The Online Mastermind Game

Cheong Kuen Leion

WEK000412

Artificial Intelligence Department

Approved by:

Supervisor: Dr. Rukaini Haji Abdullah

Moderator: Mr. Woo Chaw Seng

Abstract

This online Mastermind game project is developed to give the chances to people from all around the world to play with each other though they are not known each other. Besides that, this project is also to test the effectiveness of the technique, which is called as Genetic Algorithm on solving a problem. In the other words, the aim of this project is to develop an online Mastermind game using genetic algorithm between a computer and a player or (ideally) a group of players over the network. This game will let the player feel more comfortable and they can imagine that they are playing with another virtual player. Besides that, the player can even know and practice how the computer system is going to solve the puzzles by using the genetic algorithm by showing some hint to the player during the game. This game shows the basic concept that is used in solving problems.

Acknowledgement

The path that I have gone through in completing this study is not always smooth and easy. However, with the help and guidance of countless people has made this study possible.

I take this opportunity to express my deepest gratitude to Dr. Rukaini, my supervisor for his inspiration, valuable guidance and constructive criticism throughout the study. She has been very patient in giving valuable advice and comment on my drafts, which finally brought this study to an end.

The process of writing thesis could be extremely painful. But with the help and support of fellow friends, I finally overcome the fear and insecurity. Thus, to my friends, I express my sincere appreciation.

Finally, I apologize for any mistakes present in the following account, which I assume fully responsibility.

Best regard,

_____

(Cheong Kuen Leion)

WEK000412

Content

List of Table

List of Figure

Chapter 1 – Introduction

1.1 Overview

The complexity of problem solving of the human mind is one of the research areas of Artificial Intelligence (AI). The universally accepted definition of Artificial Intelligence (AI) is the study of how to make a computer to do things, which at the moment people do better. Some of the early work in the field focus on the formal task such as game playing and theorem proving. The games were implemented the logical thinking method which is also used by human minds. Games and theorem proving are considered as intelligence if it can also be done by people.

Through the years since Artificial Intelligence is discovered, more and more research has been done to discover the techniques in problem solving. One of the researches that were developed to test the problem solving technique is through the classical games. In this method, users are asked to play in single or multiplayer mode. This allows the users to interactively play with the system and find out the technique of problem solving.

1.2 Objectives

The main objectives of the project are:-

- To introduce the technique used in AI, which can be practically used in problem solving (Genetic Algorithm)
- To test the effectiveness of the techniques when used in the game.

- To seek the intelligence of the game in the process of solving the games.
- To teach the human players the strategies of the game.
- To use the present technologies, this can be implemented in network games (JAVA, My SQL etc).

1.3 The Concept of the Game

The structure of the game wholly based on the online games that existed in the internet nowadays. It has a user-friendly interface, which will let the users to easily understand about its features and facilities. This game is considered as a classical game since it is well known and has been played by most of the people from around the world for decades. The concept of this game is simply and easily understood.

Basically, this game has three modes of game play. This includes:-

- User against computer (system).
- User against another user in the same computer.
- User against another user in a different computer.

User can choose whether he/she want to play with the computer system or with another user through the same computer or as online with different computer.

Before the user can start playing the game, user has to log in first. This means that user has to fill in his/her ID name and password in order to let the system to check the authentication and trace the user's profile and history. This feature will help the system and the user to follow up the performance of the user by keeping the detail and history of the user. Each time the user has finish the game, the system will automatically save his/her performance into a database system.

In order to get an ID name and a password, the user or player need to sign up by filling in a particular form about their personal information. This information will be kept into a database. Player can choose whether to put his/her personal information to the public or just keep it for the system administration.

When the player has logged-in, he/she has to choose the playing mode (single, 2 players or multiple players). After he/she has chosen the game mode, the player is requested to choose the setting of the game. The player has to decide the number of the pins for the color and marker, number of rows (optional) and set of games to be played. Once the setting is done, the system will take the first move by being the code creator. As the opponent of the system, the player should be the code breaker. The next turn when the position has changed which mean that the computer system is the code breaker, the system will pop out some sentences at a side of the window to show the hint on how the system solve the combination. The purpose to show this hint is to let the user learn on how to solve problem by using genetic algorithm. While the 2 players mode, the first player will take the first turn as code creator and of course the second player will

be the code breaker. In the multiplayer mode, the player who starts the game will be the code creator and the players who join the game will be the code breaker.

During the game, the code creator will make a combination of colors depend on the setting that has been done. After the code creator has decided the color combination, the second player (code breaker) has to start his/her move. The code breaker will choose the color for the first pin of the first row and continue to the next pin in the same row. After the player has decided his/her color combination, the system will automatically place the marker pin to the correct the available color pin in the mark columns. By using this information, the code breaker will try to set a new color combination to match the color combination that set by the first player (code creator). The code breaker will use his/her skill and intelligence to set this new color combination to put into the next row. The intelligence and tactical skills that used by the code breaker will determine the success of him/her in solving the color combination. The efficiency of the technique and the level of intelligence of the players are shown by the number of the rows that are used to solve the combination.

Each time the color combination solved, marks will be given. The players will then continue to play the second round after the color combination has been solved or the number of rows has up to the end. This time the player will change side, which means the first player will become the code breaker and the second player as the code creator and take the first move. The game will continue until

the end according to the setting that has been set before the game or one of the players leave the game.

At the end of the game, the system will automatically inform the player who is the winner and depend on the scores that the players have. The player can review the tactic used, time used for each row and the movement used by him/her at the end of the game. This is because this information has been automatically recorded in the system database. The reason to keep records of this data is to let the players to measure their intelligence according to their techniques and performance. The user also can compare their performance through a number of games that they played. After each game, users will be asked to continue with a new game or quit form the game.

1.4 Project Scope

This project is to develop a simple classical game (invented since 1970) that uses an AI technique that can be shared and played over the internet. It can be a medium of leisure or entertainment without losing its purpose to test the efficiency of the technique implemented in the game to the user or player.

While the users play this game, they will face some challenges on guessing the combination color that is created. In order to solve this problem, the user will use some different tactics and skills. Continuing playing this game will make the user become more advance and intelligence to solve the code. The record of the users'

achievement will be as good as references for the users. From the record, the player can learn his/her mistake and try to improve it.

From this game, the users can know the level of their problem solving skills, which tactic should be most efficient to solve a puzzle, the performance of their techniques and learn the best technique o solve the color combination.

1.5 Project Planning Schedule

Table 1.1: Project Schedule

ID	Task Name	Duration	Start	Finish
1	Get the thesis title	1 day	Thu 6/6/02	Thu 6/6/02
2	Check the previous thesis as reference	10 days	Fri 6/7/02	Thu 6/20/02
3	**Requirement Gathering**	**26 days**	**Fri 6/28/02**	**Fri 8/2/02**
4	Proposal of the approach	5 days	Fri 6/28/02	Thu 7/4/02
5	Execute requirement gathering	21 days	Fri 7/5/02	Fri 8/2/02
6	Documentation	11 days	Fri 7/19/02	Fri 8/2/02
7	**Analysis**	**12 days**	**Thu 8/1/02**	**Fri 8/16/02**
8	Resource feasibility	3 days	Thu 8/1/02	Mon 8/5/02
9	Technical feasibility	4 days	Tue 8/6/02	Fri 8/9/02
10	Schedule feasibility	5 days	Mon 8/12/02	Fri 8/16/02
11	**Design**	**20 days**	**Mon 8/19/02**	**Fri 9/13/02**
12	User Interface	6 days	Mon 8/19/02	Mon 8/26/02
13	Tutorial Interactive movement	3 days	Tue 8/27/02	Thu 8/29/02
14	Database design	6 days	Fri 8/30/02	Fri 9/6/02
15	Prototyping	5 days	Mon 9/9/02	Fri 9/13/02
16	Training on coding, database	10 days	Mon 9/16/02	Fri 9/27/02
17	**Implementation**	**41 days**	**Mon 9/30/02**	**Mon 11/25/02**
18	Design and coding	21 days	Mon 9/30/02	Mon 10/28/02
19	Setup database system	16 days	Tue 10/29/02	Tue 11/19/02
20	Coding review	4 days	Wed 11/20/02	Mon 11/25/02
21	**Testing**	**14 days**	**Tue 11/26/02**	**Fri 12/13/02**
22	Testing	5 days	Tue 11/26/02	Mon 12/2/02
23	Bugs fix	9 days	Tue 12/3/02	Fri 12/13/02
24	Maintainant plan documentation	11 days	Mon 12/16/02	Mon 12/30/02
25	Final review	2 days	Wed 1/1/03	Thu 1/2/03
26	Pass up thesis	1 day	Tue 2/18/03	Tue 2/18/03

Chapter 2 – Literature Review

2.1 Introduction

Literature review is a process of assembling the information about the project and its development. The development of the game concentrates on the efficiency of the techniques used in the game without losing its benefits and excitement. It is meant to develop the level of thinking of player. The technique used in the development is one of many technique found by the AI researchers. The technique has been implemented in a few system especially games because it is easier to understand and probing capabilities of solving problems.

