**Catching exceptions**

Inside the catch block you have the choice of either handling the situation quietly (like printing an error message or showing a warning popup)

**try**{

openFile("somefile.txt");

} **catch**(FileNotFoundException exception) {

*// Handle the situation by letting the user know what happened*

System.out.println("Cannot find that file");

}

OR you can elude the situation and just re-throw the exception:

**try**{

openFile("somefile.txt");

} **catch**(FileNotFoundException exception) {

*// Running away from the responsibility*

**throw** exception;

}

However, re-throwing the exceptions means that whoever is calling "this" method will now have to surround it with another try-catch block and do the same!

**Multiple catch statements**

Since a try block can include more than one statement, and methods can actually throw more than one type of exceptions, you sometimes end up having to cater for different types of exceptions at the same time:

**try**{

openFile("somefile.txt");

array[index]++;

} **catch**(FileNotFoundException exception) {

*// Handle all the possible file-not-found-related issues here*

} **catch**(IndexOutOfBoundsException exception) {

*// Handle all the possible index-out-of-bounds-related issues here*

}

You can have as many catch statements as you need until you cover all possible Exception types that could be thrown inside the try statement.

**Catching all exceptions**

Another option is to simply catch ALL exception types by catching the general type Exception, this means that whatever exception is thrown within this try-catch block, it will be caught and handled in this catch statement

**try**{

openFile("somefile.txt");

array[index]++;

} **catch**(Exception exception) {

*// Handle all the possible exceptions here*

}