School of Computing

FACULTY OF ENGINEERING AND PHYSICAL SCIENCE

UNIVERSITY OF LEEDS

Final Report

Investigating Artificial Intelligence for Playing 'Speed Clue', a Stochastic Game involving Logical Inference

Katie Mitchell

Submitted in accordance with the requirements for the degree of BSc Computer Science

2019/2020

40 credits

The candidate confirms that the following have been submitted:

Items	Format	Recipient(s) and Date
Deliverable 1 - Project Report	*Report PDF*	*Minerva (11/05/20)*
Deliverable 2 – Software Code	*Gitlab Repository -* *https://gitlab.com/sc16km/comp3931-individual-project*	*Supervisor and assessor (11/05/20)*

Type of Project: Exploratory Software

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

K. Mitchell

Summary

This project considers the possibilities of Artificial Intelligence within the game of 'Speed Clue', which is a variation on 'Cluedo' that focusses on the logical inferences and ignores the movement around the board aspect. This game has been selected because it's properties are different to those that have already been extensively researched such as Chess, therefore, studying 'Speed Clue' has the potential to reveal new insights.

Due to the complex nature of the logical deductions involved in the game which vary depending on the situation, it is likely that an artificial player will need to be dynamic. Therefore, an aim of this project is to document a range of possible strategies in detail which are not necessarily mutually exclusive. This will mean that in future research, multiple approaches could be combined to create an artificial player with a mixed strategy approach.

In this project, I have also implemented a platform that allows a range of players to participate in a game of 'Speed Clue' which can involve random or pre-defined deals for effective testing. This software ensures the rules are always adhered to and is designed dynamically to allow the types of players to be interchanged; currently it can handle a basic and an advanced artificial player as well as a human participant. The aim was to develop the basic and advanced artificial players so that they can beat a human opponent, with the advanced strategy being more successful. The results from testing these players should allow for comparisons with the alternative strategies that are planned but not implemented within this project.

Acknowledgements

Firstly, I would like to thank my supervisor Dr Brandon Bennett for his support and guidance throughout the project, particularly for going above and beyond to find resources and information on topics I was unfamiliar with.

Secondly, I would like to thank everyone who played the many games of Cluedo and Speed Clue with me at the beginning of this project so I could test the various strategies, their patience and support was invaluable.

Finally, I would like to thank Dr Natasha Shakhlevich whose feedback from the interim report and during the progress meeting helped guide the focus of this project into something I was even more passionate about.

Table of Contents

Chapter 1 - Introduction

Artificial Intelligence (AI) has been a significant field of study in Computer Science since the development of the modern computer which highlighted the question of whether human intelligence could be matched or surpassed by technology. The term ‘Artificial Intelligence’ was first adopted at the 1956 Dartmouth Conference [1, p3] where Computer Scientists *Herbert Simon* and *Alan Newell* presented the *Logic Theorist*; a program designed to prove theorems from *Principia Mathematica* which is recognised as the first Artificial Intelligence program. However, even before the field of Artificial Intelligence was truly acknowledged, researchers such as *Claude Shannon, Alan Turing* and *Arthur Samuel* were considering how computers might play strategy-based board games such as *Chess* and *Checkers* [2].

The research into techniques for *deterministic* board games with *perfect information* has been extensive and success in this field dates back to 1952 when A. S. Douglas first produced software that mastered the game of *Tic-Tac-Toe* [2]. Human champions in *Chess*, *Checkers* and *Go* have all been defeated by AI programs through the development of high-quality search-intensive techniques and advancements in the field of *Deep Learning*. The methods that succeeded for these *deterministic* games require *perfect knowledge*; therefore, they are less successful for games such as *Poker*, *Scrabble* and *Monopoly* which involve *chance* and *imperfect information*.

1.1 Project Aims and Objectives

The aim of this project is to explore the possibilities of Artificial Intelligence to investigate strategies for the game of *‘Speed Clue’,* (discussed in *Section 2.3.1*) which is a variation of ‘*Cluedo’* that only uses the cards and removes the board, therefore, it focuses on the logic not the movement aspect of the game. Furthermore, in this game *imperfect information* and *inference* play major roles and these characteristics are different to those in games which have been extensively researched such as *Chess* or *Checkers*. As such, one of the main aims of this project is to analyse and design a variety of existing and potential strategies, some of which will be compared in this project and others may form the basis of future research projects. These aims are broken down into the following objectives:

1. Investigate and design a variety of strategies that the artificial player can use with the aim of winning the game.
2. Produce a playable implementation of the game ‘*Speed Clue’* that allows for artificially intelligent and human players.
3. Implement and compare at least two of the strategies found from objective 1, test them against each other and humans.

1.2 Deliverables

This report is one of the deliverables for this project and will include:

- A detailed literature review on *Game Theory*, Artificial Intelligence techniques and researched '*Speed Clue'* strategies.
- The design of a variety of potential strategies which can be used in future projects.
- The design process, implementation and evaluation of the software produced.
- Comparisons of the success of the different strategies used by the artificially intelligent player against other artificial players and humans.

The second deliverable will be the prototype software which will implement the artificial agent capable of playing '*Speed Clue'* with different strategies. Several versions of the software will be produced to meet each of the key milestones at the end of each of the stages and these will be evaluated and documented at the end of the report. Each milestone is considered as the successful completion of the objectives in *Section 1.1* which have to be completed chronologically.

1.3 Methodology

During semester 1, the focus for the project was performing a background literature review to guide the decision on the project aims and objectives. As the decision was made to not begin the software development until after the intermediate report had been produced the plan for the semester was less formal than the one described below for semester 2. Estimates could not be made on the length of time background research into the various sections described in *Chapter 2* would take because it was unclear how many research papers would need to be analysed to gain enough information. At the end of semester 1 the Gantt chart in *Appendix D* was produced based on the original aims, however, after further consideration the objectives have been amended, and a new project plan was rapidly developed during the first week.

The timeline for this project in semester 2, as shown in the Gantt chart in figure 1, is based on four key stages with the continuous stage being completed throughout the project in parallel with stages 1, 2 and 3 that occur consecutively. The continuous stage concentrates on researching, writing and editing the main report which should not be left until the end of the project as it is important to document how the results of earlier software versions may affect the direction and focus of the project.

As this is an exploratory software project, the software that will be produced will be used as a research tool to explore the possibilities of Artificial Intelligence and compare strategies for *'Speed Clue'* therefore, it does not need to be of a commercial standard. Furthermore, for each software version the focus is on delivering the working code quickly to allow for more experiments and development not on error checking or the user interface.

At the end of each stage an objective from *Section 1.1* will be accomplished, the numbered stages in *Figure 1.1* correspond to objectives 1,2 and 3 respectively and achieving each is a milestone for the project. The first stage will involve researching existing strategies for '*Speed Clue'* and designing some new alternatives. The aim of the second stage is to develop a basic system that implements the rules of *'Speed Clue'* and outlines the play through of a game, without specifying a strategy for the player. Once this stage has been successfully completed, the third stage can begin which involves implementing at least three types of players. The first artificial player will use a basic strategy which can be used as a baseline to compare against an alternative strategy and finally the software should be extended to allow a human player to participate. During this stage we will test and compare the strategies against each other and the human player.

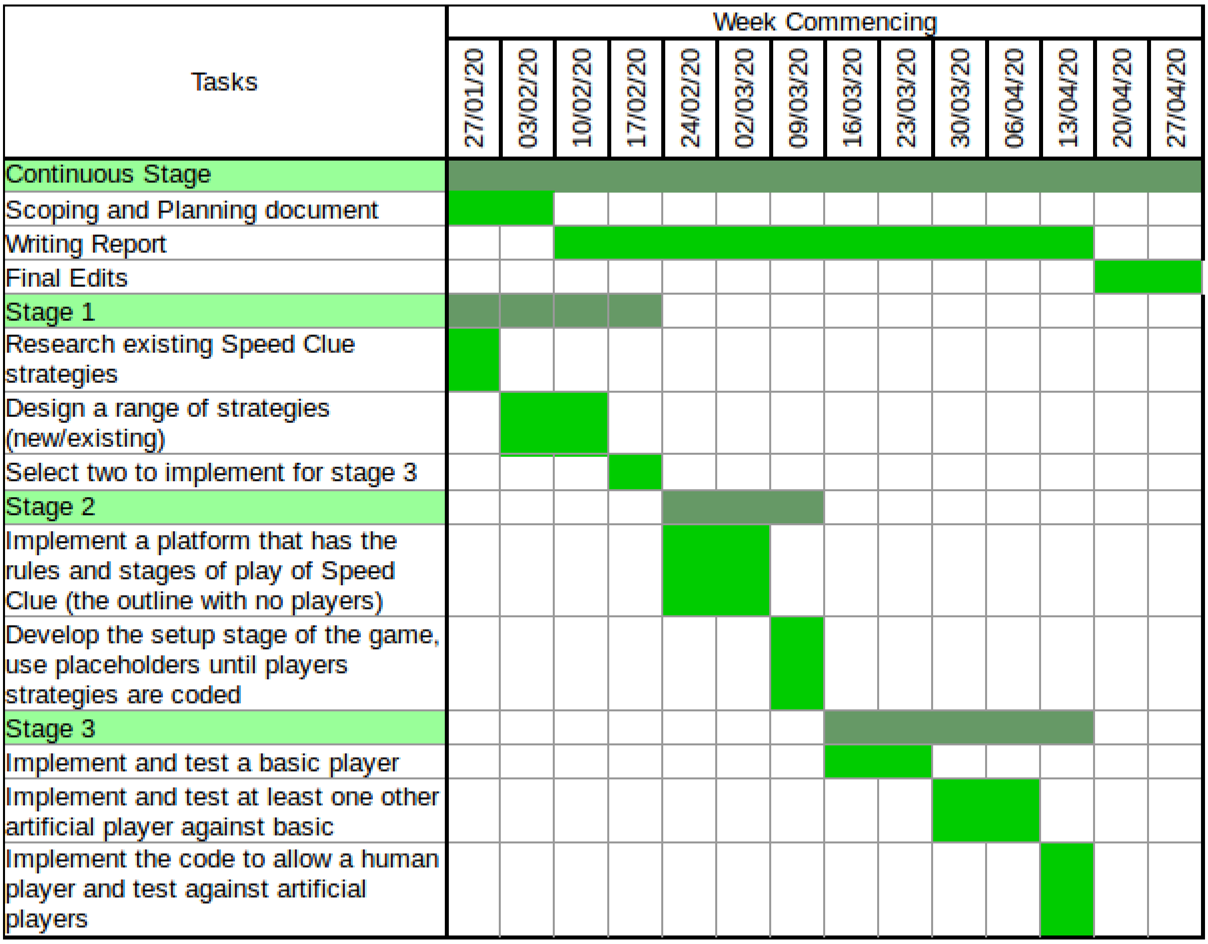

Figure 1.1 Gantt chart representing the project timeline for semester 2 with focus on accomplishing objectives at the end of each stage.

1.4 Risk Mitigation Strategy

This is an exploratory software project and as such there is no risk related to not delivering a production ready system, however, in order to test multiple strategies, I will need to deliver a minimum viable product which is in stage 2. As the focus on this project is on Artificial Intelligence techniques with regards to logical inference, the strategies can be tested on '*Speed Clue'* which is a variation of *'Cluedo'* that involves the cards but not the movement around the board aspect, implementing the rules for this is simpler which means there is little risk involved in creating the basic system. Also, a key aim of this project is to generate the strategies so they can be used in future research, therefore, if all the planned software is not implemented then it can be continued in future projects.

1.5 Legal, Social, Ethical and Professional Issues

Although the field of Artificial Intelligence overall has a large number of ethical issues, very few are considered relevant when considering the applications within board games. The problems of Artificial Intelligence making people redundant in the workforce or making decisions that could be life threatening, for example with self-driving cars (and the further complications regarding this about who should be held responsible) do not apply to this project. However, any project in this field has the possibility to impact or encourage the research into super intelligent AI, which if successful, could have either positive or disastrous consequences.

As part of the initial plan involves human user testing this will be an ethical issue that needs to be considered. The informed consent of all participants will need to be obtained and recorded and any vulnerable or potentially compromised subjects will need to be given due consideration.

As this is an exploratory software project and no software will be deployed, there is no stakeholder, therefore, there are no professional issues to be discussed. Likewise, if this software was released there could be legal issues related to copyright infringement of using the names and logos associated with the board game. However, research suggests that the idea of the game mechanics of '*Speed Clue'* which is used in this project are not copyrighted and the external materials used as the basis for the design of the game are recorded in *Appendix A* which covers any legal issues. Finally, the overall issues of Artificial Intelligence in society such as the loss of jobs are important to consider, however, the research in this project does not have any direct social issues. This report considers a very specific topic of artificial intelligence with relation to board games which does not impact society in general.

Chapter 2 - Background and Related Research

Background research into both *Game Theory* and Artificial Intelligence techniques is required in order to gain a deep understanding of the aim of this project and which methods could be used to successfully complete the objectives (refer to *Section 1.1*). The knowledge gained from this literature review is then used to evaluate which existing strategies should be used for the artificial player by analysing and classifying the nature of the game *'Speed Clue'*.

