**Exception Handling буюу Онцгой тохиолдолыг барьж авах**

**Exceptioin** гэдэг нь програм ажилж байх явцдаа алдаа гарч болох юм. Энэ алдааг барьж авдаг объектийг **exception** гэдэг юм. Харин энэ алдааг засаж боловсруулах, аижллах боломжийг **Exception Handling** бидэнд олгодог юм. Тэгэхээр **exception handling** нь бидний програм дотор гарсан алдааг боловсуулах боломжийг олгодог юм байна. Энэ алдааг боловсруулахад **try, catch, throw, throws, finally** гэсэн 5 түлхүүр үгээр зохион байгуулдаг болно. Точвхондоо **try** блок дотор алдаа гарч болзошгүй кодуудыг хийн бичиж өгнө. Хэрвээ **try** блок дотор алдаа гарвал энэ алдаа шидэгдэх ба энэ алдааг нь харгалзах **catch** блок барьж аван боловсруулдаг. **Try** блокийн дараа ядаж нэг **catch** блок эсвэл **finally** блок байх ёстой.

**Бичигдэх хэлбэр нь:**

*try {*

*// алдаа гарч болох кодууд*

*}*

*catch (ExceptionType1 exOb) {*

*// гарсан алдааг боловсруулах кодууд*

*}*

*catch (ExceptionType2 exOb) {*

*// гарсан алдааг боловсруулах кодууд*

*}*

*// ...*

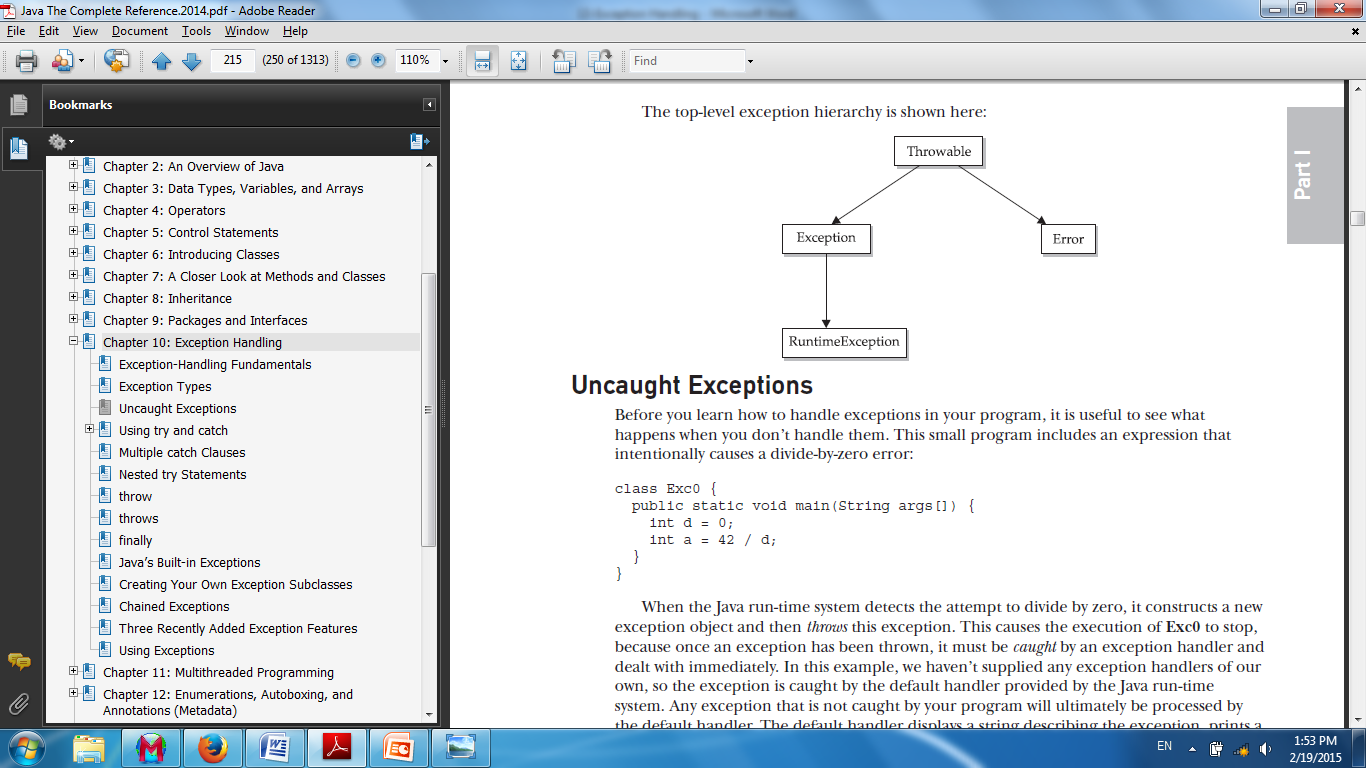
*finally {*

*// try блокийн дараа биелж болох кодууд*

*}*

**Exception –ий төрлүүд**

Бүх exception –үүд бол **Throwable** гэсэн классаас үүсдэг. Тиймээс **Throwable** класс нь бүх exception классуудын эцэг класс юм. Энэ класс нь Exception болон Error гэсэн хоёр хүү класстай. Error нь бүх алдааны, Exception нь бүх онцгой тохиолдлуудын эцэг класс болдог. Бас энд Exception классын хамгийн гол хүү класс болох **RuntimeExcetion** гэж дууддаг класс байна.



