магадгүй шууд хандаж байж болно
- Функц нь нэг зорилготой биш
  - Хувьсагчдыг ачааллах
  - Өгөгдлийн сан руу бичих
  - Зарим тооцоолол хийх
- Функц нь буруу өгөгдлөөс өөрийгөө хамгаалаагүй
  - ytdRevenue * 4.0 / (double) crntQtr

Example

- Функц нь шидэт тоонууд ашигласан
  - 100, 4.0, 12, 2, болон 3
- Ашиглагдаагүй параметр
  - screenX, screenY
- Функцын нэг параметр буруу дамжуулагдсан
  - &prevColor
  - Функц дотороос тухайн хувьсагчид өөрчлөлт хийгээгүй
- Хэт их параметр
  - Дээд тал нь 7 байх (11 зохимжгүй)
- Параметрүүдийн эрэмблэлт нь муу

Routine

- Функц нь компьютерийн шинжлэх ухаан дахь гайхалтай зохион бүтээлүүдийн нэг
  - Програмыг хялбар унших
  - Програмыг хялбар ойлгоход
- Мөн
  - Зай хэмнэх
  - Гүйцэтгэлийг сайжруулах
- ОК, Функц гайхалтай гэдгийг угаасаа мэднэ
  - Програм бичихдээ ашигладаг
  - Функц үүсгэх зөв шалтгаан нь юу юэ?
    - Үндсэн шалтгаан нь кодын давхардлаас зайлсхийх

Valid Reasons to Create a Routine

- Төвөгтэй байдлыг бууруулах
  - Хамгийн чухал шалтгаануудын нэг
  - Функц үүсгэсэнээр мэдээллийг нууцлана
    - Үүний талаар бодох шаардлагагүй
    - Функцыг бичих үед л бодно
  - Бусад шалтгаанууд
    - Кодын хэмжээг багасгах
    - Үйлчилгээг сайжруулах
    - Зөв байхыг сайжруулах
  - Функцын хийсвэрлэлгүй бол
    - Комплекс програмыг хүний оюун ухаанаар зохицуулах бараг л боломжгүй

Valid Reasons to Create a Routine

- Ойлгогдохоор хийсвэрлэх

```
if ( node <> NULL ) then
   while ( node.next <> NULL ) do
      node = node.next
      leafName = node.name
   end while
else
   leafName = ""
end if
```

Доорхыг унших

```
leafName = GetLeafName( node )
```

Дээд түвшинд хийсвэрлэх нь 8 мөрөөс илүү

Valid Reasons to Create a Routine

- Кодын давхардлаас зайлсхийх
  - Функц ашиглах хамгийн түгээмэл шалтгаануудын нэг
- Дэд классыг дэмжих
  - Даран тодорхойлоод шинэ код багыг бичих
  - Дэд классад хэрэгжүүлэлтийг хийх явцад алдаа гарах эсдлийг бууруулна

Valid Reasons to Create a Routine

- Дарааллыг нуух
  - Хоёр процессын аль нь түрүүлж ажиллах дарааллыг нуухад тохиромжтой
    - Програм хэрэглэгчийн өгөгдөл болон туслах өгөгдөл авдаг
      - Аль мэдээллийг нь түрүүлж унших ёстой вэ?
    - Стекээс өгөгдөл авах popStack() хувьд
      - Стекийн оройн элементийг унших
      - Стекийн оройн заагчыг нэгээр хорогдуулах

Valid Reasons to Create a Routine

- Гүйцэтгэлийг сайжруулах
  - Олон газар байлахаас нэг газар байгаа кодыг илүү сайн сайжруулж чадна
  - Нэг сайн бичсэн алгоритм, тэр функцыг ашигласан бүх газар ажиллана
- Бүх функц жижиг байх ёстой юу?
  - Үгүй, Зарим ажлууд нэг том функцээр сайн гүйцэтгэгдэх нь бий

Design at the Routine Level

- Функц нь үйлдэлтэйгээ хэрхэн уялдаж байгаа нь чухал
  - cosine() - сайн уялдсан
  - consineAndTan() - муу уялдсан
- Зорилго нь
  - Функц бүр нэг л зүйлийг хийх
  - Өөр юу ч битгий хий
- Функцын уялдаа маш хүчтэй байх ёстой
  - Хамгийн сайн уялдаатай функцууд
    - Нэг үйлдэл, нэг функц
    - sin(), getCustomerName(), eraseFile(), calculateLoanPayment(), ageFromBirthdate()

