Exception Handling

Try-throw-catch

* Try block contains code you are trying to execute
* Can also contain code that throws an exception if something unusual happens, when the exception is thrown the code in the try block stops and the flow of control is transferred to the catch block
* The catch block has one optional parameter that can specify the exception thrown
* All exceptions are subclasses of the base Exception class
* A user can define their own exception classes, they require:
  + A single string argument
  + A getMessage() method
* E.getMessage() returns a string describing the exception, its also the string that was passed as the exception argument

Ex. Of an exception class

public class DivisionByZeroException extends Exception {

public DivisionByZeroException() {

super("Division by Zero!");

}

public DivisionByZeroException(String message) {

super(message);

}

}

* In the constructor, you need to pass to the super constructor the precise kinds of information needed in the catch block when the exception is fired. Above it is “Division by Zero!”

Things you can do with an exception

1. Use e.printStackTrace() to print the name of the exception, description and complete stack trace including the line where the exception occurred
2. Print e.toString() to print the name and the description of the exception
3. Print e.getMessage to print the description of the exception. This is the most used method.

* You can have multiple catch blocks that each executes for a given exception, whichever one matches will be the one executed so order matters.
* Only one exception will ever be thrown inside a try block since the code stops once an exception occurs
* If a method throws an exception and does not catch it, then the method invocation ends immediately.
* Exception handling is an example of **event driven programming**
  + When using event driven programming, objects are defined so that they send events to other objects that handle the events; an event itself is also an object.
  + Sending an event is called **firing and event.**
* It is possible to nest try-catch blocks
  + If a set of try-catch blocks are placed inside a larger try block, and an exception is thrown in the inner try block that is not caught, the exception is thrown to the outer try block for processing and may be caught in one of its catch blocks.
* If multiple catch statements are used make sure to use different variable names for the different exceptions
* A catch block can contain code that throws an exception
  + You may throw the same exception or a new one while handling an exception
* All descendants of the RuntimeException class for example ArrayIndexOutOfBoundsException are unchecked exceptions
  + Runtime exceptions should not be handled – instead fix the code