**Алдаа гарах програмын жишээ авч үзье.** Доорх кодыг бичээд ажиллуулахад алдаа гарах буюу **exception** шидэж байна

**class** Exc1 {

**static** **void** subroutine() {

**int** d = 0;

**int** a = 10 / d;

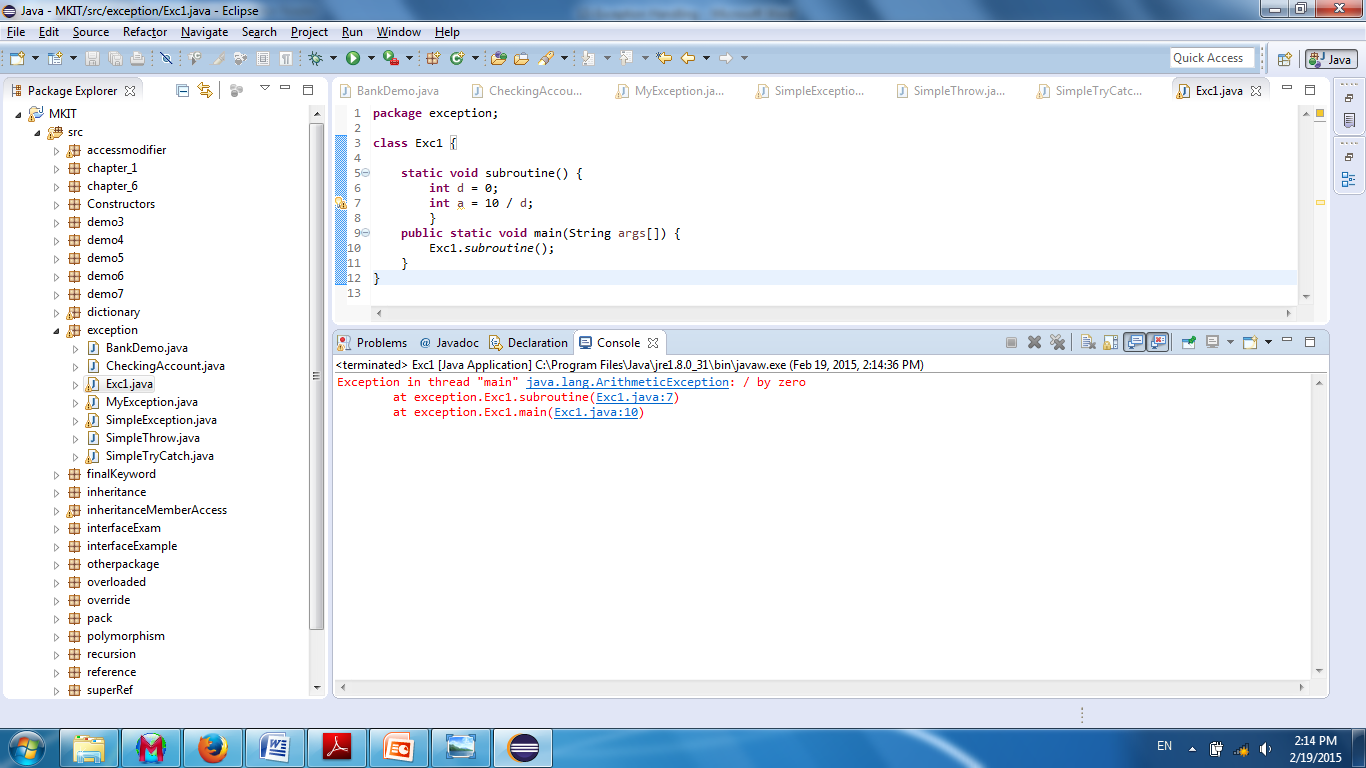
}

**public** **static** **void** main(String args[]) {

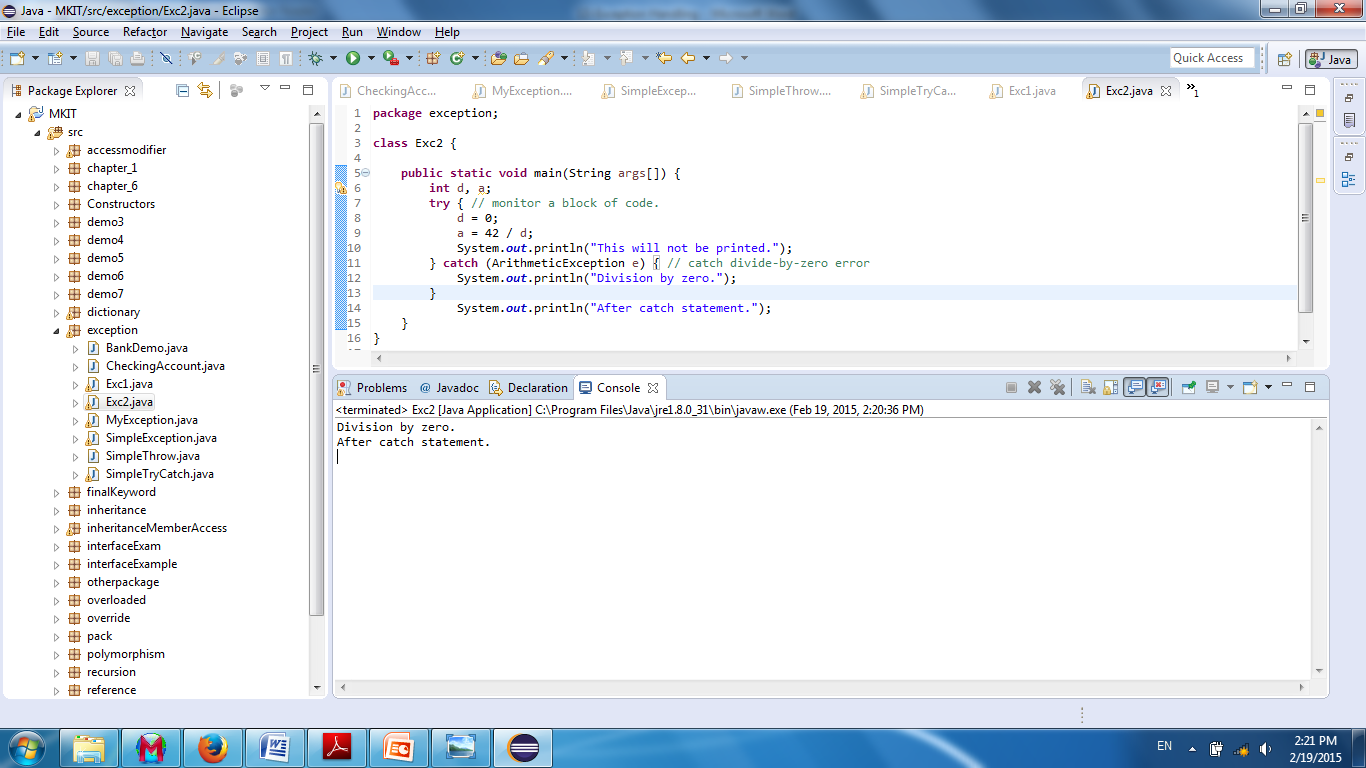
Exc1.*subroutine*();