2.1 Game Theory

Game Theory is the study of strategic decision making in situations where the outcomes are dependent on the interacting decisions of all the other contributors involved [4]. The practical applications of *Game Theory* today range from biologists modelling natural selection to political scientists using the techniques to analyse voting [5], however, arguably it is most significant in the field of economics.

2.1.1 Defining a Game

In order to investigate the types and features of games, we need to define the concept of a game and what the components are. This will prevent any confusion as the word '*game'* is often used to refer to an activity in which fixed rules are followed to attempt to solve a puzzle or win against an opponent, to indicate one particular occasion on which a game is played or to specify a player's degree of skill [6]. The definitions that I will be using are based on those found in a modern text [3] which takes inspiration from the works of *John von Neumann* and *Oskar Morgensten* who are considered responsible for inventing the mathematical theory of games [4].

- A **game** is described by its own set of rules.
- A **play** is any instance of a *game*.
- A **state** represents a unique configuration of all components within a *game* for each *play*.
- A **move** (also known as an action) is a decision about which choice to make in a state.
- A **choice** represents the particular *move* chosen in the play.
- A **strategy** is a plan that the player can use to choose which *move* to make in every possible *state*.
- An **outcome** is the result of a *move*.
- A **payoff** is the reward (positive or negative) that is a consequence of the *outcome*.
- A **rational player** is one that tries to optimise their individual *payoffs* (whilst being aware that the other players are also doing the same), has preferences and beliefs about the world.

2.1.2 Game Types

2.1.2.1 Zero-Sum and Non-Zero Sum Games

In *Game Theory* a *zero-sum game*, the main property is that the sum of the *payoffs* for all of the player's equates to zero, furthermore, in order for one player to have a positive *payoff*, the other player(s) must receive a negative *payoff* [7]. *Chess* and *Checkers* are examples of two-player *zero-sum games* as the *payoff* corresponds to whether they win (+1), draw (0) or lose (-1) the game. As there are only two players in these games, there can only be one winner (unless there is a draw) and this means that the total *payoffs* from both players equals zero. Real-world games rarely fit into the category of *zero-sum* as in most games one player's gain does not necessarily cause an equivalent loss to another player, these games are classified as *non-zero-sum games* [3].

2.1.2.2 N-Player Games

Games and the strategies used to achieve the overall objective (which is usually to win) are largely affected by the number and type of players participating in the game. *One-player games* are *non-zero-sum* and often have simpler *strategies* for deciding the next move that the player should take as they do not need to consider any consequences of other player's tactics. Although the lack of opponents does simplify the *strategy*, it is common for *one-player games* to involve an element of *chance*, for example, in the card game *'Solitaire'* the cards are shuffled and dealt randomly which complicates the process of determining the resulting outcome for a particular move. For this classification of a game, simple uninformed or informed search algorithms are often the most suitable approach for calculating the path to achieve the optimal game state [2].

Two-player games have historically been the most commonly studied in *Game Theory* with examples of artificial players successfully winning in *Chess, Checkers, Backgammon* and *Go*. These games, as with many *two-player games*, are adversarial which increases the complexity of the *strategy* that a player uses as there is another player also trying to win, whose *actions* impact the *moves* available to the first player. Although there is no strategy that is currently known to exist that can be applied universally to all *two-player games*, there are many effective strategies that do exist such as '*Minimax'* and its variants which are discussed in S*ection 2.2.3* [2]. Games that consist of three or more players are referred to as *N-player games*, they can still be *zero-sum*, however, the *payoffs* (whether negative or positive) do not have to be equal for all players involved.

A factor that affects both *two-player* and *N-player games* is whether all the players in the game are active participants or if they are non-player characters [2]. In games such as *Chess* and *Monopoly* all the players are actively trying to win the game, whereas, in role-playing games such as *Dungeons and Dragons* one player acts in a non-playing capacity as the 'Dungeon Master' so they are not working to achieve the same goal as the other players.

2.1.2.3 Cooperative vs Non-Cooperative Games

A game is considered *cooperative* if it allows *negotiations* between players to be enforced by placing them in a contract [3] such that the *payoff* from a successful coalition is shared among the members or if there is an opportunity before the game begins for the players to make a binding agreement about which strategy they will use together [8]. *Cooperative Game Theory* can be applied to games as well as other fields because it considers how much power each player holds in different coalitions which is particularly relevant to areas such as *political science* [7].

One-player and *two-player* adversarial games are generally *non-cooperative* because if there is only one player a coalition can not be formed and in the case of the *two-player* game it does not make sense to attempt to assist the opponent that you are trying to beat. In the case of an *N-player game*, however, it may be a strategic *move* to form a coalition with one player in order to prevent or delay another player winning the game.

2.1.2.4 Deterministic vs Stochastic Games

If a game involves an aspect of randomness such as a dice roll or selecting a card from a shuffled deck, it is considered *stochastic* and *non-deterministic*, for example, *Poker* and *Backgammon*. In these games, the resulting *outcome* of a *move* is not determined solely by the action that the player takes because the element of chance means that even if a player makes the same decisions in multiple plays of the game, the resulting *payoffs* will not necessarily be the same [2]. It is possible to have effective strategies that can handle *stochasticity* by calculating the probability of the random events occurring and then predicting the most likely outcomes.

In a *deterministic game*, at any point during the game a player can calculate what the *outcome* of making a particular *move* is and what actions their opponent(s) can choose to take in response. In theory, in completely *deterministic* games such as *Chess*, a player is able to predict the end *payoff* for every possible sequence of actions between themselves and their opponent, however, it is not always practical to calculate the full selection of paths if there is a time limit.

2.1.2.5 Observability in Games

Observability in a game refers to the amount of information that is available to a player throughout the game, for example, *Checkers* is a game of *perfect information* because the board state as well as previous and possible future moves are constantly available to both players [2][4]. Research into games of *imperfect information* has not been as successful, for example *Poker* and *Scrabble*, because they pose a challenge of how to calculate the best *strategy* without having all the facts. Although categorising games in *Game Theory* mainly focuses on whether a player has *perfect* or *imperfect information*, they can also have *complete information*. This means that the players share common knowledge, therefore, they are aware of the structure of the game and the *payoff* of all possible *moves* which can be taken, an example of a game with *complete* and *imperfect information* is *Poker* [3].

2.1.2.6 Sequential vs Simultaneous Games

In a *sequential* game, no two players can *move* at the same time and each player on their turn may have to *move* multiple times, whereas, in a *simultaneous* game, all players can only make one move which must be done at the same time independently [3]. Strategically what is important is not just the difference between the ordering of events (one after another or at the same time) but the amount of information available when making the decision. *Chess*, for example, is a *sequential game* because when a player makes their move they know all previous moves and have *perfect information*, whereas, in a *simultaneous game* the player's move whilst not being aware what their opponent(s) are also doing. Although this is similar to the difference between *perfect* and *imperfect information*, some games can have a mixture of *simultaneous* and *sequential moves* whereas information can be either *perfect* or *imperfect* as they are mutually exclusive [4].

2.2 Artificial Intelligence

2.2.1 Representation

One of the key challenges in all fields within Artificial Intelligence is representing and storing knowledge in such a way that it mimics a human and can be processed by a machine. There are many types of representation including *grammars*, *trees* (decision or behaviour) and *graphs* (such as finite state machines). The decision of which one to use can have a significant impact on the performance of the Artificial Intelligence algorithm that uses it [2].

2.2.2 Utility

In game theory, *utility* is a measure of rational decision making where actions that have a preferred *outcome* have a higher utility [1]. A *utility function* gives a numeric value for the *outcome* of a *move* or entire *game*, it is used in relation to deciding which path to take in a search algorithm or as a means of measuring the quality of a *representation* in a machine learning algorithm [4][2]. Similarly, *heuristics* are used in algorithms to find an approximate solution to a problem when the priority is speed as opposed to complete accuracy. One of the most common heuristic state evaluation functions is a weighted linear function of the form:

$$w_0f_0 + w_1f_1 + w_2f_2 + \ldots + w_nf_n$$

Where (f_0, f_1, f_2, … ,f_n) represent the features and components of a particular *game*, for example, the values or numbers of playing pieces on the board and (w_0, w_1, w_2, … ,w_n) represents the adjustable weighting of how important the individual features are [1].

2.2.3 Minimax

Minimax is the basic adversarial search algorithm that is used most often to find the best *move* for a player to take in games that are *deterministic*, *two-player zero-sum games* with *perfect information*. The algorithm generates a whole *game tree*, from the root to the terminal states, which displays all the possible alternating moves for two players, MAX (who moves first) and MIN (who we assume is trying to minimise the payoff for MAX with their move). A *utility function* is then applied to each of the terminal states (end leaf nodes) to calculate their values which are then recursively passed up the tree to determine the maximum or minimum utility value for MAX and MIN's turns respectively. Once the root node of the tree is reached, MAX is able to select the move with the highest *payoff* [1].

In many board games there is an aspect of limited time allowed for each move which does not allow for the whole game tree to be searched with a depth-first search, for example, in *Chess* each player has approximately 150 seconds per move. The most common method for handling this with a minimax search is to apply *alpha-beta pruning* to the tree which can reduce the time complexity from $O(b^m)$ to $O(b^{m/2})$ where b is the worst-case branch number and m is the maximum depth [1]. *Alpha-beta pruning* prevents the exploration of branches within the tree that analysis indicates that a node will never be reached because that path will never be chosen in the interest of maximising the *payoff*, then it is pruned from the tree [9].

Expectiminimax is a variation on minimax that handles *stochasticity* in games by including chance nodes in between the levels of MIN and MAX nodes. The branches leading from each chance node represent the possible results of the random action (such as the possible roles of a dice). Each chance node is labelled with the *action* and the probability it will occur and we calculate the expected value instead of the minimax value.

2.3 The Game of 'Cluedo'

'*Cluedo'* (or Clue as it is known in North America) is a strategy-based board game for 2-6 players and the variation described in *Section 2.3.1* has been chosen to be studied for this project. It is a murder mystery game that requires players to be the first to identify who murdered the victim, where the crime took place and which weapon was used by strategically moving around the game board and asking and answering questions. Each of the *suspects*, *locations* and *weapons* are represented by cards, these are either held by one of the players or are the "solution cards" which the player is trying to discover using reasoning and logical deductions based on the questions that are asked. Players move around the board according to their dice rolls and use suggestions to gather information, the details of how a suggestion is made can be found below in *Section 2.3.1*. These movements are significant because a player's piece must be in a location on the board to allow them to make a suggestion containing that location. Each player is also given a sheet from the detective notebook which they can use to record information, an example of the style of the sheet is shown in *Figure 2.1*.

Players							
WHO?							
Green							
Mustard							
Orchid							
Peacock							
Plum							
Scarlett							
WHAT?							
Candlestick							
Dagger							
Lead Pipe							
Revolver							
Rope							
Wrench							
WHERE?							
Ballroom							
Billiard Room							
Conservatory							
Dining Room							
Hall							
Kitchen							
Library							
Lounge							
Study							

Figure 2.1 Representation of a sheet that a human player would receive in a game of '*Cluedo'* or '*Speed Clue'*

2.3.1 'Speed Clue'

This is a variation of the board game '*Cluedo'* for 3-6 players which includes the 6 *suspect*, 6 *weapon* and 9 *location* cards but it does not use the dice or the board, therefore, there is no movement aspect to the game. This simplifies the strategies and the setup which involves the card types being shuffled in their piles and then one of each type of card being drawn and set aside as the solution cards (none of the players are able to see which have been selected as their goal is to discover this). The remaining 18 cards are then shuffled together and dealt to the players, for certain numbers of players such as 4, this will mean that some of the players receive more cards than their opponents, however, for the purpose of this project we will start by considering 3 players which ensures everyone receives 6 cards.

As in the original version of 'Cluedo' a player is able to make one accusation per game, which must be on their turn – this is when they believe they have deduced which are the solution cards and say, I accuse [suspect], with [weapon] in [location], inserting the cards they have inferred. The player then looks at the solution cards secretly, if they have correctly deduced the cards they have won the game, otherwise, they lose and can no longer ask but still must answer questions to allow their opponents the opportunity to make an accusation.

On a player's turn, if they choose not to make an accusation, they are able to make a suggestion instead saying, I suspect [suspect], with [weapon] in [location], as there is no board game, the rule from *'Cluedo'* which insists a player's piece must be in a location on the board to suggest that location is irrelevant. Once the player has made their suggestion, the responsive stage begins with the next player to the left, if they have at least one of the cards that you have suggested, they choose to secretly show you only one and the suggestion is disproved. However, if they do not have any of the cards the next player repeats the same process until a player has disproved the suggestion or none of the players, except perhaps the player who has asked the question, can disprove the suggestion.

If a player is shown one of the cards they suggest, then it is evident that it can not be one of the solution cards, so the player should record this on their sheet as shown in *Figure 2.1*. It is also important to record if a player does not have a card or if a player may have a card (in the situation where one opponent secretly shows another opponent a card, they must hold one of the three cards from the suggestion). A player is also allowed to make suggestions that refer to cards they hold in their hand, this can be used as part of a strategy in order to narrow down the cards that the opponents can answer about.