2.2 Artificial Intelligence (AI)

2.2.1 Problems and AI

Game playing and theorem proving share the property that human who do them will is considered to be displaying intelligence. There is a thought that computer can perform better than human by exploring a large number of solution path and then selecting the best one. But this assumption turns out to be false since no computer is fast enough to overcome the combinatorial explosion generated by most problems.

AI focuses on the sort of problem on decision to be made in life routines such as what to buy for birthday present, what to wear to participate in a party, what to eat for breakfast and many others. This type of reasoning is called common sense

reasoning, which includes reasoning of physical objects and their relationships to each other.

To handle larger amount of information and knowledge, some progress was made on the mentioned task. These included perception, natural language understanding and problem solving in specialized domain such as medical diagnosis and chemical analysis.

A person who knows how to perform task from several categories learns the necessary skills in a standard orders. First, perceptual, linguistic and common sense skills are learned followed by expert skills such as engineering, medicine and finance. It might seem to make sense that earlier skills are easier and this more amenable to computerized duplication that the specialized or later ones. For this reason, much of the initial AI work was concentrated in those early areas. But it turns out that this assumption is not right. Although expert skills require much knowledge than do the more mundane skills ad that knowledge is usually easier to represent and deal with inside programs. As a result, the problems area where AI is now flourishing most as a practical discipline are primarily the domains that require only specialized expertise without the assistance of common sense knowledge.

2.2.2 AI Techniques

An AI technique should be represented in such a way that:-

- The knowledge captures generalizations. If knowledge does not have this property, inordinate amounts of memory and updating will be required
- It can be used in many ways although it is not totally accurate or complete
- Humans can understand it
- It can overcome the sheer bulk by narrowing the range of possibilities that must usually be considered
- It can easily be modified to correct errors and reflect changes in the world and the world's view

There are three important AI techniques

- Search

  Provides a way of solving problems for which no more direct approach is available as well as a frame work into which any direct techniques that are available can be embedded

- Use of knowledge

  Solving problems by exploiting the structure of the objects that are involved

- Abstraction

  Separating the important features and variation from the many unimportant ones that would otherwise overwhelm any process

2.3 Problem Spaces And Search

There are four things that should be done in order to build a system to solve a particular problem: -

i. Define the problem precisely.

ii. Analysis the problem.

iii. Isolate and represent the task knowledge that is necessary.

iv. Choose the best problem solving technique to solve problems.

2.3.1 Define Problem

A single statement is not enough as human tends to have their own definition and it is very incomplete statement of the problem which to be solved. Specification should take places from the general statements such as rules of engagement, perimeters, instances and others. In addition, explicit move must be taken towards the previously implicit goal, which accumulates the benefits.

There are two serious practical difficulties by using too many rules: -

i. No people could ever supply a complete set of such a rule. It would take too long and could certainly not be done without mistakes.

ii. No program could easily handle all the rules.

In order to minimize the problems, riles should be written as general as it can be. It is useful to introduce some convenient notation for describing patterns and substitutions.

The state space representation forms the basis of most AI methods. Its structure corresponds to the structure of problem solving in two important ways: -

i. It allows for a formal definition of a problem as the need to convert some given situation into some desired situation using a set of permissible operations.

ii. It permits the definition the process of solving a particular problem as a combination of known technique, which is each of them is represented as a rule defining a single step in the space and the search, which is the general technique of exploring the space to try to find some path from the current state to a goal state. Search is a very important process in the solution of hart problems for which no more direct techniques are available.

In order to provide a formal description of a problem, there are things to be considered: -

- Define a state space that contains all the possible configurations of the relevant objects and the impossible ones it can. It is possible to define this space without explicitly enumerating all of the states it contains.
- Specify one or more states that would be acceptable as the solutions for the problem. These states are called goal states.
- Specify one or more states within that space that describe possible situations from which the problem solving process may start. This state is called the initial state.
- Specify a set of rules that describe the actions or operators that available. Doing this will require giving though to the following issues: -

i. The unstated assumptions that present in the informal problem description.

ii. The broadness of the rule.

iii. Number of workload that are required to be computed and represented in the rules.

2.3.2 Control Strategies

The strategy is to decide which rule to apply during the process to search for solution to a problem. The first requirement of a good control strategy is that it causes motion. Control strategies that do not cause motion will never lead to a solution. The first requirement of a good control strategy is that it must be systematic. It corresponds to the need for global motion, which is over the course of a single step.

There are some techniques or algorithms that can be used as control strategies: -

- Breadth-First Search
    - Create a variable called node list and set it to the initial state.
    - Until a foal state is found or node list is empty, do: -
        - Remove the first element from node list and call it E. if node list was empty, quit.
        - For each way that each rule can match the state described it in E, do: -
            - Apply the rule to generate a new state.
            - If the state is a goal state, quit and return this state.

- Otherwise, add the new state to the end of node list.

- Depth-First Search
  - From the existed tree, s single branch of the tree is pursued until it yields a solution or until the decision to terminate that path is made. It makes sense to terminate a path if it reaches a dead-end, produces a previous state or becomes longer than some pre-specified utility limit. In such a case, backtracking occurs. The most recently created state from which alternative moves are available will be revisited and a new state will be created. This from of backtracking is called *chronological backtracking* because the order in which steps are undone depends only on the temporal sequence in which the step is always the first to be undone. This form of backtracking is what the simple term backtracking usually means.

  - The process can be described precisely as
    - If the initial state is a goal state, quit and return success.
    - Otherwise, do the following until success or failure is signaled
      - Generate a successor, E of the initial state. If there are no more successors, signal failure.
      - Call Depth-First Search with E as the initial state
      - If success is returned, signal success. Otherwise continue in this loop.

- Heuristic Search
    - In order to solve many hard problems efficiently, it is often necessary to compromise the requirements of mobility and systematically and to find the best a control structure that is no longer guaranteed to find the best answer but that will almost always find a very good answer. This will lead to the idea of a heuristic. A heuristic is technique that improves the efficiency of a search process that possibly by sacrificing any claims to completeness. Heuristic is like a tour guide. They are good to the extent that they point in generally interesting directions but they are bad to the extent that they may miss point of interest to particular individuals. Some heuristic help to guide a search process without sacrificing any claims to completeness that the process might previously have had. Others in fact many of the best ones may occasionally cause an excellent path to be overlooked but on the average, they improve the quality of the paths that are explored. A good solution, which possibly non-optimal to hard problems are hope to achieved when using good heuristic. There are some good general-purpose heuristics that exploit domain-specific knowledge to solve particular problems. One example of a good general purpose heuristic that is useful for combinatorial problems is the nearest neighbor heuristic, which works by selecting the locally superior alternative at each step. For general-purpose heuristics, it is often possible to prove error bounds, which provides reassurance that one is not paying too high a price in accuracy for speed.

- In many AI problems, it is not possible to produce such reassuring bounds. This is true for two reasons: -
  - It is often hard to measure precisely the value of a particular solution in the real world problems.
  - It is often useful to introduce heuristics based on relatively unstructured knowledge for real world problems. It is often impossible to define this knowledge in such a way that a mathematical analysis if its effect in the search process can be perform.

- There are many heuristics that although they are not as general as the nearest neighbor heuristic, they are nevertheless useful in a wide variety of domains. Without heuristics, people would become hopeless ensnarled in a combinatorial explosion. This alone might be a sufficient argument in favor of their use. But there are other arguments as well, which are: -
  - Rarely every situation needs the optimum solution but a good approximation will usually serve very well. In fact, there is some evidence that people when they solve problems, are not optimizers but rather are satisfiers.[1]

- A heuristic function that maps from problem state descriptions too measure the desirability, whish is usually represented as numbers. Well-designed heuristic function can play an important part in efficiently

[1] Simon, H.A. 1981. The Sciences of the Artificial, 2nd Ed. Cambridge, MA:MIT Press

guiding a search process toward a solution. Sometimes very simple heuristic function can provide fairly good estimate of whether a path is any good or not.

- Few of the heuristic search techniques are: -
  - Generate a possible solution. For some problems, this means generating a particular point in the problem space.

- For others, it means generating a path from start state: -
  - Test to see if this is actually a solution by comparing the chosen point or the endpoint of the chosen part to the set of acceptable goal states.
  - If a solution has been found, quit. Otherwise, return to step 1.

