COMP110 WORKSHEET 2: FLOWCHARTS AND PSEUDOCODE

Version 5.0
Computing
COMP110

Ed Powley

Introduction

The open-world RPG **Fallout 4** contains a terminal hacking minigame, in which the player must guess a secret n-letter word. In this worksheet, you will model this minigame using a UML activity diagram (which is similar to a flowchart) and a piece of pseudocode.

In the minigame, the player is given a list of possible n-letter words, one of which is the secret word. The player chooses one of the words. The terminal responds by displaying a **likeness** score, defined as the number of letters which match the secret word (i.e. the same letter in the same position). For example if the secret word is HOUSE and the guess is MOUSE, the likeness is 4 out of 5. If the guess is HOPES, the likeness is 2 out of 5 (the letters S and E do not count as they are in the wrong positions). The minigame ends when the player guesses correctly, or after four incorrect guesses.

"I'm gonna run some diagnostics while you're tinkering. Take your time."

— Nick Valentine, Fallout 4

To complete this worksheet:

(a) **Draw** a UML activity diagram for the overall minigame.
(b) **Write** a piece of pseudocode giving an algorithm for playing the game — that is, an algorithm that a human or an AI could use to successfully solve the puzzle.

Note that for part (b), the baseline is that your algorithm would be able to solve the puzzle given enough guesses (so may fail to solve some instances given the limit on number of guesses). Higher marked solutions will avoid making guesses which earlier guesses have ruled out, with top marks reserved for solutions that try to solve the puzzle in the fewest guesses possible.

Submission instructions

Begin by **forking** the base repository for this worksheet, the link to which can be found in the Assessments section on LearningSpace.

Write your **pseudocode** in the README.md file, making appropriate use of Markdown formatting to ensure that it displays properly when viewed through the GitHub web interface. Also upload your **UML activity diagram** as an image, and embed it in the README.md file.

The terminal hacking minigame in *Fallout 4*.

You may use any tool you wish to produce your UML activity diagram, be it a software tool or pen and paper. If you use pen and paper, upload a scan or a photograph of your UML activity diagram, ensuring that the resolution and image quality are sufficient for the UML activity diagram to be easily legible.

Open a **pull request** with your completed work.

Remember that **it is better to submit incomplete work than to submit nothing at all**. Any attempt, even unfinished, at producing a UML activity diagram and a piece of pseudocode will receive a 30% mark.

Marking Rubric

Criterion	Weight	Near Pass	Adequate	Competent	Very Good	Excellent	Outstanding
Basic competency threshold	30%	A reasonable attempt at the worksheet was not submitted by the formative deadline.					
PROCESS: UML activity diagram quality	30%	Diagram is not present. Diagram is very unclear or almost impossible to follow. Presentation is unsatisfactory, with poor or no use of standard symbols.	Diagram is somewhat unclear or difficult to follow. Diagram partially describes the minigame, but with errors or omissions. Presentation is adequate, though usage of standard symbols is lacking.	Diagram is reasonably clear and can be followed. Diagram describes the minigame with some errors or omissions. Presentation is competent, with good usage of standard symbols.	Diagram is somewhat clear and easy to follow. Diagram mostly correctly describes the minigame. Presentation is good, with mostly correct usage of standard symbols.	Diagram is clear and easy to follow. Diagram correctly describes the minigame. Presentation is excellent, with correct usage of standard symbols.	Diagram is very clear and easy to follow. Diagram correctly and comprehensively describes the minigame. Presentation is outstanding, with correct usage of standard symbols.
PROCESS: Pseudocode quality	40%	Pseudocode is not present. Pseudocode is very unclear or almost impossible to follow. Formatting is unclear or inconsistent. The algorithm fails to solve the problem as stated.	Pseudocode is somewhat unclear or difficult to follow. Pseudocode partially describes the algorithm, but with errors, omissions or ambiguities. Formatting is a little unclear or inconsistent. The algorithm makes a reasonable attempt at solving the problem.	Pseudocode is reasonably clear and can be followed. Pseudocode describes the algorithm with some minor errors, omissions or ambiguities. Formatting is somewhat clear and consistent. The algorithm solves the problem mostly correctly.	Pseudocode is somewhat clear and easy to follow. Pseudocode mostly describes the algorithm, with only minor ambiguities. Formatting is mostly clear and consistent. The algorithm solves the problem correctly.	Pseudocode is clear and easy to follow. Pseudocode correctly describes the algorithm. Formatting is clear and consistent throughout. The algorithm solves the problem correctly and somewhat efficiently.	Pseudocode is very clear and easy to follow. Pseudocode correctly and comprehensively describes the algorithm. Formatting is very clear and consistent throughout. The algorithm solves the problem correctly and efficiently.