}

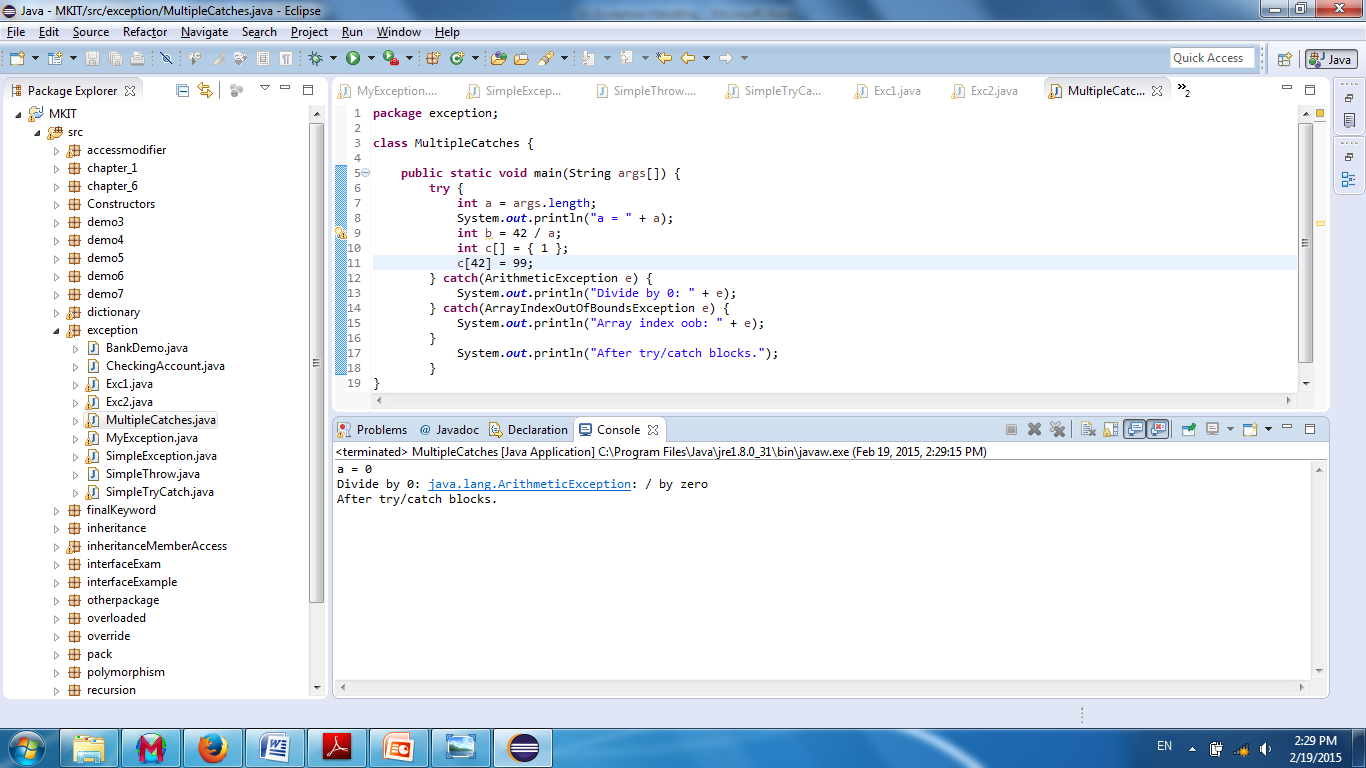
}



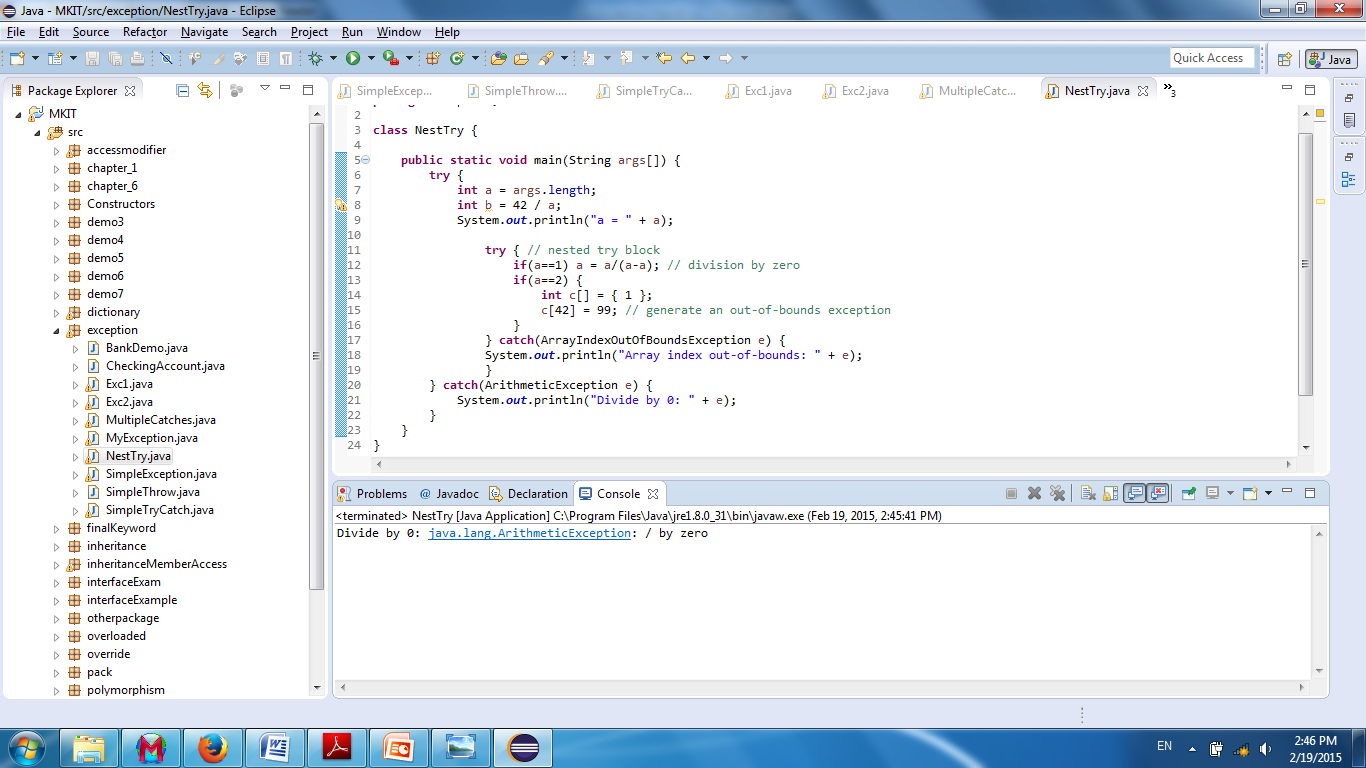
**try болон catch –г хэрэглэснээр:**



Мөн үүнээс гадна олон **catch** блоктой хийж болно. Энэ жишээнд 2 ширхэг catch блок байна. Хэрвээ хамгийн эхний ArithmeticException –ний хэсэг ажилавал дараагийн exception үүсэхгүй юм. Яагаад гэвэл Exception Handling нь exception –г үүсэх үед програмын ажиллагааг зогсоох ба уг алдааг боловсруулах хүртэл дараагийн командуудыг биелүүлэгүйд гол зүйл оршиж байга боно.