- If generation of possible solutions is done, then this procedure will find a solution eventually, if one exists. Unfortunately, if the problem space is very large, eventually it would take very long time.

- Hill Climbing
  - This technique is a variant of generate-and-test in which feedback from the test procedure is used to help the generator decide which direction to move in the search space. In a pure generate-and-test procedure, the test function responds with only a yes or no, but if the test function is augmented with a heuristic function that provides an estimate of how

close a given state is to a goal state, the generate procedure can be exploited.

- This technique is always being used when a good heuristic function is available for evaluating states and when there has no other useful knowledge is available. The algorithms that describe simple hill climbing are: -
  - Evaluate the initial state. If it is also a goal state, then return it and quit. Otherwise continue with the initial state as the current state.
  - Loop until a solution is found or until there are no new operators left to be applied in the current state. Then
    - Select an operator that has not yet been applied to the current state and apply it to produce a new state.
    - Evaluate the new state
      - If it is a goal state, then return it and quit.
      - If it is not a goal state but it is better than a current state, then make it the current state.
      - If it is not better than the current state, then continue the loop.

- Simulate Annealing
  - Simulated annealing is a variation of hill climbing that at the beginning of the process, some downhill moves may be made. This is to make sure the

process do enough exploration of the whole space early on. This feature is to make sure the final solution is relatively insensitive to the starting state.

- Simulated annealing as a computational process is patterned after the physical process of annealing that physical substances such as metals are melted meaning that rose to high energy level and then gradually cooked until some solid state is reached.[2] Physical substance usually moves from higher energy configurations to lower ones. These will make the valley descending occur naturally.

- In the physical valley descending that occurs during annealing, the probability if a large uphill move is lower than the probability of a small will occur and this probability is represented by the following function: -

$$P = e^{-\Delta e/kT}$$

Δe, positive change in the energy level

T, temperature

K, Boltzmann's constant

- There is a rate at which the system is cooled and this condition is called the annealing schedule, which is very sensitive to the physical annealing process. If cooling occurs too rapidly, stable regions of high energy will form. If however, a slower schedule is used, a uniform crystalline

[2] Kirk Patrick, S., Gelatt, Jr., C.D., and M.P. Vecchi. 1983, Optimization by simulated annealing: Science 220 (4598)

structure, which corresponds to a global minimum, is more likely to develop.

- The algorithms of simulated annealing are: -
  - Evaluate the initial state. If it is also a goal state, then return it and quit. Otherwise, continue with the initial state as the current state.
  - Initialize BEST-SO-FAR to the current state.
  - Initialize T according to the annealing schedule.
  - Loop until a solution found or until there are no new operator left to be applied in the current state.
    - Select an operator that has not yet been applied to the current state and apply it to produce a new state.
    - Evaluate the new state, compute ΔE = (value of current) – (value of new state)
      - If the new state is a goal state, then return it and quit.
      - If it is a goal state but is better than the current state, then make it the current state. Also set BEST-SO-FAR to this new state.
      - If it is not better than the current state, then make it the current state with probability p. This step is implemented by invoking a random number generator to produce a number in the range [0,1]. If

the number is less than p, then the move is accepted. Otherwise do nothing.

- Revise T as necessary according to the annealing schedule.

- Return BEST-SO-FAR as the answer.

- There have three components at annealing schedule: -
  - Initial valve to be used for temperature.
  - Criteria that will be used to decide when the temperature of the system should be reduced.
  - The amount by which the temperature will be reduced each time it is changed.

2.4 Game Playing

Charles Babbage, the 19th Century computer architect, thought about programming his Analytic Engine to play chess and later of building a machine to play tic-tac-toe.[3] Chude Shannon wrote a paper in which he described mechanisms that could be used in a program.[4] A few years later, Alan Turing described a chess-playing program. His program played checkers and in addition to simply playing could learn from its mistakes and improve performance.[5]

---

[3] Bowden, B.V. 1953. Faster Than Thought. London: Pitman

[4] Shannon, C. E. 1950. Programming a computer for playing chess. *Philosophical Magazine* [series 7] 41: 256-257

[5] Samuel, A. L. 1962. some studies in machine learning using the game of checkers. In Computers And Thought, Ed. E.A. Feigenbaum and J. Feldman. New York: McGraw-Hill

2.5 Present Game System Analysis

i) http://www.swissdigital.com/mastermind /mastermind.cfm?sessionint=yes

The game system has a database for keeping the record of the players' name and only for those who has quite high marks. The user can only player the role as code breaker. This means that the player can only be the one who guess the combination code generated by the computer system. And the player has no right to be the code maker.

This system was developed by Patrick Schipper for SwissDigital GmbH. The game will start as soon as the player clicks on the start button. By using the radio buttons, the player will choose the designated color pegs to be put on the board and click the set button. There is also a reminder showing how many lines are still remaining to be used by the player.

ii) http://geneura.ugr.es/~jmerelo/Genmm/Genmm.shtml

This system is using the genetic algorithm technique. It was developed by Geneura Team from the Department of Computer Architecture and Technology, Granada University. This program used an evolutionary algorithm to find the solution. It keeps a population of solutions, which are ranked according to their consistency to the answers (black and white pegs) to the guesses already made. When the algorithm finds a solution that is consistent with all guesses, it is played. The system is programmed using C++ EO class library. The game itself is an example of evolving objects, that is the solution to the mastermind problem itself.

The structure of the game is initially user will input eight digits in the range of 1 to 9. Each digit will represent a color combination that will be break by the system.

iii) **http://www.javaonthebrain.com/java/mastermind/mastermind.html**

This game is programmed by using Javascript. Same as the upper system, the player of this system can only be the code breaker. User has to click on the start button before he or she can start the game. After the start button has been clicked, the system will randomly generate a combination code. User then has to start guessing the combination color created by the system. After the user set the input, the system will then mark the combination. The user can only know the color combination created by the system during the end of the game. In other words, the user can only see the combination color if the user win or lose the game.

2.5.1 Similarity of the Present Game System

The game systems that are used in the analysis for this project have some similarity. The similarity that mentions are: -

- Simple interface but easy to understand. Though each of the system has its own interface but each of them has the same purpose, which is user-friendly.
- Using different method to represent the pegs. Some are using the radio buttons to represent the pegs and some are using the color buttons.
- Using different programming language to develop the game system.
- Will wait the user has made up decision before processing to the next row of the game.

2.5.2 Advantages of the Present Game System

The advantages of the existed system that will also be implemented into the project system are: -

a. Interactive play among users or players via the Internet.

b. User-friendly interface.

c. Login interface with password to recognize the user.

d. A database system to record the player's information and their played record.

e. Unlimited row to let the player to set the agreement of the game before the start of the game.

Chapter 3 – System Domains

3.1 History Of The Game (Mastermind)

Mordecai Meirowitz, an Israel telecommunication expert, invents this game in 1970-1971. During the International Toy Fair at Nuremberg in February 1971, he showed the game to a small English company called Invicta Plastics Ltd. The small Leicester based company bought up the property rights and under the guidance of its founder, Mr. Edward Jones-Fenleigh refined it and released it in 1971-1972.

In 1973, this game won the first ever Game of the Year Award. The game also received a Design Centre Award and the Queen's Award for Export Achievement. Until now, it has been sold over 55 million copies in 80 countries around the world. This product of game is still being made and sold today.

3.2 A Look Into The Game

Minimum two players play this game. One player will be the code breaker and the other one will be the code maker. As a code breaker, the player has to find the solution for the color combination set by the code maker in ten guesses. In the classic game, it has been set to guess the color combination within ten rows. So, the player (code breaker) has to succeed to guess the color combination within that ten rows, otherwise the player is considered lose the game.

While for the code maker, this player has to set a tow of color combination to let the code breaker to guess. There has six different colors and the code maker has to make a row of four pegs with different or same colors depend on the player creativity.

When the code breaker successfully breaks the color combination set by the code maker, points will be count according to the amount of rows that the code breaker used to break the code. The fewer rows used to guess the color combination, the more percentage for the player to win the match.

