TARGET CLEAR

A LEVEL AQA

PRE-RELEASE 2025

Programming Tasks (Extension)

Extension 1

The random game has default values of 10 for **MaxNumber** and 50 for **MaxTarget**. Introduce new functionality for levels in the game which adjust these values. Introduce a new menu which allows the user to select from the following options:

Game Mode	MaxNumber	MaxTarget
Easy	6	30
Medium	20	100
Hard	50	1000
Extreme	100	750

Extension 2

Introduce new functionality of “Timed Challenge Mode”. In this mode, the user is given a fixed number of attempts (e.g. 20) to identify all the targets. If the user fails to identify the targets within the given attempts, the game ends, and the final score is displayed. If the user achieves the challenge, award an additional 50 points. Add the necessary input prompts and logic to handle this new game mode.

Extension 3

Modify the application to include two **Targets** lines, enabling a two-player game. Both lines should be shown on the screen at each turn, one above the other, together with the **NumbersAllowed** list. The two players should use the same **NumbersAllowed** list which should operate as normal between each turn. Player 1 should identify targets in **Targets** list 1. Player 2 should identify targets in **Targets** list 2.

A player wins the game by being the first to achieve 20 points. A player loses as normal if one of their targets reaches the first index in their **Targets** list.

Extension 4

Modify the application to include two **NumbersAllowed** lists, enabling a collaborative two-player game. Each player has their own **NumbersAllowed** list. On each turn, each player should enter an expression which can only use values from their own list. This will evaluate to two operands. The players should then enter a third expression which uses these two operands to identify a target. The players must work together to identify targets.

Extension 5

Modify the **CheckIfUserInputEvaluationIsATarget** method to allow a different number of points to be awarded depending on how close the user’s calculation is to a target. Award 4 points if the user hits the target. Award 3 points if the user’s calculation is within 5 of the target and 2 points if the user’s calculation is within 10 of the target.

Extension 6

Modify the **CheckIfUserInputValid** method to allow the user to use parts of the numbers in the **NumbersAllowed** list. For example, in the training game the final number in the list is 512. Allow the user to input this as individual digits 5, 1 or 2 or the values 51 or 12.

Extension 7

Introduce the concept of a Meteor event. A Meteor event has a 33% chance of happening and can only happen once per game. A Meteor event strikes the **Targets** list between two randomly selected indices causing everything to the left of the strike to move two positions to the left and everything to the right of the strike to move two positions to the right. Any targets which are caused to move outside the bounds of the **Targets** list can be deleted. If a Meteor event causes a valid target to move into element zero of the **Targets** list, the game is over.

Extension 8

Introduce a new option to the game to allow an infix expression entered by the user to be evaluated using backtracking instead of converting it to RPN.

Extension 9

Modify the **CheckIfUserInputEvaluationIsATarget** method so that when a target is matched, the number is removed from the **Targets** list and all values to the right are shunted down to fill in the gap.

Extension 10

Introduce a complexity score which is based on the number of values from the **NumbersAllowed** list together with the number of operators used to encourage more complex expressions and risk. Award the player an additional 10 points if they use all the values from the **NumbersAllowed** list and an additional 5 points per operator they use.

Extension 11

Introduce a multi-game mode which displays the average score and number of turns taken over five plays of the game.

Extension 12

Modify the **CheckIfUserInputValid** function to ensure that the user input can also include spaces between numbers and operators. For example, the input “3 + 5 * 2” should be considered valid. Update the regular expression in the function to accommodate this change.

Teacher Notes: *This question was not chosen because it is too easy to “cheat” the answer by using string.split(‘ ’) to split the input on the space and then recombine it back into string without the spaces. Additionally, identifying spaces in a string using regex requires the escape character \s which is beyond the specification.*

Extension 13

The application will only allow the user to enter “complex” mathematical expressions which include both operators and operands. Update the **CheckIfUserInputValid** and the **EvaluateRPN** methods to allow the user to enter a single-digit expression.