**Try блок доторх try блок:** try блок дотор try блок байж болно. Хэрвээ доторх try блок дотор catch блок байхгүй бол дараагийн catch блок руу шилжих юм. Жишээ нь:



**Throw**

Одоог хүртэл та зөвхөн програм ажилж байх явцад шидэх exception –г барьж авч байгаа. Хэдий тийм боловч **throw** –г таны програмд хэрэглэн тодорхой exception –г шидэх боломжтой болгож байгаа юм. Throw –н ерөнхий хэлбэрийг доор үзүүллээ.

throw *ThrowableInstance*

Энд *ThrowableInstance* нь Throwable төрлийн объект эсвэл Throwable –н хүү класс байх ёстой.

**class** ThrowDemo {

**static** **void** demoproc() {

**try** {

**throw** **new** NullPointerException("demo");

} **catch**(NullPointerException e) {

System.***out***.println("Caught inside demoproc.");

**throw** e; // rethrow the exception

}

}

**public** **static** **void** main(String args[]) {

**try** {

*demoproc*();

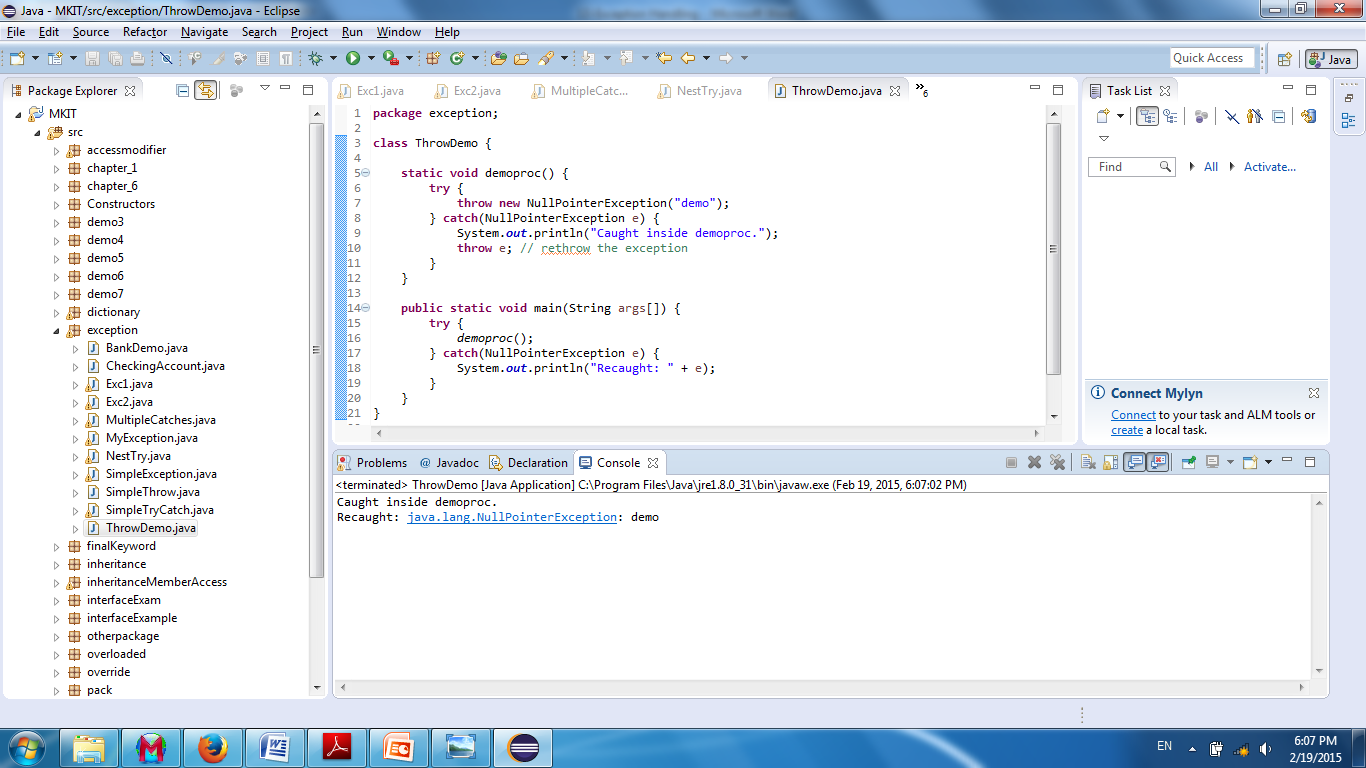
} **catch**(NullPointerException e) {

System.***out***.println("Recaught: " + e);

}

}

}

****

**Throws**

Энэ нь exception –г функцийн урсгал дээрээс барьж авах арга юм. Throws –н ерөнхий хэлбэрийг доор үзүүллээ.