3.3 Game Operational Study

To play this game, six colors of pegs are used to create a combination of four colors. That means, there are 1296 ways of choosing the four pegs: -

$$6 \text{ X } 6 \text{ X } 6 \text{ X } 6 = 6^4 = 1296 \text{ ways}$$

There are 14 different ways of marking the guess according to the input: -

[0] No marker

[1] 1 white marker

[2] 1 black marker

[3] 2 white markers

[4] 1 black marker and 1 white marker

[5] 2 black markers

[6] 3 white markers

[7] 1 black marker and 2 white markers

[8] 2 black markers and 1 white marker

[9] 3 black markers

[10] 4 white markers

[11] 1 black marker and 3 white markers

[12] 2 black markers and 2 white markers

[13] 4 black markers

To start a first guess of the color combination, we don't have any information or clue to know what colors are in the solution. So, one might think that any color will do the same result. But this has been proved to be false. For instance, if the first guess is 4 yellow pegs, it can only be marked in five ways (only the ways that use no white markers, that are [0], [2], [5], [9], and [13]). However, if the first guess uses more than 1 color (e.g. BBCC), then it can be marked in several ways, which gives more logical information about the solution.

There are only five essentially different first moves: BBBB, BBBC, BBCC, BBCG, and BCGR. Before the player plays the first guess, there are 1296 possible solutions. After the first guess is marked, the number of possible solution will decrease. Computer analysis shows the number of solution left in each case as in Table 3.1.

Table 3.1: Number of solutions

First guess	BBBB	BBBC	BBCC	BBCG	BCGR
Mark [0]	625	256	256	81	16

Mark [1]	**0**	**308**	**256**	**276**	**152**
Mark [2]	**500**	**317**	**256**	**182**	**108**
Mark [3]	**0**	**61**	**96**	**222**	**312**
Mark [4]	**0**	**156**	**208**	**230**	**252**
Mark [5]	**150**	**123**	**114**	**105**	**96**
Mark [6]	**0**	**0**	**16**	**44**	**136**
Mark [7]	**0**	**27**	**36**	**84**	**132**
Mark [8]	**0**	**24**	**32**	**40**	**48**
Mark [9]	**20**	**20**	**20**	**20**	**20**
Mark [10]	**0**	**0**	**1**	**2**	**9**
Mark [11]	**0**	**0**	**0**	**4**	**8**
Mark [12]	**0**	**3**	**4**	**5**	**6**
Mark [13]	**1**	**1**	**1**	**1**	**1**

For instance, if the first move is BBBC, and the mark is [3] (= 2 white markers), then logically there are only 96 possible solutions left. As a special case, note that if the first move is BBCC, and the player is lucky enough to get marked [10] (= 4 white markers), then there is only one possible solution, CCBB, so the player win in just 2 moves.

If the player chooses BBCC as a first move, then no matter how unhelpful it is marked, it still have at most 256 possible solutions left. The player chooses to play BBCC because it gives the player the smallest peak of figure. If the player uses BBCG as the first move, then the most possible peak figure for the next row is 276 solutions. This method turns to

be a good strategy for choosing the best move. After the second move is marked, the analysis shows that the player are left with at most 46 possible solutions. Repeating this procedure for each of the remaining guesses gives rise to the strategy table.

Chapter 4 – Methodology

4.1 Methodology of System Development

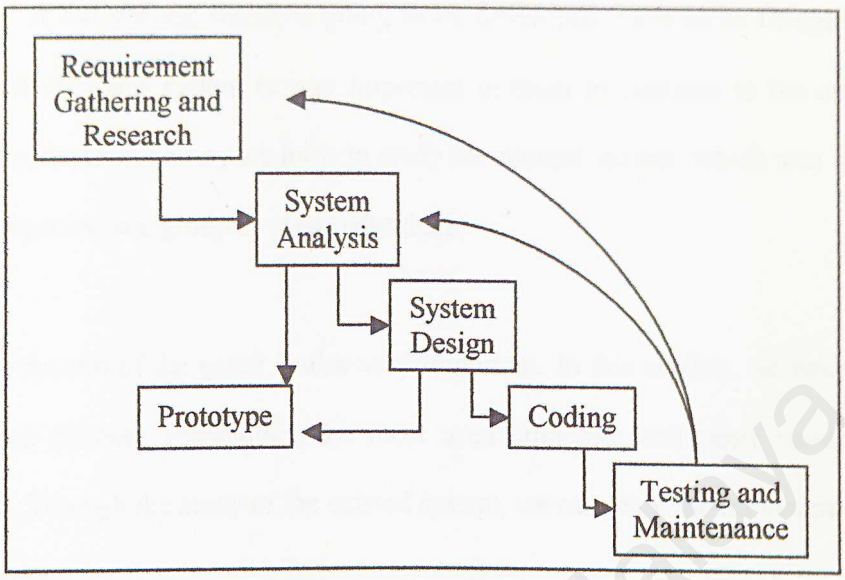

Figure 4.1: Waterfall and Prototype Model

To develop the game system, I choose waterfall and prototype modeling. Choosing this model has its own advantage. The advantages of choosing this model are: -

a. Easy to recognize important level.

b. Provide the overview of the whole activity.

c. Easy to allocated project schedule.

d. Make sure the system implements all the requirements and every function of the system can be detected.

e. Prototype controls every level of the process in order to define other design strategies.

f. A test can be done to ensure the right performance of every function.

4.2 Requirement Gathering and Research

In this phase, a lot of researches have to be done. First of all, the problems, motives and objectives of the system, which is going to be developed, have to be recognized. The understanding of this system is very important in order to continue to the next phase. Besides the mention above, we have to study the present system, which was developed by other organization, group or even individual.

Study the domain of the game is also very important. In this session, we have to study the original product, techniques, the most used strategies and other feature that is important. Through the study of the existed system, we can learn the improvement of the game.

4.3 System Analysis

4.3.1 Architecture Analysis

The system that is going to be developed has divided into four main modules: -

a. User Module

This module includes: -

- User profile
- User login system
- User sign up (register)
- Keep track of the user's achievement

b. Interface Module

In this module, the important feature that will be count on is: -

- How the game board is going to present so that it will look more interesting and interactive.
- How to make the system suitable for all kind of user around the world (user-friendly system and easy to understand)

c. Play Module

- Single player mode
- In the single player mode, the computer will show the hint on how the system solving the problems. This hint will be presented in a text form.

d. Engine Module

This module shows how the logical execution being happen and also keep all the help files and manual to the user.

4.3.2 Software Analysis

i. JAVA

- Develop Graphic User Interface (GUI)
- Portability where applications will run without modification on a variety of platforms.
- Can communicate with other applications.
- Can integrate Internet components and remote databases as well.

- Has the multimedia capability of graphics, images, animation, audio and video.
- Can run on the Internet and the www.
- Can be written quickly and correctly in a manner that takes advantage of pre-built software components.
- Can take advantage of the flexibility and the performance improvement of multi-reading.
- Richer file processing.
- Can be used to develop engine module and some of the interface module.

ii. PHP

- Provide the facilities to major database management system.
- Easy to get through the Internet.
- PHP work in the web server to communicate between the client server and database.
- Easy to use because its syntax is the combination between C, JAVA and Perl.
- PHP that is compiled as Apache module provide a few extra utilities such as user authenticity and cookies, not like other CGI scripting language.

iii. OS – Linux (Has changed to Microsoft Windows XP)

Reason: -

- Alternative to world of commercial operating system (OS).
- PC exclusive.
- More friendly to other operating system.
- Developed for the use of networking and Internet and etc.

- For server and database implementation.
- More user-friendly and globally usage.

iv. Apache HTTP server (Has changed to Microsoft Internet Information System (IIS))

Reason:

- Web server software.
- Best performance and compatibility.
- More compatible running under Microsoft Windows XP.

v. MySQL

- Can operate well in Linux platform and Windows platform.
- Is not memory consumption software but effective like other Database Management System (DBMS), which are more popular.
- A security trusts software.
- Multi-user and multi-thread SQL server.

4.3.3 Hardware

Computer server, which contains minimum: -

a. Intel Pentium II

b. 64MB RAM

c. 200MB of memory space (Hard disk space)

d. 256 svga color screen monitor.

e. Other peripherals (keyboard, mouse etc.)

f. Modem

4.4 System Design

During the phase, the system architecture is created, which include the entire component inside of the system. It is also involve overall explanation about system development and predicted the successfulness of the system. In this phase, all the characteristic of the system such as the system architecture, database design, data flow process and the flow charts.

4.5 Coding

The next phase is coding and system implementation. It is very complex and the most important between all the process in the Software Development Life Cycle (SDLC). It is the component that decides whether system really succeeds and fulfilled its objectives.

A lot of study and analysis has been done carefully to seek the right and perfect development environment for the system. Nevertheless, there are three aspects, which are carefully measured. They are: -

- Control structure
- The algorithm
- Data structure

4.6 Testing and Maintenance

When the system has been finished coded, it would go into this phase where it is tested to see its effectiveness of the program or system execution. It is important as to detect the errors or unpredicted limitations of the system and to track back the problem of the coding. With this process, the modules are free from any problem and the system could operate nicely and confidently.