2.3.2 Classification

The game types discussed in *Section 2.1.2* can be used to classify the game '*Speed Clue*' as follows:

- *Zero-sum* : Since there can only be one winner in this game it is zero sum, when one player gains information they move closer to a solution which is detrimental for their opponents.
- *N-player* : This game can be played with 3-6 players, there is a 2 player variation of '*Cluedo*' which could be modified for '*Speed Clue*', however, this will not be considered in order to allow focus on strategies instead of game variants.
- *Non-cooperative* : The aim for a player is to discover the cards in the envelope first by working against your opponents, gaining the most information from them whilst minimising the amount of information you reveal.
- *Stochastic* : '*Speed Clue*' involves randomness because the suspect, weapon and location decks of cards are shuffled, and a random card is selected from each to form

the envelope. The remaining cards are also randomly dealt to all players; therefore, a player may not answer the same suggestion with the same response in all games.

- *Imperfect information* : Players begin each game with imperfect information, they are aware of cards in their hand and their aim is to establish which cards are in the envelope either through exhaustive elimination (seeing all cards in their opponents' hands) or by suggesting a card and discovering no player holds it. Furthermore, a player may reach a solution and still not have perfect knowledge as they do not need to know what every player holds if they can establish which three cards are not held by any of the players.
- *Simultaneous and sequential* : In this game, each player takes their turn to make a suggestion and then waits for their suggestion to be disproved or for all opponents to respond that they do not own the cards which is sequential game play. Since we are considering games with three players, whilst one player is responding to the suggesting player, the other player will be simultaneously recording their response on their sheet.

2.3.3 'Cluedo' and 'Speed Clue' Strategies

Although this style of game has not been as extensively researched, there are some general strategy ideas for asking and answering questions as well as movement around the board. The general basis for all strategies is that suggestions and answers should be made with the aim to maximise a player's information gain whilst reducing the potential for other players to gain information. One of the key differences in these strategies is whether players should use cards in their hand as part of their suggestions, this also relates to movement around the board because it determines how much time players should spend in locations that they own. The advantage of using cards in the hand as part of a suggestion is that it limits the cards that the other players can answer about, therefore, it narrows the potential results of the query which may be useful if the player wants information about the location of a specific card. However, if a player repeatedly uses one or more of the cards in their hand in suggestions, it is possible that their opponent's may deduce that they hold that card which is a disadvantage.

During the game of 'Cluedo', players roll dice to move their playing pieces which represent the suspects (pawns) around the board, they are only able to make suggestions about a location if their pawn is within that location. Whilst playing the game, player's also need to move strategically, for example, making use of secret passages in order to make their suggestions. Their movements also need to account for the issue that they do not have full control of their pawns movements, when a player suggests a person and a weapon in a location; the corresponding weapon and suspect pieces are moved into that location which may disrupt the planned route that the player who owns that pawn was taking. This can, however, be used as part of a strategy – if a player works out the location that is in the envelope and an opponent is in or near that location on the board, during their turn they may suggest that opponent's character piece as the suspect to move them away.

The strategic approach that players take to questioning can be guided by various levels of logical deduction and human verbal and non-verbal information, such as facial expressions or body language that may indicate they are near a solution. Research has been done into several strategies for artificial players that focus on the logical deductions, in Kingston's paper [10] he analyses and compares five strategies. The first of these, '*No intelligence*', represents an exhaustive elimination search which is the simplest questioning strategy that a player can implement, they continue suggesting until they hold or have seen the other eighteen cards. It was hypothesised that this strategy would be out-performed by any of the 'intelligent' strategies, however after investigation, despite recording knowledge of opponent's possible cards, making logical deductions and also not asking about cards that have been deduced, '*Deduction only'* did not perform substantially better [10, p.333]. '*Next-possible'* is the third strategy which focuses on confirming deductions made about possible cards in the next player's hand, whereas, '*Previous-possible'* aims to reduce the opponent's options for answering to prevent them hiding cards; in both of these strategies the same cards are excluded from questioning as in *'Deduction only*' and the priority is to find out about location cards. Both strategies showed varying performance depending on the cards dealt and the number of players involved but neither strategy was significantly better overall than '*Deduction-only'* which could be attributed to a variety of factors, notably that they do not consider all other players, only the next or previous. The final strategy, '*Next-not-held*' tries to account for this by asking about cards that are known to not be held by the next two players; the aim of this is to find cards that are not held by any player which as shown by the results is a faster approach than gathering information about cards they do possibly hold.

Conclusions drawn from the results in this paper [10] form the basis for the basic and advanced artificial players discussed in *Sections 4.2 and 4.3* which are designed to implement the least (*'Deduction-only*') and most successful (*'Next-not-held'*) intelligent strategies respectively. The implementation of the game for this paper [10] only involved playing 3 or 6 artificial players against each other, all of which implemented the same strategy which may not accurately represent which is the strongest approach. In order to compare which is the most successful, they can be tested against each other and human opponents in various games to assess if other player's strategies impact their effectiveness and to achieve objective 3 (refer to *Section 1.1*)**.** The 'intelligent' strategies proposed in this paper [10] allow for a variety of extensions that may improve them which are discussed in *Section 4.4***,** such as making use of cards in the player's hand to shrink the search space, considering all player's hands instead of focussing on one and using a mixed strategy approach. The main issue with the final three strategies proposed by Kingston [10] is that they rely on the player having knowledge of some cards in a certain player’s hand, if these strategies could be combined and prioritised this would avoid defaulting back to ‘*Deduction-only*’ which was shown to be the least effective.

When answering questions, all strategies are designed so a player should aim to minimise the amount of information that their opponents can gain. At the simplest level this means if the suggestion you are trying to disprove involves a card you have already shown the suggesting opponent, you should show them it again, therefore, they do not gain new information. This can be further extended so that if the suggestion does not contain any cards that you have shown to the suggesting opponent but it does contain a card you have shown to someone else, you should reveal this card as it minimises the amount of new information across all opponents. As most strategies aim to discover location information through questioning, it follows that during the responding stage if a player has the option between a suspect or weapon card and a location card to show, they should avoid revealing the location as this increases the challenge for their opponent [10].

Chapter 3 – Game Implementation

In order to compare the effectiveness of strategies for 'Speed Clue', a platform will need to be produced that allows 3-6 human and artificially intelligent players to play the game which will involve implementing the rules and setup described in *Section 2.3.1.* To achieve the objectives (*Section 1.1*) the program will need to allow artificial players to use either the basic or advanced strategy, which are explained in *Sections 4.2 and 4.3* respectively, and have a suitable user interface to also test against human players. The code, therefore, will have to be dynamic to accommodate the variety in the players and give the user the option to specify the deals to compare results against Kingston's paper [10, p.333] or generate them randomly for more extensive testing. In this section, the choice of software language will be justified, followed by an explanation of the design of the class structure of the code and finally the implementation and choices made for the user interface will be explained.

3.1 Implementation Language

An important decision in the implementation of this platform is which software language is the most appropriate to be used. As the code needs to be dynamic as stated above and there will be various software versions which should enhance but not break previous code, an object-oriented programming language should be used because of the modularity it allows. The platform should mimic the real game in setup, rules and the stages of play for players and the human players do not care about how this is implemented, therefore, the abstraction from object-oriented programming will be useful. Data can also be encapsulated which is vital in this project because players should only have access to their own knowledge bases; in the game '*Speed Clue*' players are not able to see their opponents playing sheets (seen in *Figure 2.*1) which is why they have *imperfect information,* otherwise there is no challenge. Although there is an extensive list of object-oriented programming languages including C++, C#, Ruby and many more, I chose to compare and decide between the languages that I have the greatest experience with, Python and Java, so there was not a steep learning curve.

Assuming a player does not make any deductions and uses an exhaustive elimination strategy, they will select cards in order from each of the categories to form a suggestion, although they will not ask about cards in their hand or ones which they have seen. The maximum number of rounds it would take for them to reach the solution would be if the last suspect, weapon and location cards on the list were those in the envelope because they would have to ask about all the remaining cards that they do not hold. In this scenario assuming the player starts first, for 3 players it would take this player 12 rounds of guessing to come to a solution, whereas for 6 players it would take the player 14 rounds of guessing. Therefore, the maximum total number of possible guesses for this strategy would be 13 rounds of full guessing (assuming no other player worked out the solution) and one extra guess at which point the player would make the accusation, so 79 guesses. As discussed in *Section 2.3.3* there is evidence that strategies that implement logical deduction to guide question asking allow the game to be solved in a lower average number of rounds, therefore, for performance considerations we can consider 14 to be a maximum. Although Java is considered a more powerful language than Python and the strategies may be more complex, the reduction in the number of rounds suggests that Python will be capable of running many instances of the game in a reasonable amount of time.

One of the main aims of this project is to provide a design plan of potential strategies that could be researched further, and the code should be designed so that it can form a basis for future projects. An advantage of using Python is that as well as being easy to write, it is dynamic and human-readable, for example, unlike Java, variable types do not need to be declared, therefore the code is more similar to the English language. This means that future students should be able to understand how the code operates without having extensive knowledge originally.

Both Python and Java are object-oriented languages that have automatic memory management and have command line as well as Graphical User Interface options (see *Section 3.3* for the decision on which is chosen for this project). They also have extensive standard libraries which improves the reliability of the code as well as the speed of development because methods do not have to be written from scratch. For this project, Python is the better option because it is naturally easy to read and write and is powerful enough to handle the processes involved.

3.2 Class Structure and Details

Before beginning to implement the game, the class structure needed to be designed to ensure an object-oriented approach was being used that would allow for extensions to the game when different strategies were added. To separate the code into classes, we can establish the objects in the game and their relationships to each other (shown in *Figure 3.1*). One instance of a *Game* contains several *Players* (which can be human or artificial with various strategies) and a *GameState* which represents how the game changes over time, this makes use of a *Deck* which is made from a set of *Cards*. Whilst considering the *Card* and *Player* classes which are discussed in more detail below, it became apparent that enumerations could be useful for representing a card's type and a player's knowledge type about a card.

When the program is run, the initial setup involves the user choosing how many human players and how many artificial players will be involved in the game as well as selecting the strategy types for the artificial players and if the deals should be randomised or selected from a list. So, the first objects that are created are the artificial players which are instances of the *Player* class, their strategy, name and player type all need to be specified. One of the key attributes of the *Player* class is the knowledge base which represents the sheet that human players store their information on, for further information refer to *Section 4.1.1*. The manipulation of this knowledge base depends on their strategy; however, it is always initialised as a dictionary with keys that represent each player and card combination (corresponding to a cell in the table displayed in *Figure 2.1*) and values which are from the *KnowledgeType* enumeration (presented in *Figure 4.1*).

These *Player* instances are then used to generate the *Game* instance which initialises a *GameState* object. The *GameState* class is responsible for handling the changing state of the game as play continues until a player finds a solution, firstly, it generates a deck using the class *Deck* which contains all of the *Card* objects that are necessary for the game, each of which has a name and a card type which is an instance of the *CardType* enumeration class. The deck that is generated by the *GameState* object is then used to deal the cards, unless a deal is specified by the user during the initial setup, a singular suspect, weapon and location card are selected from the *Deck* object and the remaining cards are dealt into the hands of the *Player* objects. Finally, to complete the setup of the game the knowledge base mentioned above is updated for each player according to the *Card* objects in their hand.

The *Game* object monitors if the game has been completed (if a player has found the correct solution), otherwise, it allows players to get their turns in order and moves between phases accordingly. Whereas, the *GameState* class is responsible for ensuring the rules are adhered to and game play is as expected, it handles each of the states that a player can be in on their turn – starting, accusing, suggesting and ending. Within this class are the methods to handle generating suggestions, disproving suggestions, updating the player's knowledge base and checking if the player is ready to make their accusation.