Design at the Routine Level

- Дараалсан уялдаа
  - Функц доторх үйлдлүүдийг тодорхой дарааллаар гүйцэтгэх
  - Төрсөн өдөр өгөгдөх үед ажилтаны нас болон тэтгэвэрт гарах хугацааг тооцоолох
    - Хэрэв насыг тооцоолоод үр дүнгээс нь
      - Тэтгэвэрт гарах хугацааг тооцоолох
      - Дарааллаар уялдсан / Sequential cohesion
    - Нас болон Тэтгэвэрт гарах хугацааг тус тусад нь тооцоолдог бол
      - Ижилхэн төрсөн өдөр өгөгдлийг ашиглана
      - Мэдээллээр уялдсан / Communicational cohesion

Design at the Routine Level

- Мэдээллийн уялдаа
  - Холбоогүй функцууд ижилхэн өгөгдөл ашиглах
  - Тайлангийн үр дүнг хэвлэдэг функц
    - Тайлангийн өгөгдлийг reinitialize хийдэг функц
- Түр зуурын уялдаа
  - Ижил хугацаанд функц доторх үйлдлүүд зэрэг хийгдэх
    - startUp(), completeNewEmployee(), shutdown()
    - ...

Design at the Routine Level

- Процедур уялдаа / Procedural cohesion
  - Функц дэх үйлдлүүд тодорхой дарааллаар хийгдэнэ
    - Функц нь дараах дарааллаар авна
      - Employee name
      - Address
      - Phone number
- Логик уялдаа / Logical cohesion
  - Ижил функцэд байгаа үйлдлүүд нь дамжин орж ирсэн удирдлагын утгаас хамаарч хийгдэнэ
  - If, case-ийн тусламжтай ялагаатай үйлдлүүд дуудах
    - Ялгаатай орчинд
      - Windows, Linux

Good Routine Names

- Функцын хийдэг бүх зүйлийг тодорхойлох
  - Функцын нэрэнд тодорхойлогдсон байх
    - Бүх гаралт
    - Гаж нөлөө
  - Тайлангын нийт дүнг бодож, гаралтын файл нээдэг бол
    - ComputeReportTotals()
    - ComputeReportTotalsAndOpenOutputFile()
      - Хэтэрхий урт
    - Функц нь өөр нөлөөлөл үзүүлдэг бол
      - Нэр нь бүүр урт болно
    - Шийдэл
      - Богино дүрслэсэн нэр ашиглахгүй байх
      - Давхар нөлөөлөл нь шалтгаан болдог

Good Routine Names

- Утгагүй, тодорхой бус, сул үйл үгээс зайлсхийх
  - Зарим үйл үгүүд ямар ч утгагүй, хэт ерөнхий
    - handleCaluclation(), performServices(), outputUser(), processInput(), dealWithOutput()
    - Юу хийдэг нь мэдэгдэхгүй
  - Илүү дээр
    - handleOutput() - formatAndPrintOutput()

Good Routine Names

- Тоогоор функцын нэрийг ялгахгүй байх
  - Програмистууд заримдаа тоогоор ялгах дугаартай функцын нэр ашигладаг
    - outputUser, outputUser1, outputUser2
  - Энэ нь ялгаатай хийсвэрлэлийг илэрхийлэхгүй
    - Маш муу нэршил
- Функцын нэр хэр урт байх шаардлагатай вэ
  - Судлаачид хамгийн боломжит урт нь
    - 9 - 15 тэмдэгт
    - Хэт урт, хэт богино нэр ойлгомжгүй болдог
    - Илүү ярвигтай функцын хувьд илүү урт байж болно

Good Routine Names

- Функцын нэрэнд буцаах утгыг тодорхойлсон байх
  - Утга буцаадаг бол
    - Буцаах утганд зориулж функцыг нэрлэсэн байх
      - cos()
      - customerId.next()
      - printer.isReady()
      - pen.currentColor()
- Объектын хүчтэй үйл үгээр нэрлэх
  - Сайн функцын нэрүүд
    - printDocument(), calcMontlyRevenue(), checkOrderInfo()
  - Объект хандлагат технологид
    - document.print(), orderInfo.check(), montlyRevenue.calc()

Good Routine Names

- Эсрэг утгатай үг ашиглах
  - Тэгш хэмтэй биш, толгой эргүүлэм
    - FileOpen() - _lclose()

add/remove	increment/decrement	open/close
begin/end	insert/delete	show/hide
create/destroy	lock/unlock	source/target
first/last	min/max	start/stop
get/put	next/previous	up/down
get/set	old/new	

Good Routine Names

- Нийтлэг үйлдэлд тогтсон журам гаргах
  - Нэг төсөлд объект бүрт unique id байна
    - Тогтсон журам гаргахгүй бол
      - employee.id.get()
      - dependent.getId()
      - supervisor()
      - candidate.id()