5.1 System Design

The Mastermind game system can be divided into four modules: -

a. User Module

b. Interface Module

c. Game Module

d. Engine Module

Figure 5.1 below show the interaction between the four main modules.

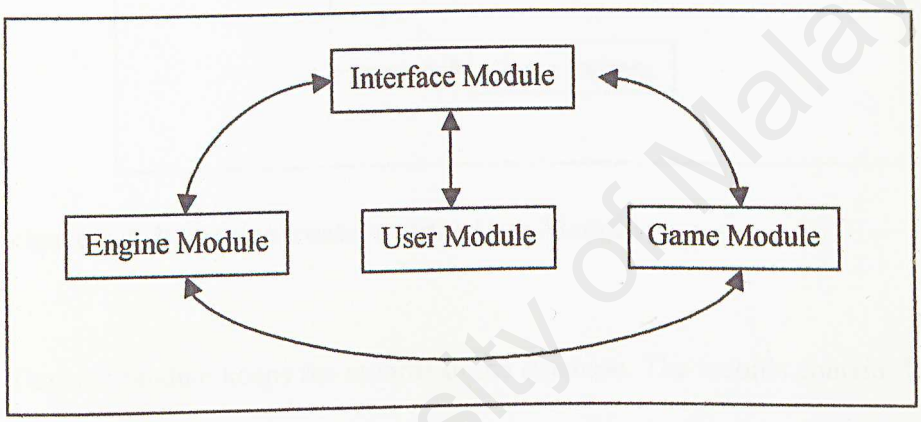

Figure 5.1: Interactive between the modules

Though every module of the game system has its own level of operation and execution but their process is generated by the reaction between the user and the system itself. Every part of the system module is connected because the component work simultaneously generated by the input from the user.

5.1.1 User Module

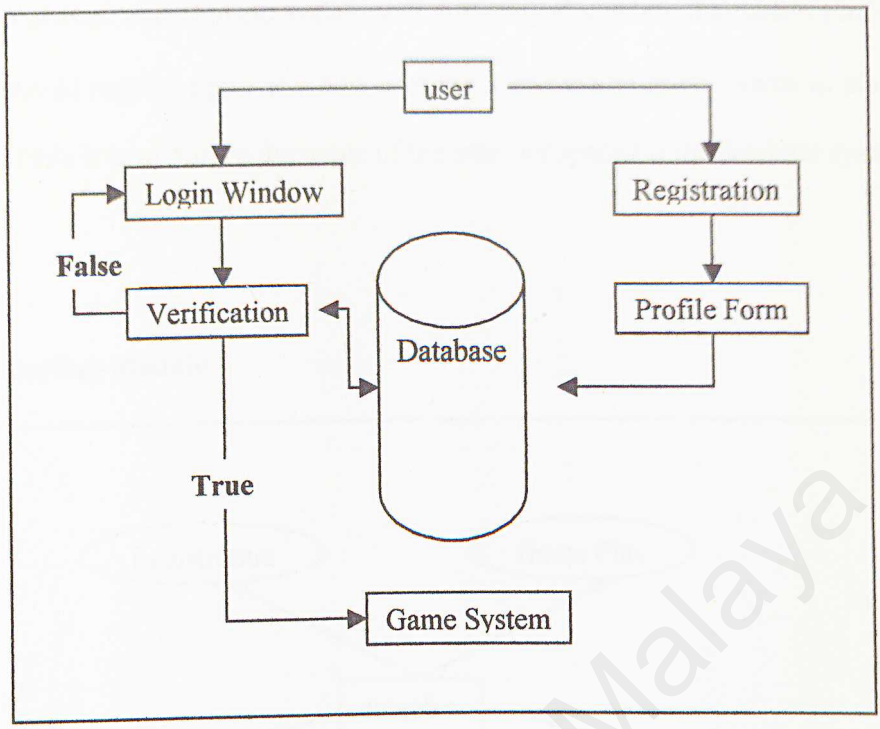

Figure 5.2: Diagram structure of the User Module

The user module keeps the records in the database. The records contain the time, date, opponents' name, scores, ranking and user profile. Before the user can play the game, the user has to login first. If the user is new to the system, there will be no record of the user in the database and of course the user should have to register as a new user. The login window is a gate to enter the game. After successfully verifying the login name and password provided by the user and is true, and then the user will allow continuing with the game. Otherwise, the user has to re-input his or her login name and password.

When playing the game, any moves made by the player will be recorded in the database. So, the player can see the steps that he or she made for each game that they played. The

module also keeps the timeline of the user. If the user doesn't login to play in a period of time, the database management system will automatically delete that user's profile and the user should register again as a new user the next time he or she wants to play. The purpose of this is to minimize the usage of the memory space for the database system.

5.1.2 Interface Module

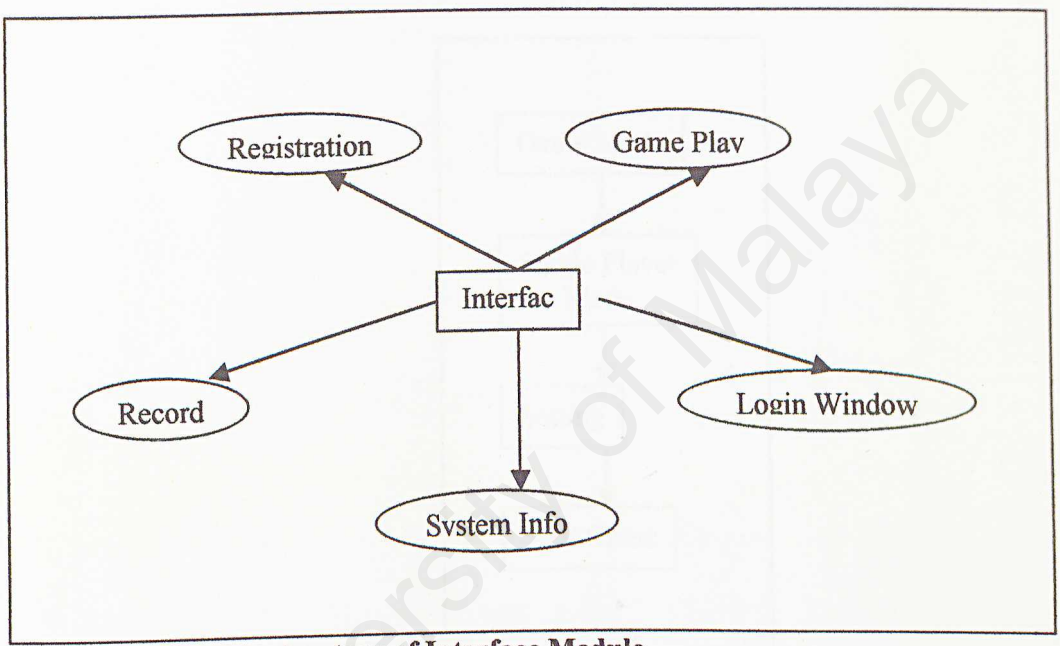

Figure 5.3: Diagram structure of Interface Module

This module is a medium for communication and to change the information between user and the system. It is a two ways interaction. All the output from the internal process made by the system is displayed on the screen. In order to make this easy to understand, the interface has to be user-friendly and some components have to change on real time action.

5.1.3 Game Module

The game module is where the game is divided into modes.

Single Player Mode

The single player mode is where the user plays with the system, which is using the genetic algorithm technique. The user will be the code breaker at the first round and will take turn to be a code maker at the other round. The system will show the hint on how the system solves the problem by providing some text in a text box.