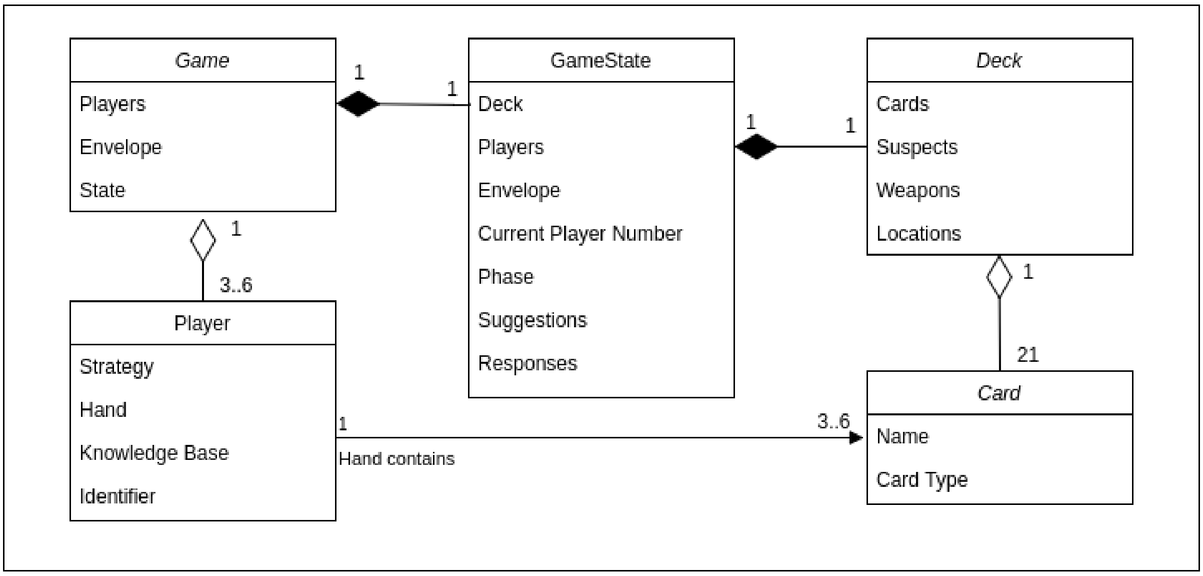

Figure 3.1 A UML diagram demonstrating the relationships between each of the classes (enumerations have been omitted as they are used for simplicity, they are not necessary for the program to run as intended).

3.3 User Interface

Testing the effectiveness of strategies between artificial players provides an insight into which factors are significant in improving their chances of winning, however, it is widely acknowledged that an important test for Artificial Intelligence is whether it can beat a human opponent [2, p.8]. Therefore, one of the key stages of implementing this program was to develop a user interface that mimics the game of '*Speed Clue'* effectively whilst not impacting performance. As this project does not focus on the board game aspect of '*Cluedo'* and the attention is on developing strategies; the interactions that the human player needs to have can all be performed via a *text-based interface*. Although the initial setup of the game is also console-based, the aim is to ensure that the human playing against the artificial players does not see this, therefore, they will not be aware which strategies the players are using. Although there was initial consideration to allow multiple human players to play simultaneously, this would involve potentially hosting the game on a server to allow the players to interact on separate machines (and not see each their opponent's hands). This option was not deemed as important as focussing time on researching and implementing strategies, so there will only be one human player interacting with the game at a time, however, multiple humans will be tested.

For the initial setup of the game, the user who is running the tests will have a variety of configuration options and these will be displayed on the console as shown in *Figure 3.2*. The programme needs to allow for randomised deals, to test the performance of the strategies against each other and the human players. It is also important to allow the user to manually configure the card deals in order to compare the results against those found in Kingston's paper [10, p.333] to check that the strategies have been implemented correctly and produce similar results to the predictions.

```
File  Edit  View  Search  Terminal  Help

(base) kmitchell@kmitchell-HP-Pavilion-Notebook:~/comp3931/comp3931-individual-project$ python cluedo.py
Please setup your game of SPEED CLUE
Would you like to include a human player? - please enter Y or N
Y
Would you like randomised deals? - please enter Y or N
Y
How many artificial players would you like to include - please enter a number between 2-5
2
For player 0 would you like a basic or advanced artificial player - please enter Basic or Advanced
Basic
For player 1 would you like a basic or advanced artificial player - please enter Basic or Advanced
Advanced
Please enter human player's name
Katie
```

Figure 3.2 The setup of a game of 'Speed Clue' with a human player and random deals

The human player will be given a '*Cluedo*' sheet (see *Figure 2.1*) to record the information that they gather and shown their hand when the game begins. As they are able to use their information sheet without guidance, the programme must display the cards they hold in their hand as well as their opponent's names, but not strategies, at the start of each game. On their turn they will be firstly asked if they are ready to make an accusation, then they will be prompted to enter a suspect, weapon and location card which, according to their response, will be checked against the envelope for an accusation or suggested to the next player. The interface will show their opponent's responses which will be of the format 'Player X does not hold any of the cards' or 'Player X is showing you the card Y' where X and Y are variables. Responding to suggestions is also necessary for the human player, so they will be able to select which of the cards to show if they hold them in their hand. *Figure 3.3* demonstrates the view of a human player

Validation and verification is a key component to this interface, the artificial players cannot cheat, and the human player needs to be prevented from attempting to cheat. During their suggestion their inputs to the system will need to be checked that they represent cards and that the programme has interpreted it correctly, a full instruction guide, and card list will need to be available at any time. Also, when the human player is responding to an opponent the system needs to check if they hold any of the cards and verify that their response is correct.

```
File  Edit  View  Search  Terminal  Help
WELCOME TO YOUR GAME OF SPEED CLUE
The players are:
Alice : AI
Bob : AI
Katie : Human
The cards in your hand are:
Conservatory
Lead Pipe
Ballroom
Colonel Mustard
Billiard Room
Rope
Game begins
PRESS ENTER TO MOVE TO NEXT PERSONS TURN
Bob is suggesting  Professor Plum  with the  Lead Pipe  in the  Library
YOUR TURN
Can you disprove suggestion - please enter Y or N
Y
Which card would you like to show - please enter card name e.g. Miss Scarlett
Lead Pipe
Bob has finished their turn
PRESS ENTER TO MOVE TO NEXT PERSONS TURN
Do you want to suggest or accuse - enter Suggest or Accuse
Suggest
Which suspect do you want to suggestion? - please enter full name e.g. Miss Scarlett
Miss Scarlett
Which weapon do you want to suggestion? - please enter full name e.g. Rope
Wrench
Which location do you want to suggestion? - please enter full name e.g. Library
Library
Katie is suggesting  Miss Scarlett  with the  Wrench  in the  Library
Alice responds with card Library
Are you ready to make an accusation - please enter Y or N
```

Figure 3.3 A screenshot of the game as viewed by a human, their option to respond to an opponent and make a suggestion.

3.4 Workflow

Regardless of whether human players are involved, or which strategies artificial players use, the stages of play in the game remain the same. The flowchart in *Figure 3.4* represents the actions and decisions that the game needs to handle, and they were used as a basis for designing the code, particularly the *GameState* class.

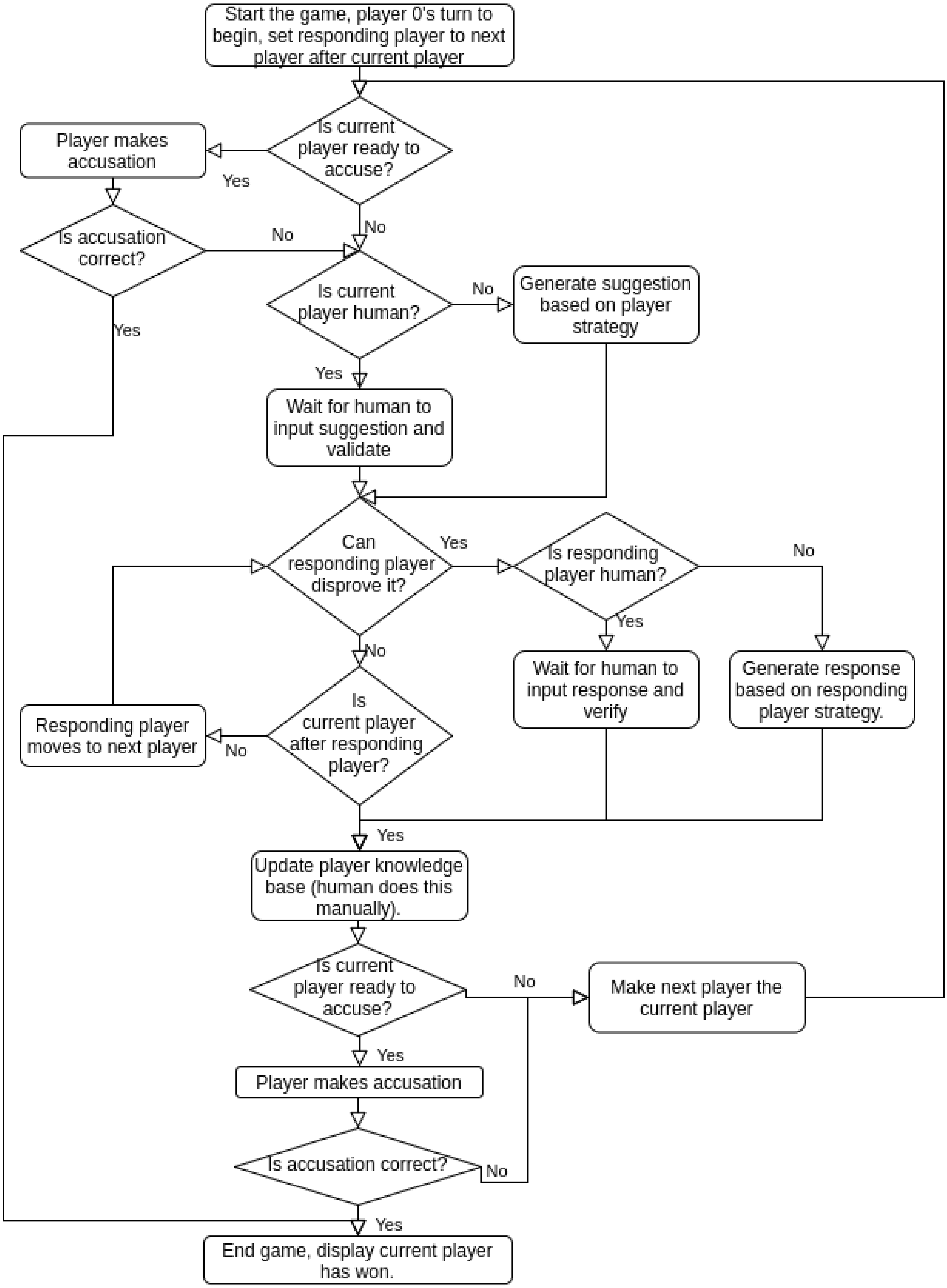

Figure 3.4 A flowchart representing the game play aspect of the implementation of '*Speed Clue'*.

Chapter 4 – Artificial Player Implementation

This chapter will provide an overview of the two types of artificially intelligent players that will be tested and evaluated in *Chapter 5*, as well as an analysis of how the implemented strategies could be enhanced and developed further. One of the key tasks in developing the artificial player was maintaining their knowledge base and updating it using logical deduction, therefore, the decisions regarding this are explained in this chapter.

4.1 General Artificial Player

During an artificial player's turn, since all players in this game are using deductions as opposed to relying on cards seen, they firstly update their knowledge base with information gathered from suggestions and responses of opponents since their last turn. In a human game of '*Speed Clue*' players usually continually update their information sheet, noting responses and suggestions made by opponents when it is not their turn. However, implementing this in the code is not as efficient as storing all responses and suggestions made during the game and allowing players to access this. After this update, the player makes an accusation if they are ready, otherwise, they generate a suggestion based on their strategy and record the information in their knowledge base. According to the rules of '*Speed Clue*', any player is also allowed to accuse after they make a suggestion as it is still their turn, therefore, the programme checks if the player is ready to accuse and then handles the accusation or moves play on to the next player. As well as having different strategies for questioning, the *basic* and *advanced* players also differ in how they respond to an opponent's suggestion.

4.1.1 Knowledge Base

In '*Speed Clue*' and '*Cluedo'*, a player's success fundamentally relies on how they make use of the information sheet. Since a human is limited in how much information they can recall, after several rounds it is unlikely that a player would be able to remember all the suggestions and responses and then formulate a suggestion without noting down the information that they gather. An artificial player can store vastly more information; therefore, they could generate suggestions by going through the stored suggestion and responses list, however, this would be inefficient as the same deductions would be repeated each turn. Each artificial player needs a knowledge base that can be updated at the start of their turn and when opponents respond to their suggestion so it can be used to evaluate if they are ready to accuse.

Python offers a variety of data structures that could have been used to represent the knowledge base, including nested lists or arrays of lists, however, the simplest approach after consideration was to create a dictionary in which each key-value pair represented a cell on the information table see *Figure 2.1*. A dictionary was selected because it is mutable, therefore, when the artificial player updates their information, they can overwrite the value for an existing key. Although the value element can be changed, the key type in a dictionary has to be immutable, therefore, a tuple can be formed of a *Card* and *Player* object which corresponds to the cell references in a table which generally use the card and player as the row and column identifiers respectively.

During the setup of the game a dictionary is generated for each player which they have exclusive access to – other artificial players cannot see or manipulate their knowledge base. The keys in the dictionary are tuples of the form (card, player) for each card in the deck and player in the game. The value for each of these keys is a member of the *KnowledgeType* enumeration, shown in *Figure 4.1*; if the artificial player owns the card then the value can be A, E or F, otherwise, it will be A-D. Within the dictionary keys, there is also a (card, 'overall') tuple for each card in the deck which can take the *KnowledgeType* value A, G or H.

The dictionary items that have the (card, 'overall') key for each card in the deck are included to reduce the searching required when checking or updating the knowledge base. It stores the overall knowledge of if the card is in the envelope, not in the envelope or unknown. When the artificial player checks if they are ready to make an accusation, for each category they can check if there is a (card, 'overall') key pair which has the value *KnowledgeType.H* which is a maximum of 21 checks. Without this inclusion, the artificial player would have to check each card, player pair to see if there is a card in each category with the value *KnowledgeType.C* for each player, in a six player game this could involve 126 checks.