*type method-name(parameter-list) throws exception-list*

*{*

*// body of method*

*}*

**class** ThrowsDemo {

**static** **void** throwOne() **throws** IllegalAccessException {

System.***out***.println("Inside throwOne.");

**throw** **new** IllegalAccessException("demo");

}

**public** **static** **void** main(String args[]) {

**try** {

*throwOne*();

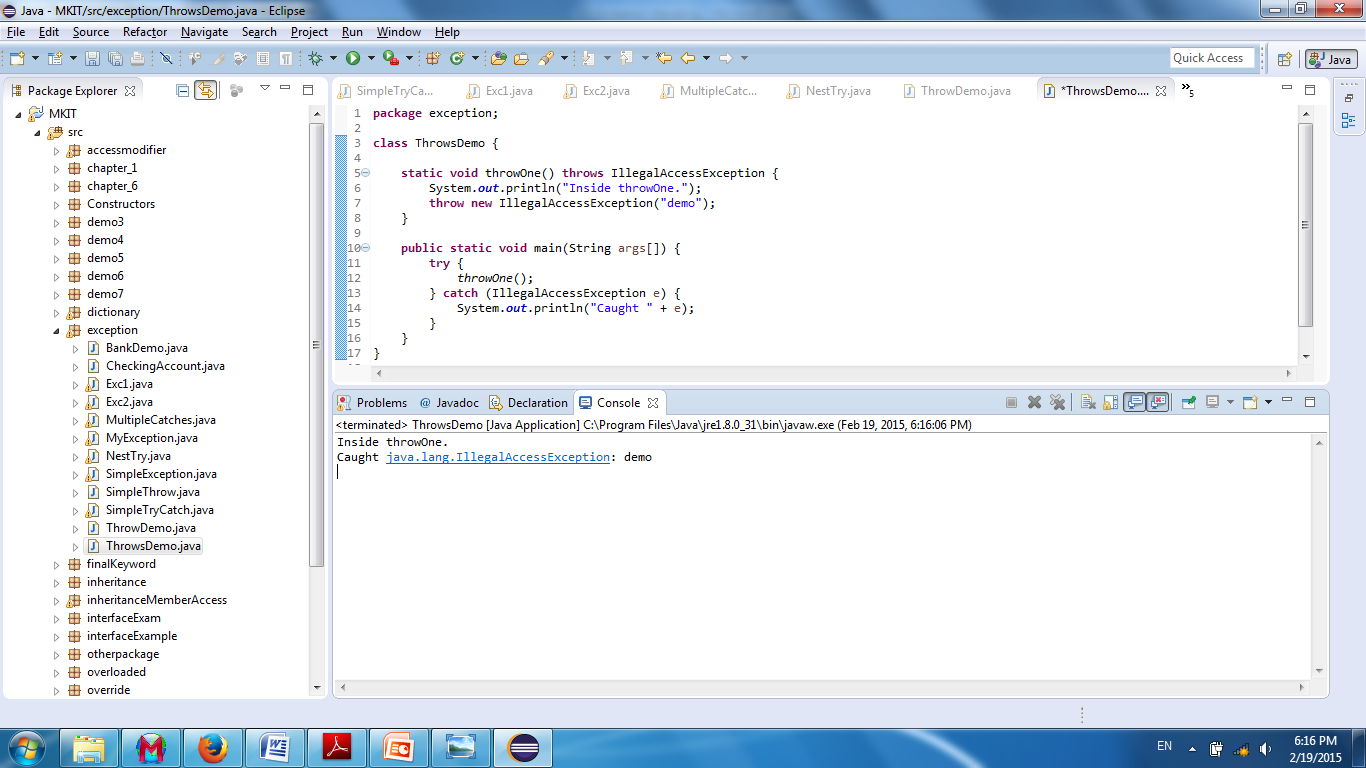
} **catch** (IllegalAccessException e) {

System.***out***.println("Caught " + e);

}

}

}



**finally**

Энэ түлхүүр үг try блокийн дараах кодын блокыг үүсгэхийг ашигладаг. Энэ блок нь exception үүссэн үү үүсээгүй юу гэдгээс үл хамаарч үргэлж биелж байдаг. Жишээ нь аливаа програм ихээхэн хэмжээний мэдээлэлтэй ажилдаг ба түүнтэй ажиллахдаа эхлээд холбогдоод дараа нь програм ажилж байх явцад алдаа гарах үед энэ finally блок дотор баазын холболтыг салгах хэсэг код бичих гэх мэт зүйлүүдийг хийж өгч болох юм.

**Санах зүйл:** Хэрвээ тухайн finally блок нь нэг try блоктой холбоотой байвал тэр try блокийн дараа биелэгдэх болно.

**class** FinallyDemo {

// Throw an exception out of the method.