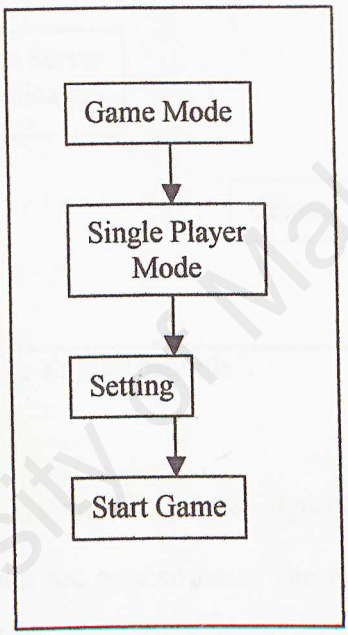

Figure 5.4: Diagram structure of the Game Module

5.1.4 Engine Module

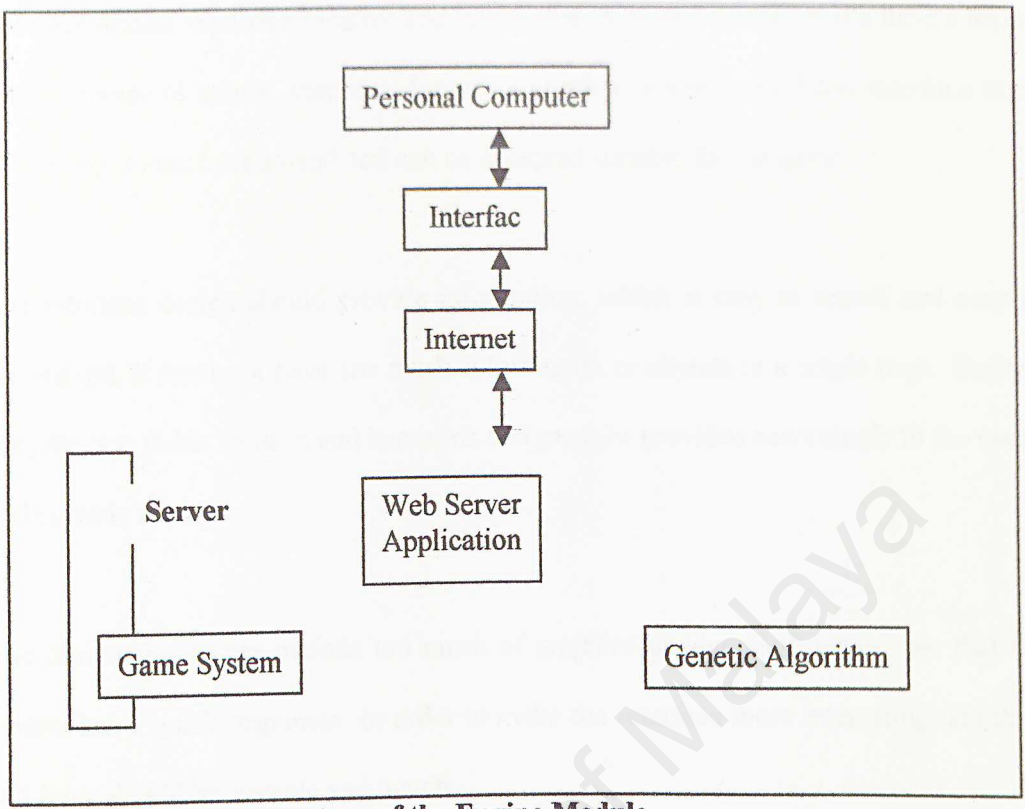

Figure 5.5: Diagram structure of the Engine Module

This is the main component of the system. The genetic algorithm concept as the brain and the game system as the operator are applied here. The module also contains the system for networking and Internet purposes. When the game starts, the game's engine also starts its operation. Here the algorithm will be the logical instrument to process, compare and decide the action.

5.2 Interface Design

Interface design requires a creative and innovative idea. The design should have a theme for the choice of colors, shapes of buttons and other components. Each interface must have a nice to see background and can be accepted suitable for the game.

The interface design should provide information, which is easy to search and easy to understand. It must not have too much information or objects in a single page. Besides that, the text fields, buttons and hyperlink text must be provided accordingly to the usage and suitable needs.

The design should not include too much of graphics or objects to make sure that the system has a quick responses. In order to make the interface more interesting, graphics and fonts should be suitable and match.

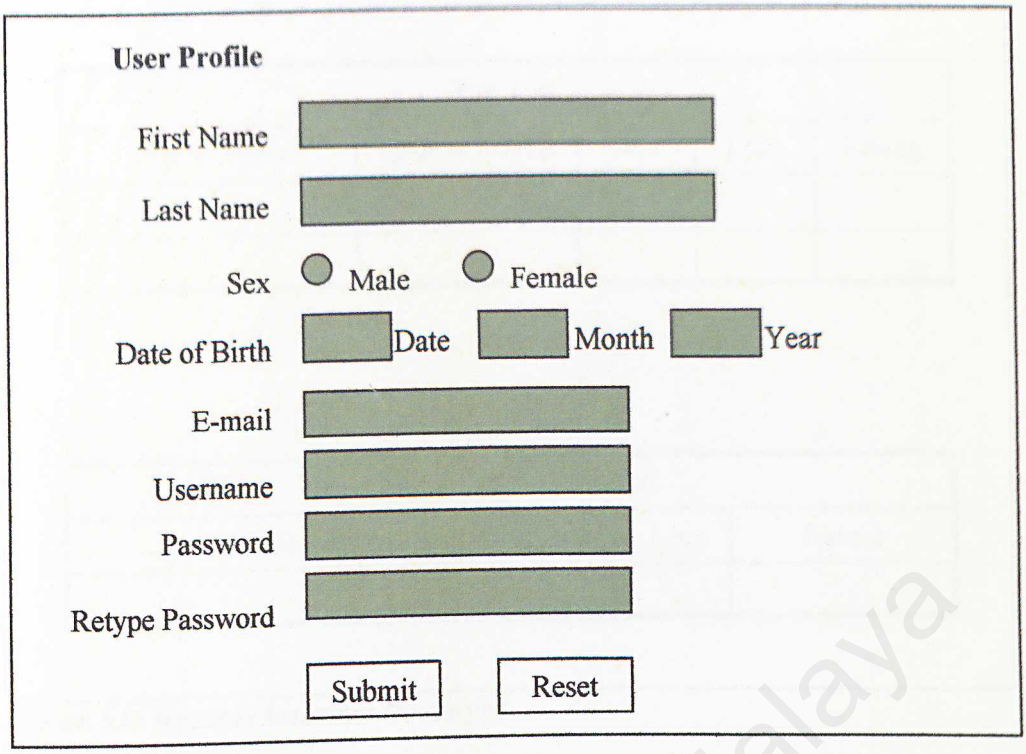

Figure 5.6: Registration Interface Prototype

Figure 5.7: Login Window Interface Prototype

Top Ten				
Player Name	Rounds Played	Win	Lose	Rating

Rating of current user			
Rounds Played	Win	Lose	Rating

Figure 5.8: Records Interface Prototype

User Profile

Name

Sex

Date of Birth

E-mail

Login Name

Back

Figure 5.9: User Profile Interface Prototype

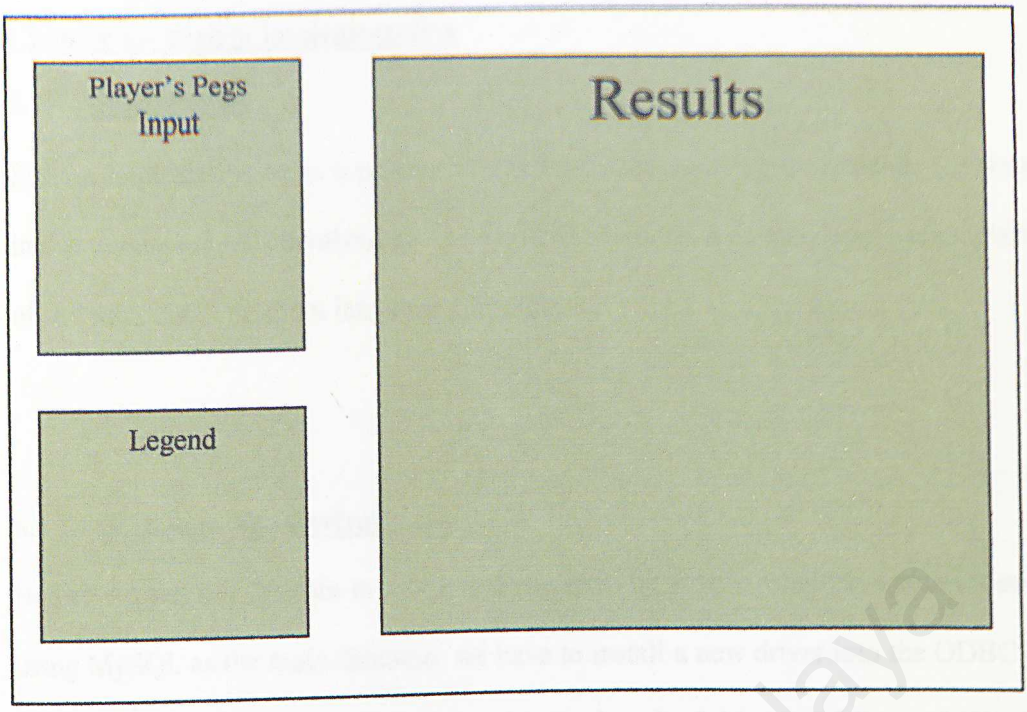

Figure 5.10: Game Play Interface Prototype

6.1 Introduction

System implementation is a process that converts the system requirements and designs into program codes. It involved coding step that translates a details design representation of software into a program language realization.