An enumeration is used for the values in this dictionary to improve the conciseness of the code and to make it easier to read. The types A-D are considered standard notation, in the rules there is guidance that advises players to record this knowledge for the cards on the information sheet. As the artificial player is aiming to reduce the amount of information that opponents gain from their responses, it is useful to record if a player has revealed a card in their hand to an opponent, hence knowledge types E and F are also required. Finally, the overall knowledge for any card can be none or known to be in or not in the envelope or known to be not in the envelope because it is in a player's hand.

```
class KnowledgeType(Enum):
        ##The different deductions that a player can make about each card
for the other player, A-D are relevant for cards you do not own, E-F are
relevant for the cards you own (to note what you have revealed), G-H refer
to if cards are definitely known to either be in or not in the envelope

        A = "No knowledge"
        B = "Owned by"
        C = "Not owned by"
        D = "May own it"
        E = "Not revealed"
        F = "Revealed"
        G = "Not in envelope"
        H = "In envelope"
```

Figure 4.1 A screenshot of the code base which shows the members of the *KnowledgeType* enumeration class.

4.1.2 Logical Deductions

The main aspect that makes the artificial players in this programme intelligent is the logical deductions that they make at different stages and how they interpret the information they gain to guide their suggestions. The deductions that are used by the players are described below; if a deduction holds then it is possible that other logical conclusions may also be true, and these are referenced at the end of each deduction statement. Pseudocode has also been included to demonstrate the effects of the deductions on the knowledge base, for further information on the knowledge base and the KnowledgeType values refer to *Section 4.1.1*. The following list is the basis of logical deductions for both the *basic* and *advanced* players:

1. If a player holds a card in their hand then it can not be in the envelope or any other player's hand. [Check if deduction 6 holds]

   Function player_holds_card(card, player)

   > KnowledgeBase[card, player] ← KnowledgeType.B
   >
   > KnowledgeBase[card, 'overall'] ← KnowledgeType.G
   >
   > **For each** remaining player **in** game **do:**
   >
   > > KnowledgeBase[card, player] ← KnowledgeType.C

2. If a player X can not respond to a suggestion of cards A, B or C then that player does not have A, B or C in their hand. [Check for A, B and C if deduction 3 holds]

   Function player_does_not_hold_cards(cardList, player)

   > **For each** card **in** cardList **do:**
   >
   > > KnowledgeBase[card, player] ← KnowledgeType.C

3. For any card C, if all of the players are known to not have C in their hand (and the player making deductions does not hold it) then it must be in the envelope. [Check if deduction 7 holds]

   Function no_player_holds_card_check(card)

   > Card_not_held ← **True**
   >
   > **For each** player **in** game **do:**
   >
   > > **If** KnowledgeBase[card, player] **does not =** KnowledgeType.C **do:**
   > >
   > > > Card_not_held ← False
   >
   > **If** Card_not_held **equals True do:**
   >
   > > KnowledgeBase[card, 'overall'] ← KnowledgeType.H

4. If a player is known to not hold two cards A and B in their hand and they show a card with value A, B or C, it follows that they must have shown C. [Deduction 1 follows for the player and card C]

Function check_if_player_does_not_hold_two_cards(cardList, player)

Cards_not_held_list ← empty list

For each card **in** cardList **do:**

If KnowledgeBase[card, player] **equals** KnowledgeType.C **do:**

Add card **to** Cards_not_held_list

Else do:

Card_held ← card

If number of elements in Cards_not_held_list **equals** 2 **do:**

KnowledgeBase[Card_held, player] ← KnowledgeType.B

5. If a player shows an opponent a card A, B or C then they might hold any of these in their hand unless for a card it is already known that they do or do not hold it.

Function player_may_hold_cards(cardList, player)

For each card **in** cardList **do:**

If KnowledgeBase[card, player] **does not equal ...**

… (KnowledgeType.C or KnowledgeType.B) **do:**

KnowledgeBase[card, player] ← KnowledgeType.D

6. If all cards in a category except one are not in the envelope, then the remaining card must be.

Function check_card_category(category)

Cards_possibly_in_envelope ← 1

For each card **in** category **do:**

If KnowledgeBase[card, 'overall'] **does not =** KnowledgeType.G **do:**

Increment Cards_possibly_in_envelope **by 1**

Envelope_card ← card

If Cards_possibly_in_envelope **equals** 1 **do:**

KnowledgeBase[card, 'overall'] ← KnowledgeType.H

7. In any card category (suspects, weapons or locations), if there is a card that is known to be in the envelope then none of the other cards in that category can be in the envelope.

   Function category_card_in_envelope_found(cardInEnvelope, category)

   For each card **in** category **except** cardInEnvelope **do:**

   KnowledgeBase[card, ‘overall’] ← KnowledgeType.G

During the initial creation of a player’s knowledge base, they have *KnowledgeType.A,* ‘No Knowledge’, about any of the player, card or overall, card combinations. The first time they use reasoning is when they are dealt their hand, for each of the cards in their hand deduction 1 is used. At the beginning of each turn as mentioned before, the first phase for the player is to update their knowledge base from the list of suggestions and responses since their last turn which can be accessed by all players. For each suggestion of cards, A, B and C, if a player responds that they cannot show a card then deduction 2 can be used to update the knowledge base. Otherwise, if they have shown one of the cards, the artificial player must check if deduction 4 holds, if it does not then deduction 5 follows. After this they check if they are ready to accuse, the artificial player will make an accusation when they know which suspect and weapon and location cards are in the envelope. Although the *basic* and *advanced* players approach suggesting differently depending on their strategy, they draw the same logical conclusions from the responses to their suggestion. When they make a suggestion, if an opponent responds they do not have any of the cards then we use deduction 2 to update their knowledge, however, if a player shows them a card, we update their knowledge base according to deduction 1.

4.2 Basic Player

The design of the *basic* player is based on the *‘Deduction only*’ strategy from the paper [10] analysed in *Section 2.3.3*, therefore, it records knowledge of cards that opponent’s possibly hold which are determined using the logical deductions in *Section 4.1.2*. The strategy used to guide question asking is simple to implement, for each of the categories – suspects, weapons and location, we firstly consider if the artificial player has worked out which card is in the envelope. If the player does not know the envelope card; we only consider the cards that have not already been deduced as not being in the envelope and from this set a random card is selected to form the suggestion. However, for a category if the card in the envelope has already been deduced, they select one randomly from all the cards in that category – this is to prevent the player repeatedly asking about the envelope card as other player’s strategies may consider this significant.

As the aim for the artificial player is to minimise the amount of information that their opponents can gain, the *basic* player when responding to a suggestion will firstly check if they have already revealed any of the suggested cards to that player. If they have shown a card to the suggesting player in a previous turn, this will be recorded in the knowledge base and they will reveal it again so the player can not gain new information. Otherwise, if they have multiple cards from the suggestion in their hand, they will show the first one in the list. Since this list is generated from the suggestion which is comprised of suspect then weapon then location, if they have more than one card and one of their options is the location it will not be shown because it will be last in the list. This is important because as explained in *Section 2.3.3*, the location card in the envelope is the most challenging to deduce as there are 9 options whereas there are only 6 for suspects and weapons, therefore, avoiding revealing information about location cards makes the opponent's task more challenging.

4.3 Advanced Player

To achieve objective 3 of this project (found in *Section 1.1*), an *advanced* player had to be implemented which in theory should improve on the *basic* player and this can then be evaluated during tests. The conclusions that were drawn from the research paper [10] discussed in *Section 2.3.3*, suggested that the 'Next-not-held' is the most effective strategy for an artificial player, although this is only in the context of all players using the same strategy. Therefore, as well as testing that the *basic* and *advanced* strategies perform as expected when tested against themselves, this project also investigates their success against each other and human players. The results of the tests should show if using the logical deductions from *Section 4.1.3* to guide question asking, which is the basis of the *advanced* strategy does improve performance in '*Speed Clue*' as expected.

The 'Next-not-held' strategy focuses on cards that are known to be absent from player's hands and if this is not relevant due to current player knowledge; it defaults to the same questioning strategy as the basic player. The ideal situation when selecting cards to use for the suggestion is to ask about two cards that are not in the next player's hand as well as one card that is not held by the player after next. If this can not be achieved, the artificial player should suggest one card that the next player does not own along with two random choices. During the selection process the same rules apply for cards as with the *basic* strategy, cards that have been seen, deduced or held by the player should not be included in the suggestion unless the card in the envelope has been worked out for that category. This strategy is designed to try to search for the solution using deduction 3 from *Section 4.1.3* by identifying cards that are not held by any players which may be more consistent than the random approach taken by the *basic* player. In order to compare the effectiveness of the *basic* and *advanced* players, only the question asking strategies were changed so the *advanced* player relies on the same approach for responding to suggestions as the *basic* player. If both question and responding strategies were altered it would be difficult to conclude which had the greater impact, however, alternative answering strategies are discussed in *Section 4.4.4*.

4.4 Alternative Player Strategies

There are a variety of strategies for question asking, answering and recording information that can be used in 'Speed Clue' and a limited number have been implemented for testing in this project. Results in *Chapter 5* will allow conclusions to be drawn about which is more successful individually, however, it is more likely that a mixed strategy approach will be more useful because human players often change strategies dependent on current information. For example, a human may begin with a random approach to question asking and later in the game start using cards in their hand along with knowledge of cards that players do not hold in order to shrink the search space and extract specific information. Furthermore, in future development as discussed in *Section 6.3* trialling a combination of the implemented strategies detailed in *Sections 4.2* and *4.3* along with those discussed in this section may be the most effective for an artificial player.

4.4.1 Using cards in hand

It is permitted in the rules of '*Speed Clue*' to use cards in your hand to form suggestions and this can be useful in various cases, however, the implemented strategies avoid it. In the situation where a player has deduced the suspect and location cards in the envelope and need to gather information only about whether opponents have weapon cards, they can form suggestions with suspect and location cards if they hold them in their hand. This would mean that their opponents would only have the option to reveal the weapon card in the suggestion if they have it, so they cannot withhold the information. Also, some strategies make deductions based on cards used in suggestions, if the artificial player realises that an opponent is doing this, they could misdirect using cards from their hand. An issue with this approach is that human players may be able to deduce that you hold a card if you repeatedly use it to form suggestions, therefore, a key element of this strategy is finding a balance between gaining specific information whilst not revealing cards in their hand by accident.

4.4.2 Extending questioning strategies to consider all players

Of the four intelligent strategies analysed in *Section 2.3.3* from Kingston's paper [10], none focussed on using information gathered about all opponents absent or possible cards to guide question asking. The 'Next-possible' and 'Previous-possible' focussed on the next and previous players respectively, however, they could be extended further to consider all players in the game. For the 'Next-possible' strategy, the suggestion could be composed of one card from the next player's possible cards and another from the list for the player after. This focusses on maximising the possible confirmations of cards during one turn which can then be used with deduction 1 from *Section 4.1.2* because if the first player responds no, then as well as gaining information that they do not hold cards, there is a higher chance of gaining a confirmation from the next player on a relevant card. Considering all players also reduces the need to default to the '*Deduction only*' strategy which has been demonstrated as being less successful. For the 'Previous-possible' strategy, if an artificial player does not have a list of possible cards for the previous player, they could consider the player before them instead and the same goal applies. If the suggesting player has already deduced which suspect card is in the envelope and the next player's list of possible cards only consists of suspects then applying the 'Next-possible' strategy would not gain useful information, however, considering the player after next may be more beneficial.

4.4.3 Recording cards used in suggestions

As mentioned above, some human players make deductions based on what cards a player uses in their suggestions as well as how they respond during the game. If a player repeatedly uses one card during their suggestions this may be because their strategy may include using cards in their hand to form suggestions (see *Section 4.4.1*). Extensive research would need to be done to evaluate how many times a player needs to include a card in their suggestions to reasonably assume that they hold it in their hand. Once the artificial player reaches the stage where they suspect that an opponent holds the card because of this technique, they may form a specific suggestion in order to confirm the information. A high level of intuition is required to use this strategy successfully and deducing the probability that an opponent holds a card based on its uses in suggestions is complex, however, it could increase the possible cards list allowing the strategies discussed in *Section 2.3.3* to be used with more success.

4.4.4 Answering strategy to further minimise information loss

Although the main variant in players' strategies is how they form suggestions and deduce information from them, it is also important to consider what information they reveal when responding to opponents. Since 'Speed Clue' is a game that relies on knowledge, a player can make their opponents likelihood of reaching a solution before them smaller if they can minimise the information they reveal. In the *basic* and *advanced* strategies implemented in this project, the player monitors which players they have revealed cards in their hand to and avoids giving new information by repeatedly showing the same cards to players if that is an option. This strategy can be extended further, for each of the cards in a suggestion, if they have not revealed any to the player asking the question then they should check if they have revealed any to the other players in the game. If so, they should prioritise revealing the card that they have shown to the most players because it is likely that the opponent making the suggestion will have already deduced that they possibly hold that card whereas they may have no information about cards which have not been shown to any players. This strategy is based on the premise that confirming a possible card is less detrimental than revealing a previously hidden card and this would need to be confirmed in tests.