**static** **void** procA() {

**try** {

System.***out***.println("inside procA");

**throw** **new** RuntimeException("demo");

} **finally** {

System.***out***.println("procA's finally");

}

}

// Return from within a try block.

**static** **void** procB() {

**try** {

System.***out***.println("inside procB");

**return**;

} **finally** {

System.***out***.println("procB's finally");

}

}

// Execute a try block normally.

**static** **void** procC() {

**try** {

System.***out***.println("inside procC");

} **finally** {

System.***out***.println("procC's finally");

}

}

**public** **static** **void** main(String args[]) {

**try** {

*procA*();

} **catch** (Exception e) {

System.***out***.println("Exception caught");

}

*procB*();

*procC*();

}

}

**Үр дүн:**

inside procA

procA's finally

Exception caught

inside procB

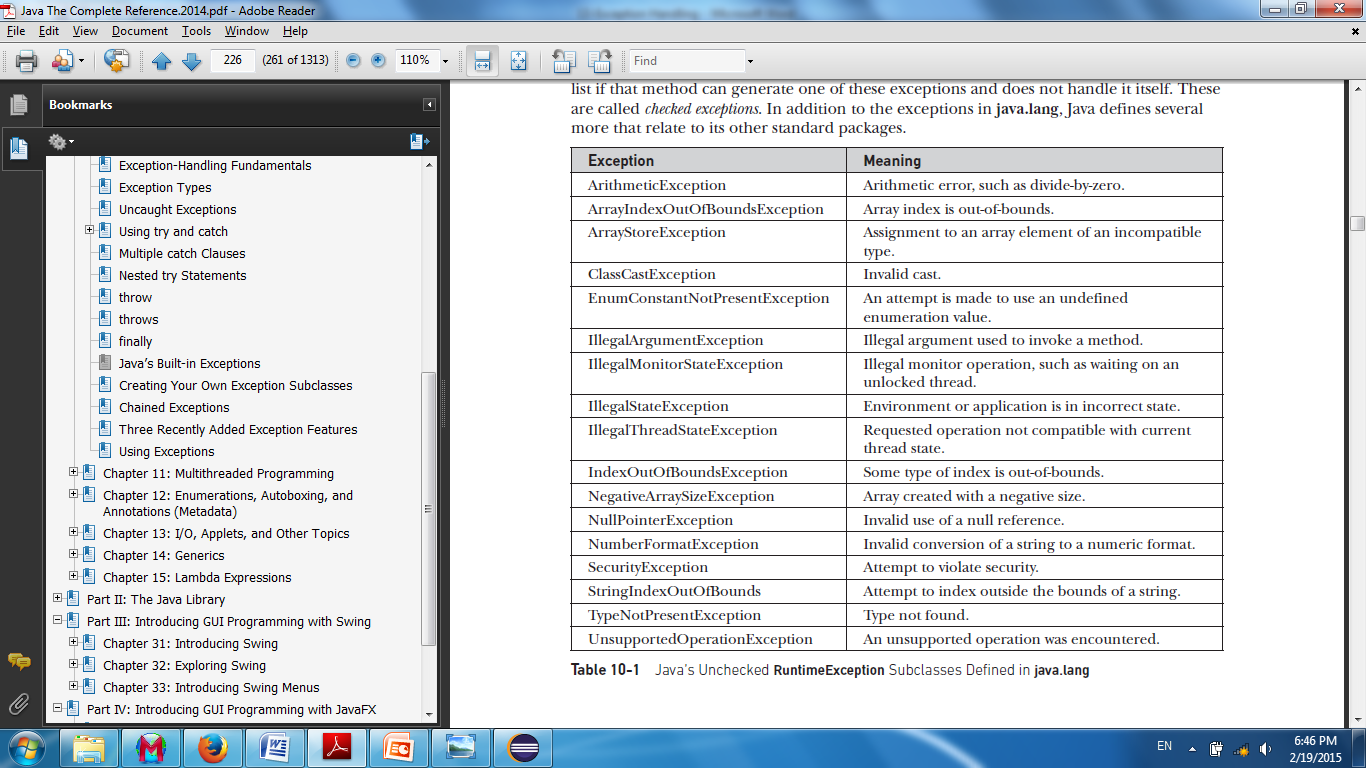
procB's finally

inside procC

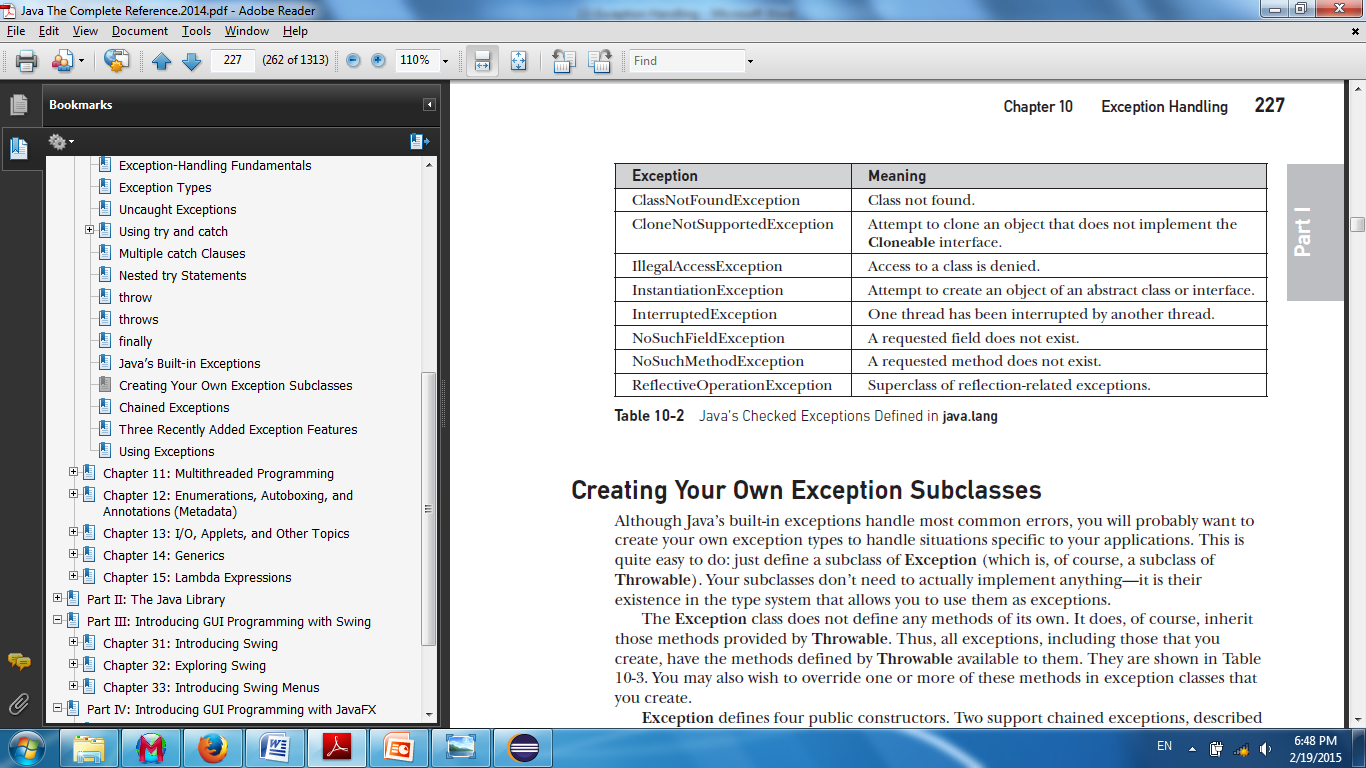
procC's finally

**Жава хэлний үндсэн суурь exception –үүд**

Энэ төрлийн exception -г unchecked exception гэх ба RuntimeException class –аас үүсмэл. Програм нь энэ төрлийн exception –г барьж авах хэрэггүй байдаг буюу алхасах, хэрэгсэхгүй болгож болно гэсэн үг. Та энэ алдаа ямар асуудлаас гарч байгааг тотоогоод өөрийн кодоо засварлана гэсэн үг. Жишээ нь массивын индекс 0 –ээс хэтэрсэн ч гэдэг юм уу.