6.2 Setting up the Environment

Before we can use the data in a database, we must be able to establish a connection. As using MySQL as the main database, we have to install a new driver into the ODBC data source because this driver does not found in the bunch of drivers in Microsoft Windows XP. To download this driver, simply go to www.mysql.com and search for driver name; MySQL ODBC 3.51. After installation of this driver, Microsoft Windows XP is ready to run the MySQL database.

6.2.1 Installation MySQL on Windows

The MySQL server for Windows is available in two distribution types:

1. The binary distribution contains a setup program which installs everything we need so that we can start the server immediately.
2. The source distribution contains all the code and support files for building the executables using the VC++ 6.0 compiler.

Generally speaking, we should use the binary distribution.

6.2.1.1 Installing the Binaries

1. If the system is working on an NT/2000/XP server, logon as a user with administrator privileges.
2. If upgrading of an earlier MySQL installation, it is necessary to stop the server. If running the server as a service, use:

   C:\> NET STOP MySQL

   Otherwise, use:

   C:\mysql\bin> mysqladmin -u root shutdown

3. On NT/2000/XP machines, if want to change the server executable (e.g., -max or -nt), it is also necessary to remove the service:

   C:\mysql\bin> mysqld-max-nt --remove

4. Unzip the distribution file to a temporary directory.
5. Run the `setup.exe' file to begin the installation process. If want to install into another directory than the default `c:\mysql', use the Browse button to specify your preferred directory.
6. Finish the install process.

6.2.2 Create and Use a Database

1. Click on the Start > Run
2. Type 'command' in the dialog box and press Enter
3. When the Windows DOS prompt, go to the folder where the mysql.exe is existed (e.g. C:\mysql\bin)
4. Run mysql.exe
5. To create a database, type

   mysql> CREATE DATABASE mastermind; //where mastermind is the name that is given to the database

6. To use an existing database, type

   mysql> USE mastermind;

6.2.2.1 Connection to the Database

An ADO Connection object represents a unique session with a data source. One of its key properties, connection string is used to specify the Data Source and other information necessary for establish a connection to the database and authentication. (*see example*).

```
<?php
$link = mysql_connect("localhost", "root", "")
    or die("Could not connect: " . mysql_error());
mysql_select_db("mastermind",$link);
?>
```

6.2.3 Installation of Hypertext Preprocessor (PHP) on Windows

There are two main ways to install PHP for Windows: either manually or by using the InstallShield installer.

If there is Microsoft Visual Studio in the Windows environment, we can also build PHP from the original source code.

Once we have PHP installed on our Windows system, we may also want to load various extensions for added functionality.

6.2.3.1 Windows InstallShield

1. The Windows PHP installer available from the downloads page at http://www.php.net/downloads.php, this installs the CGI versionof PHP and, for IIS, PWS, and Xitami, configures the web server as well.
2. Notes that while the InstallShield installer is an easy way to make PHP work, it is restricted in many aspects, as automatic setup of extensions for example is not supported. The whole set of supported extensions is only available by downloading the zip binary distribution.
3. Install selected HTTP server on the system and make sure that it works.
4. Run the executable installer and follow the instructions provided by the installation wizard. Two types of installation are supported - standard, which provides sensible defaults for all the settings it can, and advanced, which asks questions as it goes along.

5. The installation wizard gathers enough information to set up the php.ini file and configure the web server to use PHP. For IIS on Windows XP, a list of all the nodes on the server with script map settings is displayed, and we can choose those nodes to which we wish to add the PHP script mappings.
6. Once the installation has completed the installer will inform to restart the system, restart the server, or just start using PHP.

6.2.3.2 Manual Installation Steps

1. Extract the distribution file to a directory. c:\php\ is a good start. It is recommended to use a path in which spaces are not included (for example: c:\program files\php is not a good idea). Some web servers will crash if we do.
2. We need to ensure that the DLLs which PHP uses can be found. The precise DLLs involved depend on which web server we use and whether we want to run PHP as a CGI or as a server module. php4ts.dll is always used. To make sure that the DLLs can be found, we can either copy them to the system directory (e.g. winnt/system32 or windows/system) or we can make sure that they live in the same directory as the main PHP executable or DLL the web server will use (e.g. php.exe, php4apache.dll).
3. The PHP binary, the SAPI modules, and some extensions rely on external DLLs for execution. Make sure that these DLLs in the distribution exist in a directory that is in the Windows PATH. The best bet to do it is to copy the files below into the system directory, which is typically:

c:\windows\system32 for Windows XP

4. The files to copy are:
    a. php4ts.dll, if it already exists there, overwrite it
5. The files in the distribution's 'dlls' directory. If the file already installed on the system, overwrite them only if something doesn't work correctly (Before overwriting them, it is a good idea to make a backup of them, or move them to another folder - just in case something goes wrong).
6. Copy php.ini file to '%SYSTEMROOT%' directory and rename it to php.ini. Your '%WINDOWS%' or '%SYSTEMROOT%' directory is typically:
7. There are two ini files distributed in the zip file, php.ini-dist and php.ini-optimized. It is advice to use php.ini-optimized, because it is optimized the default settings in this file for performance, and security. The best is to study all the ini settings and set every element manually. If we would like to achieve the best security, then this is the way, although PHP works fine with these default ini files.
8. Edit new php.ini file:
    - Change the 'extension_dir' setting to point to the php-install-dir, or where we have placed the php_*.dll files. Please do not forget the last backslash. ex: c:\php\extensions\
    - Choose the extensions we would like to load when PHP starts. Note that on a new installation it is advisable to first get PHP working and tested without any extensions before enabling them in php.ini.

- On IIS, set the browscap.ini to point to: c:\windows\system32\inetsrv\browscap.ini on XP.
- Note that the mibs directory supplied with the Windows distribution contains support files for SNMP. This directory should be moved to DRIVE:\usr\mibs (DRIVE being the drive where PHP is installed.)
- If we're using NTFS on Windows NT, 2000 or XP, make sure that the user running the webserver has read permissions to the php.ini (e.g. make it readable by Everyone)

6.2.4 Create Hypertext Preprocessor (PHP)

By default, Javascript is the chose language for scripting PHP. But it is still perfectly legal to mix languages, as long as they are properly specified for each section of code in the web page.

There are several unique ways to indicate that blocks of script are to be executed on the server (that is the server-side code). One method is to surround the scripts with the <?php and ?> tags. In this case, any text between these tags is treated as server-side scripting commands, based on the language defined at the beginning of the PHP.

For example:

```
<HTML>
<HEAD>
<BODY>
<?php setcookie("display_error",$message);?>          // PHP scripts
```

Message has been set as cookie.

</BODY>

</HTML>

In addition to specifying scripts to execute on the server, one can clearly indicate variables that are to be replaced by actual values by using the <?=$variable?> syntax.

For example:

// Name is variable

// It contains the user's name retrieved from a database

<P> Welcome <? = Name ?>! </P>

// Date() is a function which returns the system date

<P> Today date is <? = Date( ) ?><P>

6.3 Summary

This chapter explained several program design issues such as programming standards and procedures. Proper and judicious use of these will improve the quality of the software being developed.

Chapter 7 – System Testing

7.1 Introduction

Testing is a verification and validation process. Verification refers to the set of activities that ensure that the software implements a specific function. Validation refers to a different set of activities that ensure the software which been built is traceable to customer requirements. Software testing is a critical element of software quality assurance and represents the ultimate review of requirements specification, design and coding. The objective of testing can be stated as following:

- Testing is a process of executing a program with the explicit intention of finding errors, and making the program fail.
- A good test case is one that has a high probability of finding an as yet undiscovered error.
- A successful test is one that uncovers an as yet undiscovered error.

7.2 Testing Techniques

The following types of testing techniques are applied to this system:

1. Ad Hoc Testing

   Ad Hoc or ad lib testing means that the author simply test the functioning unit, trying whatever comes to mind, in attempt to make it fail. One shortcoming of Ad hoc testing is that while the author usually fined many errors, the author can never be sure what was or was not tested. Nevertheless Ad hoc testing was a fast and efficient way of debugging code errors during the early development stage.

2. White Box Testing

White box testing basically focuses on the idea of coverage. Coverage is as measure of how much of a module or system has been exercised/executed by a test case or a series of test cases. There are four main level of coverage:

- Branch/Decision coverage – every branch/decision should be tested at least once.
- Segment/Statement coverage – every segment/statement should be covered at least once.
- Path coverage – every path should be tested at least once.
- Loop coverage – every loops should be tested for skip, one pass through the loop, more than one pass through the loop.