Chapter 5 – Testing the Strategies

To achieve the aims of this project, specifically, the third objective (refer to *Section 1.1*), the strategies that have been implemented need to be tested to compare their effectiveness. As the *basic* and *advanced* players (outlined in *Chapter 4*) are based on Kingston's [10] *'Deduction-only'* and *'Next-Not-Held'* strategies, the first set of tests are designed to evaluate the artificial player's success when competing against opponents using the same strategy. The aim of these tests is to replicate the results produced in Kingston's paper [10], therefore, they involve programmed deals. In human games of '*Speed Clue'* it is unlikely that all opponents use the same strategies, therefore, the second stage of testing involves comparing how successful the different artificial players are when they play against each other in the game environment. Finally, in order to further review their effectiveness, both players can be tested to see if they are capable of defeating human players in various game states that differ in the number of each type of artificial player.

5.1 Testing Artificial Players Using the Same Strategies

The first set of tests are intended to ensure that the *advanced* and *basic* players are implemented correctly according to the strategies from Kingston's paper [10] so their aim is to replicate similar results to those detailed in this paper. Therefore, these tests will record how many rounds it takes any player to reach a solution when their opponents are also using the same strategy. For each of the combinations of 3 and 6 *basic* and *advanced* players and the deals 1-6 explained in *Appendix B*, an average number of rounds to reach a solution is calculated over 500 games and the results are presented in *Table 5.1.*

	3 players		**6 players**	
	Basic	Advanced	Basic	Advanced
Deal 1	6.17	5.268	4.744	3.982
Deal 2	6.104	5.002	4.568	4.302
Deal 3	6.272	5.084	4.772	4.07
Deal 4	6.13	5.128	4.654	4.5
Deal 6	6.442	5.238	4.204	4.158
Deal 7	6.408	5.272	4.188	4.374

Table 5.1 The average number of rounds for a solution to be found when all opponents are the same type of artificial player for deals 1-6 in *Appendix B*.

The results from this set of tests demonstrate that overall, the *advanced* player solves a game of ‘Speed Clue’ in a shorter number of rounds than the *basic* player, suggesting that the strategy involving using logical deductions to guide questioning is more successful. This general conclusion corresponds to the evaluation of the results in Kingston’s paper [10, p.334], however, the tables are not equivalent in the variation between the deals for the players. After further comparisons of the designs for the strategies, the software tool used by Kingston [10] selects cards sequentially as opposed to randomly, therefore, deals 1-6 in *Appendix B* are less relevant for the application in this project. Although the *advanced* and *basic* artificial players were based on the general premise of the ‘*Deduction only*’ and ‘*Next-Not-Held*' strategies outlined in *Section 2.3.3*, the decision was made to choose cards randomly not systematically. The aim of this was to counteract opponents who make deductions based on what cards are included in suggestions as the random aspect should minimise repeated use of cards.

Furthermore, whilst it is important that the results from *Table 5.1* indicate that the advanced player should outperform the basic player to concur with Kingston’s paper [10], the differences in those deals are less significant. Instead, a more accurate evaluation of the strategies can be reached by testing on the deals 7 and 8 in *Appendix B* which focus on the type of cards that player’s hold in their hand instead of their placement in the categories, see *Table 5.2*. As both the *basic* and *advanced* players default to random selection of cards, on their first turn if a player has no information on suspects then despite ‘Miss Scarlett’ being before ‘Professor Plum’ in the knowledge base they are equally likely to be suggested. Therefore, the cards ‘Miss Scarlett’, ‘Rope’ and ‘Kitchen’ are the envelope cards in both deals because their position in the category is irrelevant. The deals differ in the cards that are held by the players, deal 7 aims to populate all player’s hands with as many cards of the same type, whereas, deal 8 distributes the card types as evenly as possible in order. As this project focuses on all game possibilities, the tests on these deals are conducted on *basic* and *advanced* players for 3,4,5 and 6 players. The rules of ‘*Speed Clue*’ indicate that in the situation where the non-envelope cards cannot be shared evenly amongst all players (18 cards cannot be distributed equally among 4 or 5 players), the extra cards should still be dealt leaving the latter players with fewer cards. Since these tests are not concerned with which player wins and focus instead on the number of rounds it takes to reach a solution, this should not be a concern as deals 7 and 8 are adapted for the various number of players. However, the 4-person variant is the ideal test case for deal 7 because players 1 and 2 can receive the 5 remaining suspect and 5 remaining weapon cards respectively and the location cards can be split between player 3 and 4 (for further detail on how the cards are distributed for the other games refer to *Appendix B*). Whereas, the test involving 5 players is well suited to deal 8 because each of the players can receive one suspect, one weapon and one location card with the additional 3 location cards being allocated to random players.

	3 players		4 players		5 players		6 players	
	Basic	Advanced	Basic	Advanced	Basic	Advanced	Basic	Advanced
Deal 7	8.762	6.036	6.996	4.456	5.33	4.428	4.6	4.31
Deal 8	6.332	5.098	5.63	5.036	5.136	4.83	4.564	4.754

Table 5.2 The average number of rounds for a solution to be found when all opponents are the same type of artificial player for deals 7 and 8.

From *Table 5.2* it is evident that in general, the *advanced* player should be able to reach a solution for both deals in a shorter time compared to the *basic* player, when playing against opponents of the same type, which suggests it is a more successful strategy. The 6-player test for deal 8 opposes this conclusion and further analysis of the data shows that the improvement of the *advanced* player compared to the *basic* player reduces as the number of players increases. An explanation for this is that the *advanced* strategy only focuses on finding cards that are not held by any players, this is significantly more challenging with a larger number of opponents. The trend in the data also suggests that if a larger number of artificial players are involved, the game will be solved in a shorter number of rounds. Since this applies for both strategies, the logical deductions, particularly those made at the start of a turn when updating their knowledge base from suggestions since their last turn, are important which should be considered for any future strategy approaches. Interestingly, whilst the *basic* strategy was more successful in every test for deal 8 compared to deal 7; for all tests except the 3-person case, the *advanced* strategy reached a solution in fewer rounds for deal 7 instead of deal 8. This indicates that there may not be one strategy that is categorically the most effective for an artificial player; instead a strategy should be selected based on the hand that they are dealt. An artificial player could adapt their strategy based on their current knowledge after the hand is dealt and potentially throughout the game, which may reflect a human's approach more accurately, therefore, improving the efficiency of the player at reaching a solution.

5.2 Testing Basic Player Against Advanced Player

Evidence from *Section 5.1* suggests that the advanced player can solve a game faster than the basic player regardless of the card deals, however, unless they are tested against each other they can not be accurately compared. Although the *advanced* player's question asking strategy is compatible with artificial players of the same type, the random element of the *basic* player may be more effective against the more logical approach. In order to test this, we need to evaluate the player's success in all possible combinations, however, as the order of the players is shuffled randomly at the start of each new game, the scenario of *advanced* players 1 and 2 and *basic* player 3 is identical to *advanced* players 1 and 3 and *basic* player 2. We only consider the number of each type of player, not the order changing, therefore, there are a total of 14 possible combinations of *basic* and *advanced* players for '*Speed Clue*' which involves 3-6 players. For each of these combinations, we are concerned with which strategy is most successful on average, therefore, each test involves conducting 500 random deals. The results of these tests are summarised in *Table 5.3,* each row represents a different combination and the letters 'A' and 'B' correspond to the '*basic*' and '*advanced*' players respectively. Whereas, an 'X' demonstrates that the player is not relevant for the combination, for example, players 4,5 and 6 are not considered in the three person test case. The total number of times that each player wins over the 500 random deals are recorded in brackets next to the letter representing the type of artificial player.

Test Number	Player 1	Player 2	Player 3	Player 4	Player 5	Player 6
1	A (210)	B (133)	B (157)	X	X	X
2	A (195)	A (182)	B (123)	X	X	X
3	A (161)	B (117)	B (114)	B (108)	X	X
4	A (150)	A (140)	B (112)	B (98)	X	X
5	A (125)	A (138)	A (138)	B (99)	X	X
6	A (124)	B (83)	B (103)	B (104)	B (86)	X
7	A (119)	A (128)	B (75)	B (86)	B (92)	X
8	A (98)	A (124)	A (111)	B (87)	B (80)	X
9	A (101)	A (114)	A (106)	A (111)	B (68)	X
10	A (96)	B (84)	B (70)	B (72)	B (87)	B (91)
11	A (105)	A (98)	B (76)	B (77)	B (80)	B (64)
12	A (96)	A (109)	A (84)	B (54)	B (76)	B (81)
13	A (102)	A (99)	A (83)	A (83)	B (78)	B (55)
14	A (94)	A (85)	A (81)	A (87)	A (88)	B (65)

Table 5.3 The number of victories in all possible combinations of *advanced* 'A' and *basic* 'B' artificial players in the 3-6 player game. An X represents when the player is not involved in the game, the numbers in brackets are the amount of games the player won.

As the results in *Table 5.3* demonstrate, on average the *advanced* player outperforms the *basic* player in all scenarios which further confirms the idea that using logic to guide question asking is a more successful strategy for 'Speed Clue' than using random guesses. The *advanced* player defaults to the *basic* strategy if it can not form a suggestion from the next player's hand and this random aspect accounts for the variation in results between players of the same type. Although the players using the same strategy differ slightly in their number of victories, with the highest difference for *advanced* players of 26 in test 8 between players 1 and 2; there are no results in the table that demonstrate the *basic* player is more successful on average. Whilst the *advanced* player exceeds the *basic* player on average regardless of the number of each type of player, it is important to note that the *basic* player still wins games in all the tests. If we consider test 10 in *Table 5.3* player 6 who is using the *basic* strategy wins only 5 fewer games than the *advanced* player 1, this suggests that the *advanced* player can still be improved, potentially with some of the strategies discussed in *Section 4.4.*

5.3 Testing Artificial Players against Humans

Artificial players for well-researched games such as Chess and Checkers have historically only been considered successful when they were able to beat human champions, therefore, this is another way to compare the *basic* and *advanced* strategies. Originally the plan to run these tests involved multiple human agents playing various game states which I would set up, therefore, they would not be aware which types of artificial players were involved. However, this plan had to be adapted due to unforeseen circumstances discussed in *Appendix C*, which meant that other humans could not be involved in the testing process.

Instead, the setup process is modified so the human who is now setting the game up and participating in the tests only enters how many artificial players should be involved (between 2-5) and does not specify the strategy. Then for each number of players, the program initialises a sequence of games which involve a different combination of the types of players. As the aim is to replicate a standard game of '*Speed Clue*' the deals are randomised and the human player is able to see the minimum information required, that is they view their hand at the start of the game but cannot access their opponent's hand. During the game, the human player enters suggestions into the command line (as detailed in *Section 3.3*) and can view other player's suggestions when they are made. When an opponent is making a suggestion, if another player responds with a card the human is informed that a card is revealed but no specific details are displayed. The human player is provided with a sheet to record their information as seen in *Figure 2.1* and after each game they record the winner on the sheet. Once all the games have been completed for that number of players, the human (which due to the situation is always myself as explained in *Appendix C*) is shown the type for each of the artificial players involved in the games so the corresponding results can be recorded in *Table 5.4.* Each scenario is tested three times and the number of wins for each of the players is recorded in brackets beside the player type which is either *advanced* 'A', *basic* 'B' or *human* 'H', an 'X' is used to represent when the players are not involved in *Table 5.4.*

Test Number	Player 1	Player 2	Player 3	Player 4	Player 5	Player 6
1	H (1)	B (0)	B (2)	X	X	X
2	H (1)	A (0)	B (2)	X	X	X
3	H (1)	A (1)	A (1)	X	X	X
4	H (1)	B (1)	B (1)	B (0)	X	X
5	H (1)	A (2)	B (0)	B (0)	X	X
6	H (1)	A (1)	A (1)	B (0)	X	X
7	H (2)	A (1)	A (0)	A (0)	X	X
8	H (0)	B (1)	B (1)	B (0)	B (1)	X
9	H (2)	A (0)	B (1)	B (0)	B (0)	X
10	H (0)	A (2)	A (1)	B (0)	B (0)	X
11	H (1)	A (1)	A (0)	A (1)	B (0)	X
12	H (1)	A (1)	A (0)	A (0)	A (1)	X
13	H (1)	B (0)	B (0)	B (0)	B (1)	B (1)
14	H (1)	A (1)	B (1)	B (0)	B (0)	B (0)
15	H (1)	A (1)	A (1)	B (0)	B (0)	B (0)
16	H (1)	A (1)	A (0)	A (0)	B (1)	B (0)
17	H (2)	A (1)	A (0)	A (0)	A (0)	B (0)
18	H (1)	A (1)	A (0)	A (1)	A (0)	A (0)

Table 5.4 A record of which players won in each of the three games per test.

