CHRISTOPHER NEWPORT UNIVERSITY

DEPARTMENT OF PHYSICS, COMPUTER SCIENCE & ENGINEERING

CPSC 280: INTRODUCTION TO SOFTWARE ENGINEERING

* Assignment 1: The Queen's Gambit *

Instructor: Dr. Roberto A. Flores

**** BEFORE YOU START READING ****

Make sure to record this time (and all times) working on the assignment in the PSP0 logs.

Goal

Practice Personal Software Process Level 0 (PSP0).

Description

Write a program that receives a chessboard as input and indicates whether a king is checked or not– where a king is checked if it is in a square that can be taken by an opponent's piece.

The chessboard is an 8-by-8 board, where empty spaces are represented by dots, white pieces by uppercase letters, and black pieces by lowercase letters. The figure below shows a chessboard with all pieces at the beginning of a game. Black pieces are shown at the top and white pieces at the bottom.

```
rnbqkbnr
pppppppp
........
........
........
........
PPPPPPPP
RNBQKBNR
```

Chess pieces move as follows:

- Pawns (p or P) move straight ahead, one square at a time, and take pieces diagonally.
- Knights (n or N) move resembling an "L", and it is the only piece that can jump over other pieces.
- Bishops (b or B) move any number of squares diagonally.
- Rooks (r or R) move any number of squares vertically or horizontally.
- Queens (q or Q) move any number of squares in any direction (diagonally, horizontally or vertically).
- Kings (k or K) move one square at a time in any direction (diagonally, horizontally or vertically).

The figures below show how these pieces move ('*' indicate where another piece can be taken):

Bishop	King	Knight	Pawn	Queen	Rook
`.......*`	`........`	`........`	`........`	`...*...*`	`...*....`
`*.....*.`	`........`	`........`	`........`	`*..*..*.`	`...*....`
`.*...*..`	`........`	`..*.*...`	`........`	`.*.*.*..`	`...*....`
`..*.*...`	`..***...`	`.*...*..`	`........`	`..***...`	`...*....`
`...b....`	`..*k*...`	`...n....`	`...p....`	`***q****`	`***r****`
`..*.*...`	`..***...`	`.*...*..`	`..*.*...`	`..***...`	`...*....`
`.*...*..`	`........`	`..*.*...`	`........`	`.*.*.*..`	`...*....`
`*.....*.`	`........`	`........`	`........`	`*..*..*.`	`...*....`

The movement of pawns depends on their color. Black pawns take pieces one square diagonally down. White pawns take pieces one square diagonally up. The pawn's figure above shows that a black pawn (lowercase 'p') could take another piece from the immediate square diagonally downwards.

Implementation

Write a class *Chess* with the method below (you can have additional classes and methods if needed).

```
public static char go(File file)
```

This method receives a reference to a text file containing a chessboard, which always contains 8-by-8 characters, each row in one line. Pieces will be arranged in arbitrary configurations. There will be no configuration where both kings are in check or where more than one piece checks a king.

The method should return the piece checking the king (e.g., 'q'), or '-' (dash) if no king is checked.

For example, given the board below, the method returns 'B' (i.e., white bishop is checking black king).

```
"..k....."
"ppp.pppp"
"........"
".R...B.."
"........"
"........"
"PPPPPPPP"
"K......."
```

The method could throw a *FileNotFoundException* if needed.

Initial Files

You are given a file *ChessTest*.java to validate your implementation.

In addition, you're given the Excel file *cpsc280 01a PSP0 Forms.xlsx* containing tabs with Time, Errors and Summary logs to record your PSP0 data.

Submission & Grading

Your program (any "*.java" files) and your PSP0 forms (the Excel file with logs) must be submitted by the due date to the **Gitlab repository given to you by the instructor**.

This repository must be formatted as a **Gradle** project.

- Replace *build.gradle* with the one given to you (in Eclipse, use Gradle >> Refresh… to enable it).
- Replace *.gitignore* with one generated from <http://gitignore.io>. It must be customized to OS X, Windows, Linux, Git, Gradle, Eclipse and Java.
- Gradle's **src/main/java** folder must contain your Java source code and your Excel file.
- Gradle's **src/test/java** folder must contain the Java test file given to you.

Grades are distributed as follows:

- 40% for your PSP0 forms (their data must be complete and consistent with software phases),
- 40% for the correct execution of your program (based on this description and test cases),
- 10% for correct Javadoc for **every public class and method** in your program (any method, field or class that doesn't say public/protected/private is considered public), and
- 10% for correct submission to Gitlab in the format described earlier.

Solutions to test cases must not be hardcoded.

This is an individual assignment. Be reminded that no code should be disclosed to nor received from others (including online sources) as described in the "Honor Code" section in the syllabus.

•••