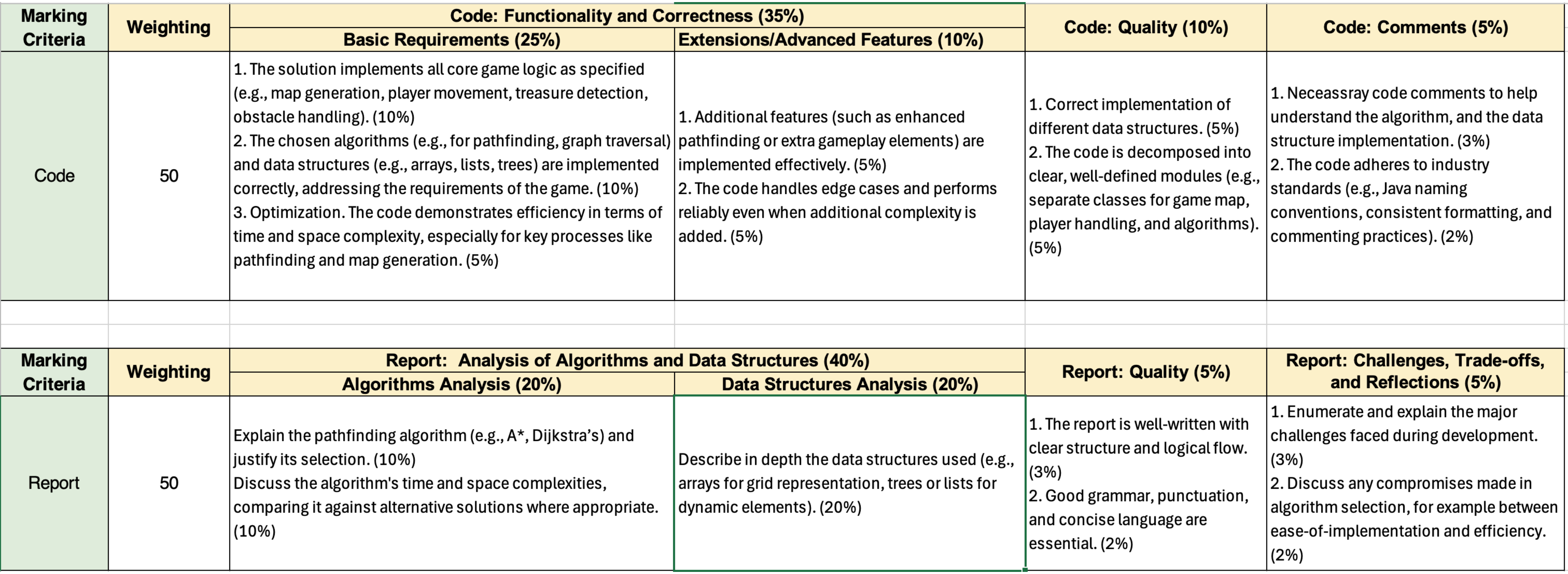

Marking Criteria	Weighting	Code: Functionality and Correctness (35%)		Code: Quality (10%)	Code: Comments (5%)
		Basic Requirements (25%)	Extensions/Advanced Features (10%)		
Code	50	1. The solution implements all core game logic as specified (e.g., map generation, player movement, treasure detection, obstacle handling). (10%) 2. The chosen algorithms (e.g., for pathfinding, graph traversal) and data structures (e.g., arrays, lists, trees) are implemented correctly, addressing the requirements of the game. (10%) 3. Optimization. The code demonstrates efficiency in terms of time and space complexity, especially for key processes like pathfinding and map generation. (5%)	1. Additional features (such as enhanced pathfinding or extra gameplay elements) are implemented effectively. (5%) 2. The code handles edge cases and performs reliably even when additional complexity is added. (5%)	1. Correct implementation of different data structures. (5%) 2. The code is decomposed into clear, well-defined modules (e.g., separate classes for game map, player handling, and algorithms). (5%)	1. Neceassray code comments to help understand the algorithm, and the data structure implementation. (3%) 2. The code adheres to industry standards (e.g., Java naming conventions, consistent formatting, and commenting practices). (2%)

Marking Criteria	Weighting	Report: Analysis of Algorithms and Data Structures (40%)		Report: Quality (5%)	Report: Challenges, Trade-offs, and Reflections (5%)
		Algorithms Analysis (20%)	Data Structures Analysis (20%)		
Report	50	Explain the pathfinding algorithm (e.g., A*, Dijkstra's) and justify its selection. (10%) Discuss the algorithm's time and space complexities, comparing it against alternative solutions where appropriate. (10%)	Describe in depth the data structures used (e.g., arrays for grid representation, trees or lists for dynamic elements). (20%)	1. The report is well-written with clear structure and logical flow. (3%) 2. Good grammar, punctuation, and concise language are essential. (2%)	1. Enumerate and explain the major challenges faced during development. (3%) 2. Discuss any compromises made in algorithm selection, for example between ease-of-implementation and efficiency. (2%)