University of Aberdeen

School of Natural and Computing Sciences

Department of Computing Science

BSc in Computing Science

2023 – 2024

Assessment 1 – Noughts and Crosses

Title: CS2020 – Software Programming	Note: This assessment accounts for 35% of your total mark of the course.

Learning Outcomes

On successful completion of this component a student will have demonstrated competence in the following aspects:

- Ability to write and run basic Java applications
- Ability to judge how Java applications should be structured
- Understanding and ability to apply, techniques to support correct code in a Java application
- The ability to analyse simple problems and design a program solution
- The understanding of software libraries and their application in order to classify them for usage
- The ability to apply programming concepts for solving problems in order to create a new program
- Knowledge and understanding of basic programming concepts and their application,

Information for Plagiarism: Your report and test cases may be submitted for plagiarism, generative AI and collusion check (e.g., Turnitin), and tools in Codio. Please refer to the slides available at MyAberdeen for more information about avoiding plagiarism before you start working on the assessment. Please also read the following information provided by the university on https://www.abdn.ac.uk/students/academic-life/academic-integrity.php

If you decide to use any Generative AI to aid you in your work, then declare this in the comments. As you're still learning Java, we'd advice against this, as it will slow down your learning of the language and its constructs.

Use Git to Save Your Work

While we're not marking your use of Git in this assignment, you are strongly encouraged to do this work inside a Git repository. By doing that you gain the benefit of rolling back your edits, or doing work in branches, and then merging it to the main branch as you accomplish tasks along the way.

Perhaps most importantly, by using Git and pushing your work to a remote repository on GitHub, you provide a safe backup of your work. You'd be surprised by how regularly fellow students discover failed hard disk drives, or that the laptop they're using needs to be returned to its owner, or be repaired due to some liquid being spilt on the keyboard.

If you push your work regularly to GitHub, then you can easily (a) pull it into Codio and carry on if you lose your laptop, and (b) move it to a different device if you need to.

By the way, 'regularly' in this case means making a commit every 5 or 10 minutes locally, and then pushing it to GitHub every hour. Small local commits mean you can roll back small changes without much trouble, and pushing remotely less frequently stores your work safely offsite.

Assessment Details

In this assessment, you will build a simple noughts-and-crosses (tic-tac-toe) game, using the console. See https://en.wikipedia.org/wiki/Tic-tac-toe for information about the game, and https://playtictactoe.org/ for a web-based example (your system will use the console for input and output, it wont be web-based!). We will provide a simple starter system which displays a board on the console and asks the user to choose a square.

Functionality and marking are as described below.

Interaction (15 pts)

- 5 pts: System repeatedly solicits user moves, makes a computer move, and shows the updated board, until the game ends
- 5 pts: System checks user input (selected square) for validity, including that the square is empty; an appropriate error message is produced otherwise.
- 5 pts: Option where computer can make the first move

Win/loss/tie detection (30 pts)

- 10 pts: System detects when either the player or the computer has won the game
- 10 pts: System detects when game is a tie (draw)
- 10 pts: Good collection of JUnit tests for win/loss/tie detection

Computer player (35 pts)

- 5 pts: computer player makes a legal move, ie takes an unoccupied square
- 10 pts: computer player will always make a winning move (complete 3-in-a-row) if this is possible
- 10 pts: if the human player has a potentially wining move, the computer player always blocks this if possible
- 10 pts: Good collection of JUnit tests for computer player

Programming style (20 pts)

- 10 pts: Java code follows good coding style, including good code comments and variable names
- 10 pts: Java code is well structured and decomposed into methods; for example win/loss detection is done using methods that check for a win in a specified row or column.

Submission Instructions

This assessment is due at **5PM** on **Thursday, 2 November 2023**

Please submit in Codio

- Coversheet for the assessment – download, edit, and then upload back to Codio the blank one you find there
- Your code
- A README file in the CA1 folder explaining what you have implemented (text or PDF)
- Mark the assignment as completed in Codio when you're done.

It is your responsibility to ensure that your code runs in Codio. If you do it on another device, and then move it to Codio, you need to ensure it runs there, and that it runs as expected.