A puzzle, or one-person game, can be represented in ML by two functions:

- a next-state function, which maps a state to a list of possible next states, and
- a wins function, which returns true if the given state counts as a win.

A simple example is a puzzle that has states consisting of positive integers, a next-state function that maps \( n \) to \([n + 2, n + 5]\), and a “wins” function that returns true if \( n = 10 \). We can win if we start from \( n = 2 \) but not from \( n = 7 \).

(a) Code a polymorphic datatype ‘a puzzle, to represent a puzzle by the pair of a next-state function and a wins function. [2 marks]

(b) Briefly contrast depth-first search, breadth-first search and iterative deepening as techniques for solving such puzzles. [6 marks]

(c) Write a function depth that accepts a puzzle, a state and a depth limit. It should use depth-first search to determine whether the puzzle can be solved from the given state within the given depth limit. [6 marks]

(d) Write a function breadth that accepts a puzzle and a state. It should use breadth-first search to determine whether the puzzle can be solved from the given state. [6 marks]

All code must be explained clearly. You may assume that any necessary ML data structures or functions are available.