Энэ төрлийн exception –г checked exception гэх ба урьдаас мэдэх боломжгүй эсвэл хэрэглэгчийн ердийн алдаанаас үүсдэг. Жишээлбэл ямар нэгэн файлыг нээх хэрэгтэй болвол тухайн файл байхгүй тохиолдолд энэ төрлийн exception байдаг. Энэ төрлийн exception –г хялбархан алхсах, хэрэгсэхгүй орхих боломжгүй байдаг. Учир нь энэ төрлийн exception –г зохицуулах ёстой.



**Өөрийн гэсэн Exception –г үүсгэх**

Та энэ төрлийн exception –г үүсгэхдээ та зөвхөн Exception гэдэг классаас удамшуулахад л болно. Жишээ нь доорх код нь энгийн өөрийн exception үүсгэсэн байна.

**public** **class** CheckingAccount {

**private** **double** balance;

**private** **int** number;

**public** CheckingAccount(**int** number) {

**this**.number=number;

}

**public** **void** deposit(**double** amount){

balance+=amount;

}

**public** **void** withdraw(**double** amount) **throws** MyException

{

**if** (amount<=balance) { balance-=amount;}

**else** {**double** needs=amount-balance;

**throw** **new** MyException(needs);}

}

**public** **double** getBalance(){**return** balance;}

**public** **int** getNumber(){**return** number;}

}

**public** **class** MyException **extends** Exception{

**private** **double** amount;

**public** MyException(**double** amount) { **this**.amount = amount; }

**public** **double** getAmount() { **return** amount; }

}