The results in *Table 5.4* provide further evidence that the *advanced* player strategy is generally more successful than the *basic* strategy since the solution was first reached by the *advanced* players 21/54 times compared to 14/54 times for the *basic* player. As discussed in *Appendix C*, these results may not truly reflect a standard human player because I have full knowledge of both strategies. For example, I could assume both types of player would not make deductions based on repeated use of cards in suggestions; despite this the *advanced* players solved the game more than the human (who won 19/54 games). Whilst playing the games it became apparent of an issue with both players, once they deduce which card is in the envelope for a particular category, in future suggestions they select one randomly. This can mean they are shown a card they have already seen so they gain no new information; changing this could improve the artificial players success rates, particularly against humans.

Chapter 6 – Conclusions

This chapter will assess if the aims and objectives have been met successfully as well as summarising the results from the tests presented in *Chapter 5* which are used to guide the plan for future work. Finally, a personal reflection is included which reviews my experience during the project process and how this will influence my approach to future challenges.

6.1 Review of Project Aims and Objectives

The aim of this project was to explore the possibilities of Artificial Intelligence to investigate strategies for the game of *'Speed Clue'* by planning a range of strategies and implementing a select few. In order to assess if this aim has been achieved, each of the objectives, originally outlined in *Section 1.1* are reviewed below:

1. **Investigate and design a variety of strategies that the artificial player can use with the aim of winning the game.**

   A literature review was conducted which considered the nature of the game '*Speed Clue'* and established the general aim of players to maximise their information gain whilst minimising their opponent's opportunity to gain knowledge. As this is a game of *imperfect information* which relies on *inference*, there are limited resources on specific strategies compared to games such as Chess. However, one paper [10] was identified that proposed a small number of strategies which were evaluated in *Section 2.3.3*, two of these formed the basis for the design of the artificial players which were implemented during the project. Analysis of Kingston's paper [10] as well as experience from playing the game enabled the design of the alternative strategies defined in *Section 4.4*. Therefore, the objective has been achieved as existing strategies have been investigated and a variety of alternatives have been designed in this report which are fundamental for the future work suggestions in *Section 6.3*.

2. **Produce a playable implementation of the game '*Speed Clue'* that allows for artificially intelligent and human players**

   This objective was achieved by the implementation of a platform which is described in detail in *Chapter 3*. The software allows one human player and a range of 2-6 artificial players to participate in a game of 'Speed Clue' which can have randomised or pre-defined deals of the cards. A text-based interface allows a user to setup a game and prompts a human player to interact with suggestions and responses.

3. **Implement and compare at least two of the strategies found from objective 1, test them against each other and humans**

From the research performed for objective 1, two strategies from literature were selected to be implemented and compared for this project. The design of both players based on these strategies is explained in *Sections 4.2 and 4.3* and their implementation was successful as demonstrated in the tests presented in *Section 5.1*. To enhance the conclusions regarding these strategies that can be found in Kingston's paper [10], further tests were conducted comparing the players against each other and human players (*Sections 5.2 and 5.3*). The results of these tests are evaluated in detail in *Section 6.2*, however, in summary, the implementation and testing accomplished this objective.

In conclusion, as the individual objectives were all successfully completed, I believe the main aim of this project has been met. A detailed review of existing strategies' successes and failures was used to design a range of strategies which could be combined to produce an artificial player that uses a mixed strategy approach, as explained in *Section 6.3*. Two of these strategies were built on to establish the artificial players that were implemented and compared against each other and humans. Although the human subject for the testing was myself, which may have impacted the results due to circumstances outlined in *Appendix C*, the artificial players could be tested against other humans in future work but the overall aim was accomplished.

6.2 Summary of Testing

The three sets of tests reviewed in *Chapter 5* conclusively demonstrate that on average, the *advanced* player is more successful than the *basic* player. Although both strategies were based on two from Kingston's paper [10], the first tests in *Section 5.1* were not identical to those in the paper [10, p.2] as expected. Further investigation highlighted that the approach taken by Kingston was systematic whereas the design of the strategies for this project relied on a random aspect to prevent opponent's (particularly humans) making deductions based on the order of suggestions. Therefore, as seen in *Section 5.1*, a new set of deals was constructed to test how the random approaches handled different situations and whilst the *advanced* player generally reached solutions in a shorter number of rounds, the strategies individually performed better in opposite deals. This supports the idea of a mixed strategy approach discussed in *Section 6.3*, however, the testing against a human player (*Section 5.3*) identified an improvement that could be made to both strategies before they are combined into a mixed strategy. Once either player has deduced the card in an envelope for a category, they then select one randomly in future suggestions and whilst this has the benefit of not revealing the information to other players of which cards they hold or are in the envelope, it often leads to opponents showing them cards again. As a human when I calculated which card was in the envelope for a category, I alternated between cards in my hand and cards I knew other players did not hold to tailor future suggestions so I could gain information about the other categories.

I predict that modifying the existing strategies to focus on gaining new information about relevant cards, without repeatedly using cards in their hand would improve the test results in *Sections 5.1 and 5.3*.

6.3 Future Work

Designing a range of strategies that could be implemented in future research was one of the main aims of this project because of the complexity of the logical deductions involved within '*Speed Clue*'. Therefore, the future work that could be done on this topic is extensive and this section outlines several options that could improve the artificial player which are based on the strategies from *Section 4.4* and the experience gained from the testing. Both the *basic* and *advanced* players were implemented successfully, however, evaluation of the tests against humans detailed in *Section 5.3* demonstrated that the artificial players required improvement to reduce the number of times a human won.

6.3.1 Improving the existing artificial players

Due to the design of the software for the programme, the issue discovered from the tests in *Section 5.3* can be fixed for the *advanced* and *basic* players by improving the method in the code base that handles selecting a random card from a category. Currently, for any category, if the card in the envelope is known then a random one is selected to form that part of the suggestion, but this is inefficient. For example, if the suspect card in the envelope is known and the card selected for the suggestion is a suspect that has already been shown by the next player, it is likely they will show this card again and the turn is wasted as no new information is gained. Instead the player should focus on trying to establish the remaining unidentified cards in the envelope, if they are using the *basic* strategy this would mean selecting a random card, which is not known to be in the envelope or in a player's hand, from the relevant categories. The *advanced* player would still ask about cards that are not held by the next two players, however, they would also only focus on the remaining categories which contain the unestablished envelope cards.

An alternative approach to selecting cards randomly in the categories with deduced envelope cards involves using the envelope card or cards in the players hand because it is guaranteed that their opponents cannot respond with them. As discussed in *Sections 4.4.1 and 4.4.3*, this risks the other players noticing the repeated use of these cards and making possible deductions without having to suggest about them. Furthermore, a safer option could be to extend the idea of the *advanced* strategy by aiming to suggest cards in these categories that are known to not be held by the next player. This would force the next player to show one of the other cards or reveal they do not hold any, ensuring that some information could be gained during a turn. If this is not an option due to knowledge of the next player's hand then the artificial player should default to suggesting a card in their hand or the envelope card for that category. Although this is still a risk, the probability of repeatedly including the same cards in suggestions is decreased and testing would need to be performed to see if this improves the artificial players success rates as expected.

After making these modifications to the strategies, the *advanced* player could also be developed to consider all opponents as suggested in *Section 4.4.2*. This approach would tailor suggestions based on the cards that are known to not be held by the most players instead of just the next player. In theory this could allow an artificial player to identify cards not held by any players with a higher probability, although this would need to be tested against the *advanced* player in this report for confirmation. I believe this development should follow from the modifications described above because in the scenario involving a card which is known to not be held by all players except one, it would be optimal to choose cards in the other categories so that the opponent's response must relate to that card. One option would be suggesting cards in the other categories that the current player holds, if this is possible. However, the risk of revealing information to other players about these cards will need to be outweighed by the possible information gained regardless of if the opponent holds the card or not and this would require complex testing.

6.3.2 Gaining more information from opponent's suggestions

As mentioned above one of the risks of using cards in a player's hand to form suggestions is that humans may notice repetitions, but this is not considered in either of the strategies developed for this project. In future work, this feature could be added to the *basic* and *advanced* strategies and tests would need to be performed to identify the optimum number of times a card must be suggested by an opponent before the artificial player records it in their knowledge base as possibly held. In order to make this information useful for the *basic* strategy, after an opponent has suggested the card more than the established number, a targeted suggestion would need to be made possibly with cards in hand instead of the normal approach. For the *advanced* strategy, if it was developed as suggested in *Section 6.3.1*, recording that the card was possibly held by a player would decrease the probability of it being included in a suggestion (as they are based on cards not held by players). These revised strategies could then be tested against the original results to evaluate if it improves their chance of winning a game.

Depending on the aim of future work, the method that an artificial player uses to update their knowledge at the start of each turn could be enhanced. Rather than just considering the suggestions and responses since their last turn, they could cross-reference them with all the previously recorded knowledge. Whilst updating the artificial players knowledge, if an opponent is now known to not hold card X and it was previously deduced that they must hold either card X or Y, then it is now clear that they must have card Y in their hand. It is unlikely that a human player can record this level of detail on their sheet, therefore, if the aim of future work is to develop an artificial player that replicates a human this should not be implemented. However, if the objective is to develop the most successful artificial player then upgrading the artificial player to consider this each turn would increase the number of deductions. Therefore, comparisons against the tests in *Table 5.1* would indicate if this would reduce the number of rounds that the player takes to reach a solution.

6.3.3 Mixed Strategy Approach

There is evidence, as explained in *Section 6.2*, that the most successful artificial player would use a mixed strategy approach. Before any alternative strategies are considered in future work, the implementation of an artificial player in the software would need to be adapted so they can change their strategy during the game based on certain criteria. In *Section 5.1,* the strategies from this report were compared against different deals and it became apparent from *Table 5.2* that their performance was impacted by the cards they held in their hand. Whilst playing the 60 games of '*Speed Clue'* required for the tests in *Section 5.3*, I noticed that I often changed the types of suggestions I made depending on my knowledge. For example, if there was a card which appeared to not be held by most players, I would include it in a suggestion, similar to the *advanced* approach. However, if I noticed that I had deduced most cards in a category except two, I would ask about the card which I suspected to be held by a player as their confirmation would imply that the other card was in the envelope. Furthermore, as well as changing the player's strategy depending on the hand they are dealt, there may be stages that are reached throughout the game which would prompt the artificial player to change their approach. Developing this option would allow combinations of the existing strategies to be trialled with new theories until the optimum artificial player is found.

6.4 Personal Reflection

Overall, I feel that the project has been successful as the aims and objectives were all successfully achieved, and a clear plan has been developed of future work which could be used by other students. I had not anticipated when starting this project how many deductions a human player of '*Speed Clue*' makes without intentionally thinking and this was one of the main challenges that I encountered. Often missing one of these logical deductions did not cause a standard coding error which is easy to detect, instead detailed step throughs of the code would have to be performed to check that the artificial player's knowledge changes as expected. In future projects that involve games of logical inference, I would advise playing a few games with human players and asking them to write down their thought processes which may help identify deductions and steps for the artificial player before coding the programme.

The initial timeline of the software implementation for this project, presented in *Appendix D***,** had to be modified in the first few weeks after I received feedback which suggested that the original project aims and objectives should be reconsidered. After consideration, I decided that a more efficient use of time would be to investigate and design a wide range of possible strategies and implement only two instead of extending the software to be applicable for '*Cluedo*'. Whilst adding the board to the game would present new challenges, there has already been extensive research into path finding algorithms which could be used by the artificial player for movement. However, as the literature review demonstrated, the aspect of logical inference in games has not been investigated as widely, so I deemed the exploration into options for strategies more innovative. This meant the allocation of time and resources had to be adjusted into the Gantt chart displayed in *Figure 1.1* which allowed the new objectives to be accomplished.

Despite this new timeline, unfortunately, I sometimes struggled with time management throughout this project as I found the structured plan difficult to implement in practice. Although I did perform the implementation steps of the software sequentially from creating the platform to adding a human player, I did not start this until much later in the project as planning the strategies took longer than anticipated. Also whilst I did test that both the *basic* and *advanced* players performed correctly after they were implemented, I did not conduct the tests discussed in *Chapter 5* until the end of the project as I realised it was easier to consider the impacts of all the results at the same time. For reasons discussed in *Appendix C*, the timeline of the project was delayed due to unexpected circumstances, so the software was completed later than expected, however, I maintained the weekly aims for implementing both the game and the players and found this ensured I was able to meet the objectives.

I thoroughly enjoyed the challenges of this project and would highly recommend any future students that have an interest in artificial intelligence to consider implementing the extensions suggested in *Section 6.3* or finding another game that involves logical deductions. It is fascinating to unpick the complex thought processes of humans and consider how an artificial player could implement them, although this can take longer than expected. I was satisfied with the outcome of this project having produced two variants of artificial players which perform as hoped. However if I had more time, I would have taken the opportunity to explore approaches that an artificial player could implement more successfully than a human, reviewing all previous suggestions and responses whenever their knowledge base was updated which would allow for more logical deductions.

