***RISK: The Conquest Game***

**Advance Programming Practice Project (SOEN 6441)**

**Concordia University - Winter 2019**

CODING CONVENTIONS

# Code Layout

**1.1 Indentation**:

* Tab with four spaces is used for indentation.
* The body of a function/method must be indented with respect to its function header; the body of **for**, **while**, or **switch** statement must be indented with respect to its first line and similarly for **if** statements and other nested structures.

**1.2 Blank Lines**:

* One blank line is used in the following circumstances:

- Between function and method definitions.

- Between major sections of a long and complicated function.

- Between public, protected, and private sections of a class declaration.

- Between class declarations in a file.

**1.3 Blank Spaces**:

* A keyword followed by a parenthesis should be separated by a space.

Ex:

while (true) {

..

}

The expressions in for statement should be separated by blank spaces.

Ex: for (expr1; expr2; expr3)

**1.4 Code Structure**: Java source files have the following ordering:

* Beginning comments

/\*

\* <h2>*Classname</h2>*

\* class description

\*

\* @author

\* @version

\*

\*/

* Package and Import statements; for example:

**package** Controller;

**import** java.util.ArrayList;

**import** java.util.Collections;

**import** java.util.List;

**import** java.util.Scanner;

**1.5 Line Length**:

* Avoid lines longer than 80 characters, since they’re not handled well by many terminals and tools.
* When an expression will not fit on a single line, break it according to these general principles:
* Break after a comma.
* Break before an operator.
* Align the new line with the beginning of the expression at the same level on the previous

line.

**2. Naming Conventions**

The conventions derive from the *‘Camel Case style’* in which a name is formed of multiple words that are joined together as a single word with the first letter of each of the multiple words capitalized so that each word that makes up the name can easily be read.

**2.1 Package**

* Package names are all-lowercase, with consecutive words simply concatenated together (no underscores).

**2.2 Class**

* Class names are written using UpperCamelCase and are generally noun or noun phrases. Interface also follow the same notation of a noun phrase or adjectives.
* Test classes are named starting with the name of the class they are tested, and end with Test.

**2.3 Methods**

* Method names follow lowerCamelCase and are typically verbs or verb phrases. JUnit Test Methods also follow the same notation of lowerCamelCase and generally start with the word 'test' succeeded by the tested method name.

**2.4 Parameter Names**

* Parameter Names follow the lowerCamelCase and one-letter parameter names are typically avoided for public methods.

**2.5 Constants**

* Constants are named with all uppercase letters and may include underscores.

# 3. Comments

Implementation of comments:

* **Single-Line Comments:**

Short comments can appear on a single line indented to the level of the code that follows.

* **End-Of-Line Comments:**

The // comment delimiter begins a comment that continues to the newline. It can comment out a complete line or only a partial line.

* **Documentation Comments:**

Doc comments describe Java classes, interfaces, constructors, methods, and fields. Each doc comment is set inside the comment delimiters /\*\*...\*/, with one comment per API.