How to Use Routine Parameters

- Оролт-өөрчлөлт-гаралт дарааллаар параметрүүдийг тавих
  - Параметрүүдийг санамсаргүй эсвэл ц.т дарааллаар эрэмблэхээс илүү
    - Эхлээд оролтын өгөгдөл
    - Оролт гаралт
    - Зөвхөн гаралт
  - Энэ эрэмблэлт нь үйлдлийн утга санааг агуулна
    - Өгөгдлөө оруулна
    - Түүнийг өөрчлөнө
    - Үр дүнгээ буцаана

How to Use Routine Parameters

Ada Example of Parameters in Input-Modify-Output Order

```
procedure InvertMatrix(
   originalMatrix: in Matrix;
   resultMatrix: out Matrix
);
...

procedure ChangeSentenceCase(
   desiredCase: in StringCase;
   sentence: in out Sentence
);
...

procedure PrintPageNumber(
   pageNumber: in Integer;
   status: out StatusType
);
```

How to Use Routine Parameters

- Өөрийн кодонд in, out түлхүүр үг үүсгэх
  - Сүүлийн үеийн хэлнүүд Ada шиг in, out түлхүүр үг дэмждэггүй

C++ Example of Defining Your Own *In* and *Out* Keywords

```
#define IN
#define OUT
void InvertMatrix(
   IN Matrix originalMatrix,
   OUT Matrix *resultMatrix
);
...

void ChangeSentenceCase(
   IN StringCase desiredCase,
   IN OUT Sentence *sentenceToEdit
);
...
```

How to Use Routine Parameters

- Зарим функцууд ижил параметр ашигладаг үед
  - Тэдгээрийг тогтсон дарааллаар ашиглах
    - fprintf()
      - Эхний аргумент нь файлыг илэрхийлдэг
    - printf()
    - fput(), put()
- Бүх параметрүүдийг ашиглах
  - Функцээр параметр дамжуулсан бол түүнийг ашигла
  - Хэрэв ашигладаггүй бол түүнийг устга
    - Ашиглагддаггүй параметр нь алдаа үүсэх магадлалтай хамааралтай байдаг

How to Use Routine Parameters

- Статус болон алдааны хувьсагчийг сүүлд нь тавь
  - Параметрүүдийн хамгийн сүүлд
    - Статус хувьсагч
    - Алдааны хувьсагч
  - Эдгээр нь функцын хоёрдогч зорилго
  - Эдгээр нь зөвхөн гаралтын хувьсагчид
- Оролтын параметрыг ажлын хувьсагч байдлаар ашиглахгүй байх
  - Энэ нь аюултай
    - Оронд нь локал хувьсагч ашиглах

How to Use Routine Parameters

Java Example of Improper Use of Input Parameters

```
int Sample( int inputVal ) {
   inputVal = inputVal * CurrentMultiplier( inputVal );
   inputVal = inputVal + CurrentAdder( inputVal );
   ...
   return inputVal;
}
```

- inputVal - хувьсагчийн тооцогдуулсан
  - Оролтын өгөгдлөө сүүлийн мөр хүртэл хадгалаж чадахгүй
  - Оролтын өгөгдөл дээр тооцоологдсон хувьсагч ашигласан
    - Оролтын өгөгдөл дахин ашиглэх шаардлага гарвал?

How to Use Routine Parameters

- Одоогын болон ирээдүйд үүсэх асуудлаас сэргийлэх зөв хандлага

Java Example of Good Use of Input Parameters

```
int Sample( int inputVal ) {
   int workingVal = inputVal;
   workingVal = workingVal * CurrentMultiplier( workingVal );
   workingVal = workingVal + CurrentAdder( workingVal );
   ...

   ...
   return workingVal;
}
```

How to Use Routine Parameters

- Параметрүүдийн тухай интерфэйсийг баримтжуулах
  - Функцууд руу тогтсон нэг загвараар өгөгдөл дамжуулахын тулд
  - Түүнийг баримтжуулах шаардлагатай
    - Функц бичих хүртэлээ хүлээлгүй, бичиг баримтанд тусгах
  - Баримтжуулах шаардлагатай параметрын тухай
    - Input-only, modified, output-only
    - Параметрийн нэгжүүд (inch, feet, meter)
    - Хэрэв тоочих төрөл ашиглаагүй үед
      - Статус код болон алдааны хувьсагч нь ямар утгыг илэрхийлэх
    - Хүлээх утгуудын завсар, хэзээ ч авахгүй байх онцгой утга

How to Use Routine Parameters

- Функцын параметр хамгийн ихдээ 7 байх
- Параметрийн хувьсагчийн нэрийг зохион байгуулах
  - Input-modify-output
  - I_, m_, o_ гэсэн угтварууд ашиглах

  - Бүтэн утга хүсэж байвал
    - Input_, modify_, output_