List of References

1. Russell, S. and Norvig, P*. Artificial Intelligence : A Modern Approach*. Upper Saddle River, NJ: Prentice Hall, Incorporated, 1995.
2. Yannakakis, G. and Togelius, J. *Artificial Intelligence and Games*. [Online]. Cham, Switzerland: Springer International Publishing, 2018. [Accessed 1 November 2019]. Available from: http://gameaibook.org/book.pdf
3. Prisner, E. *Game Theory: Through Examples*. Washington, DC: Mathematical Association of America, 2014.
4. *The Stanford Encyclopaedia of Philosophy*. [Online] Spring 2019 Ed. s.v. Game Theory, 2019. [Accessed 4 December 2019]. Available from: https://plato.stanford.edu/cgi-bin/encyclopedia/archinfo.cgi?entry=game-theory
5. Dufwenberg, M. Game Theory. *WIREs Cognitive Science.* 2010, **2**(2), 1.
6. O'Neill, M and Summers, E. Collins English Dictionary. 7th ed. Glasgow: HarperCollins, 2015.
7. Turocy, T L. von Stengel, B. *Game Theory*. [Online] [no publisher] [Accessed 5 December 2019] Available from: http://www.cdam.lse.ac.uk/Reports/Files/cdam-2001-09.pdf
8. Morris, P. Non-Zero-Sum Games. In: *Introduction to Game Theory*. New York, NY: Springer, 1994, pp.115-147.
9. Samuel, A. L. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*. 1967 **11**(6), pp 601-617.
10. Kingston, J. Comparing Question Asking Strategies for Cluedo. In: Bryson, J., De Vos, M. and Padget, J. ed. *AISB Annual Convention Society with AI, 18-21 April 2017, Bath*. [Online]. [no place]: Artificial Intelligence for the Simulation of Behaviour, 2017, pp.332-335. [Accessed 4 February 2020]. Available from: http://aisb2017.cs.bath.ac.uk/conference-edition-proceedings.pdf

Appendix A
External Materials

This appendix contains a brief record of the materials used in this project that are not my own work.

A.1 The game of 'Cluedo'

This game was originally devised in 1943 by Anthony E. Pratt and is currently published by the company Hasbrp

A.2 The variation of 'Speed Clue'

This variant without the board was used as the basis for a competition and the rules were outlined by Joseph A. Craig and can be found at this website:

https://github.com/sadakatsu/SpeedClueContest/blob/master/speed_clue_rules.md

Although there are code samples available they use Java as the language, therefore, they were not applicable for this project. If future students wish to use Java or extend this project to allow for multiple players to participate via a server, this would be a useful resource, however, I used it for the rules only.

Appendix B
Pre-defined Card Deals

These deals are used for the testing performed in *Chapter 5*, deals 1-6 are based on those found in Kingston's paper [10, p.335]. The 3 player versions of deals 1-6 have been included and for the 6 player games the hands are split in half, so player 4 receives the latter 3 cards of player 1's hand, player 5 receives the latter 3 cards of player 2's hand and player 6 receives the latter 3 cards of player 3's hand. Whereas deals 7 and 8 are shown for all possible numbers of players, although the envelope cards remain the same regardless of the remaining deals.

B.1 Deal 1

Envelope : Professor Plum, Lead Pipe, Hall

Player 1 : Miss Scarlett, Colonel Mustard, Rope, Wrench, Ballroom, Library

Player 2 : Mrs Peacock, Reverend Green, Dining Room, Lounge, Study, Revolver

Player 3 : Doctor Orchid, Dagger, Candlestick, Billiard Room, Conservatory, Kitchen

B.2 Deal 2

Envelope : Professor Plum, Wrench, Kitchen

Player 1 : Miss Scarlett, Study, Library, Rope, Lead Pipe, Reverend Green

Player 2 : Lounge, Hall, Revolver, Mrs Peacock, Ballroom, Dining Room

Player 3 : Colonel Mustard, Dagger, Doctor Orchid, Billiard Room, Conservatory, Candlestick

B.3 Deal 3

Envelope : Doctor Orchid, Rope, Ballroom

Player 1 : Reverend Green, Colonel Mustard, Revolver, Lead Pipe, Hall, Library

Player 2 : Mrs Peacock, Professor Plum, Dining Room, Lounge, Study, Wrench

Player 3 : Miss Scarlett, Dagger, Candlestick, Billiard Room, Conservatory, Kitchen

B.4 Deal 4

Envelope : Miss Scarlett, Rope, Kitchen

Player 1 : Reverend Green, Colonel Mustard, Revolver, Lead Pipe, Hall, Library

Player 2 : Mrs Peacock, Professor Plum, Dining Room, Lounge, Study, Wrench

Player 3 : Doctor Orchid, Dagger, Candlestick, Billiard Room, Conservatory, Ballroom

B.5 Deal 5

Envelope : Mrs Peacock, Wrench, Dining Room

Player 1 : Miss Scarlett, Colonel Mustard, Rope, Lead Pipe, Hall, Library

Player 2 : Professor Plum, Reverend Green, Kitchen, Lounge, Study, Revolver

Player 3 : Doctor Orchid, Dagger, Candlestick, Billiard Room, Conservatory, Ballroom

B.6 Deal 6

Envelope : Miss Scarlett, Revolver, Hall

Player 1 : Reverend Green, Colonel Mustard, Rope, Lead Pipe, Ballroom, Library

Player 2 : Mrs Peacock, Professor Plum, Dining Room, Lounge, Study, Wrench

Player 3 : Doctor Orchid, Dagger, Candlestick, Billiard Room, Conservatory, Kitchen

B.7 Deal 7

Envelope : Miss Scarlett, Rope, Kitchen

B.7.1 3 Players

Player 1 : Professor Plum, Colonel Mustard, Mrs Peacock, Reverend Green, Doctor Orchid, Dining Room

Player 2 : Dagger, Wrench, Revolver, Candlestick, Lead Pipe, Lounge

Player 3 : Hall, Study, Library, Billiard Room, Conservatory, Ballroom

B.7.2 4 Players

Player 1 : Professor Plum, Colonel Mustard, Mrs Peacock, Reverend Green, Doctor Orchid

Player 2 : Dagger, Wrench, Revolver, Candlestick, Lead Pipe

Player 3 : Hall, Study, Library, Billiard Room

Player 4 : Dining Room, Lounge, Conservatory, Ballroom

B.7.3 5 Players

Player 1 : Professor Plum, Colonel Mustard, Mrs Peacock, Reverend Green

Player 2 : Dagger, Wrench, Revolver, Candlestick

Player 3 : Hall, Study, Library, Billiard Room

Player 4 : Dining Room, Lounge, Conservatory

Player 5 : Lead Pipe , Doctor Orchid, Ballroom

B.7.4 6 Players

Player 1 : Professor Plum, Colonel Mustard, Mrs Peacock

Player 2 : Dagger, Wrench, Revolver

Player 3 : Hall, Study, Library

Player 4 : Reverend Green, Doctor Orchid, Dining Room

Player 5 : Candlestick, Lead Pipe, Lounge

Player 6 : Billiard Room, Conservatory, Ballroom

B.8 Deal 8

Envelope : Miss Scarlett, Rope, Kitchen

B.8.1 3 Players

Player 1 : Professor Plum, Colonel Mustard, Wrench, Lead Pipe, Hall, Billiard Room

Player 3 : Mrs Peacock, Doctor Orchid, Revolver, Dining Room, Study, Conservatory

Player 3 : Reverend Green, Dagger, Candlestick, Lounge, Library, Ballroom

B.8.2 4 Players

Player 1 : Professor Plum, Colonel Mustard, Candlestick, Hall, Conservatory

Player 2 : Mrs Peacock, Dagger, Lead Pipe, Study, Ballroom

Player 3 : Reverend Green, Wrench, Dining Room, Library

Player 4 : Doctor Orchid, Revolver, Lounge, Billiard Room

B.8.3 5 Players

Player 1 : Professor Plum, Dagger, Dining Room, Billiard Room

Player 2 : Mrs Peacock, Wrench, Lounge, Conservatory

Player 3 : Reverend Green, Revolver, Hall, Ballroom

Player 4 : Doctor Orchid, Candlestick, Study

Player 5 : Colonel Mustard, Lead Pipe, Library

B.8.4 6 Players

Player 1 : Professor Plum, Wrench, Hall

Player 2 : Mrs Peacock, Revolver, Study

Player 3 : Reverend Green, Candlestick, Library

Player 4 : Doctor Orchid, Lead Pipe, Billiard Room

Player 5 : Colonel Mustard, Dining Room, Conservatory

Player 6 : Dagger, Lounge, Ballroom

Appendix C
Impacts of COVID-19

The main impact that COVID-19 had on my project was preventing possible testing on human participants as had originally been planned to meet the final objective. The code was originally designed so that I would setup the game (controlling the number and types of artificial players) and a human would then play the game, unaware of the strategies being used. I adapted the program so that I could setup the game by selecting the number of artificial players and the system would then randomly run all possible combinations of the types of artificial players as shown in *Table 5.4*. Although this ensured I did not have direct knowledge of what strategies the opponents were using, I could sometimes deduce it from the types of suggestions they were making. Furthermore, since I knew how these suggestions were formed, I could also guess some of the information that they held to make these suggestions. Whilst I tried to be impartial by not noting down this knowledge, it will have potentially impacted the results meaning that although the issues I noticed are relevant, the specific data is flawed. Ideally, participants would have played these games without this knowledge and in future research this would be important to investigate.

Appendix D
Original Gantt Chart

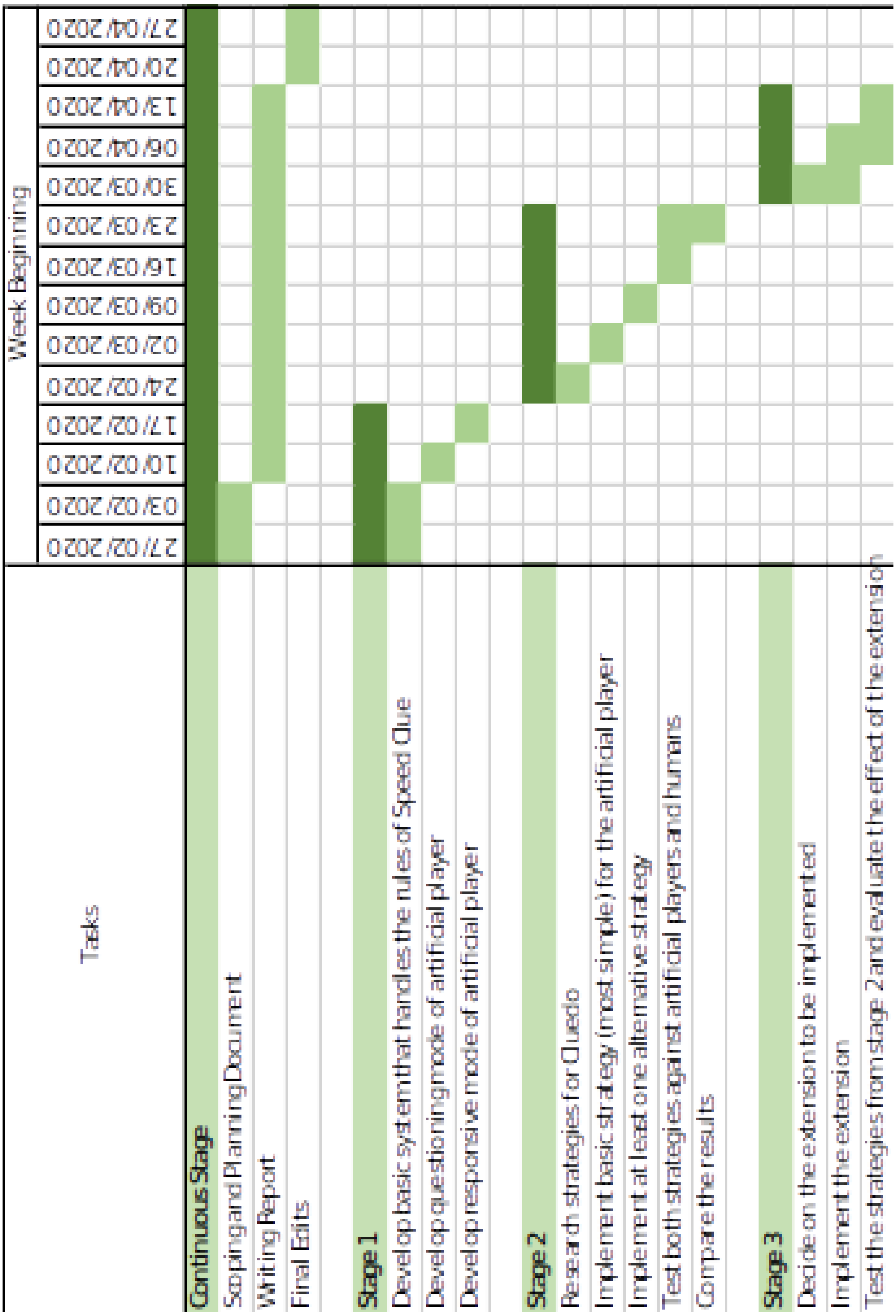

Figure D.1 – Original Gantt Chart representing Initial Project Timeline