3. Black Box Testing

Black box testing focused on functionality of the code. The main objectives is to uncover those wrong functions programmed correctly, by feeding input to the black box and take notes on what output is produced. During black box testing, the author used equivalence class partitioning. In equivalence class partitioning the author runs one test for each class of input to the module and then run additional tests using invalid data to make sure the error routines are working correctly. This test was done on the system user input forms. The author also did a boundary value analysis on those user input forms, since many errors tend to occur on the boundaries of equivalence classes. The test included test scenarios where the value sets is inside, on the outside the boundary.

Below listed the summary of units that were independently unit-tested:

a. Opening and closing of connection to the database
b. Inserting of new data into database
c. Modification of existing data in the database
d. Retrieving data from database
e. Validation of user input data before submission
f. Validation of user identity
g. Execution of SQL statements
h. Returned query results

7.2.1 Unit Testing

Unit testing focuses on the smallest software component (module). The modules are tested independently of one another to ensure their correct operation. Unit testing is performed from bottom-up-starts with the smallest and the lowest level modules and proceeds one at a time, and from top-down-begins with the upper level modules. For each module in bottom-up testing, a short program called a driver executes the module and provides the test case data, so that the module performs the way it will when embedded within a larger system. In top down testing, stubs or dummy subprograms are used to replace modules that are subordinate the module being tested. A stub uses the subordinate module's interface, may perform minimal data manipulation, prints verification of entry and returns control to the module being tested.

Unit testing involved:

- Testing the interface to ensure that information properly flows into and out of the program unit.
- Testing the boundary condition to ensure that the module operates correctly at the boundary values.
- Testing all the error handling paths.

7.2.2 Integration Testing

After the unit test, the modules are integrated into a working system. For this system, incremental approach was used. In the incremental approach, the units are added one by one to the set of integrated units.

During integration testing, two or more units in which either unit that use output data from or provide input data for another unit were tested in collection. These units have related characteristics to perform a common goal or function such as the search engine function which comprised of SQL statement generating, query form submission, and displaying query results.

7.2.3 System Testing

System testing is a series of different tests whose primary purpose is to fully exercise the computer-based system. System testing verifies that the software is properly integrated with all other system elements and that it performs its allocated function. Following are the system testing consideration:

- *The Event List*

All the possible triggers are exercised and the expected results compared with the actual results. Every function is tested by one or more events in the event lists.

- *Error Message Testing*

Every error message, which can be generated by the system is extracted from the code and placed in a table to test for appropriateness and understandability. Error message is also checked from a "national language" point of view to see that they have enough room to be translated both in the table and on the screen where they are displayed.

- *Documentation Testing*

All examples used in the user's manual is tested for correctness and for whether or not the manual gives the exact answers users will obtain when they run the examples.

7.2.4 Fundamental Testing

In addition to the functional tests based on functional behavior as defined above, there are other tests fundamental to all software. Certain of these are difficult to measure accurately. Five of these fundamental tests are:

- Usability – the usability should be based in building user interfaces that have patterns already familiar to the typical user. The user then learns to use the

software through pattern matching and paradigm shifts, exactly as they do in mastering any product.

- Reliability – reliability tests are conducted, according to mathematical models of software reliability, to ensure that the system can be probability of some function of the system failing within a specified time. Reliability testing is monitoring the mean time between failures. Reliability and consistency testing go hand in hand where the system behavior (inputs, outputs, response time) is measured for consistency.
- Performance – Performances tests are conducted to ensure that the system response time meet user expectations and does not exceed the specified performance criteria under heavy stress or volume. During these tests, response time and the transaction rate are measured the purpose of performance test is to test-run the performance of various functions of the software within a specified hardware configuration. The performance tests can couple this test with stress testing.
- Serviceability – the ability of the support person manning the phone line to acquire enough information to enable the maintenance organization to assess the error and fix it without additional information if the user's software system crashes.

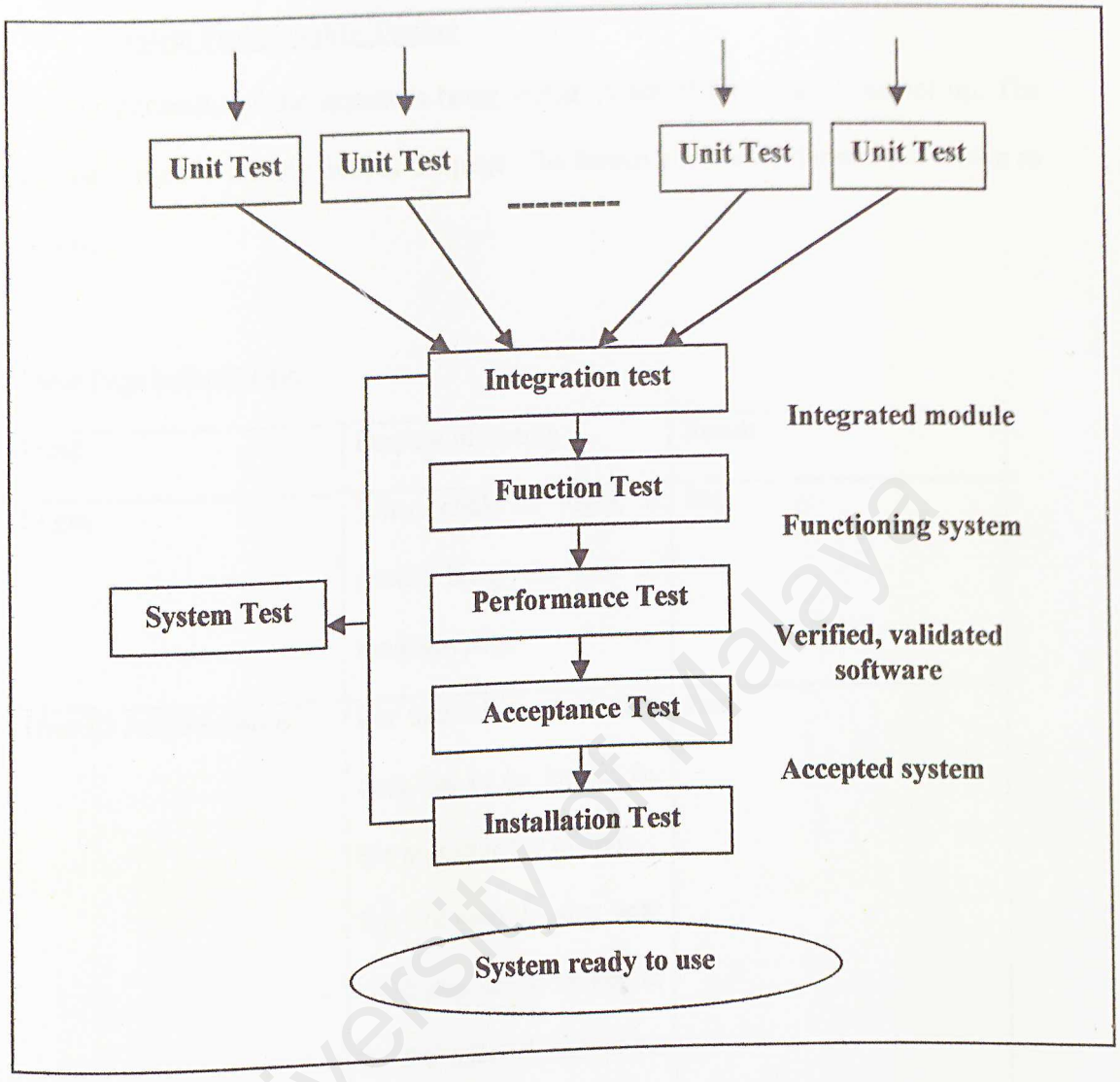

Figure 7.1: The Testing Steps

7.3 System Functionality Testing

The functionality of the system is being tested. A set of forms has been set up. The format is used to test the field in the page. The format and the field tested are shown as below:

Main Page before Login

Field	Purpose of testing	Result
Login	When click on login, it should bring the user to the login page	Pass
User ID And Password	For user id, the text field area has to be key in by the user in order to login. For password, the text field area has to be key in by the user. If has to be matched with the data in the database. It is using the user id and password key in by the user to do the matching.	Pass
Login Register	This area is for new user that is first time to use this	Pass

	system. From here, new user can enter the information like user id and password for login to enter this system.	
Home	This page is the main page for this system.	Pass

Table 7.1: Result of the testing field before login

Main Page after Login

Field	Purpose of testing	Result
Current List of Online Players	Shows the current online players and for current user to create new game	Pass
Current List of Online Game	Shows the current online game names that are played	Pass
Game Setting	*Let the user to set the number of rows and number of rounds of the game created*	Pass
Choosing Pegs	Let the player to choose their pegs of the pegs	Pass

	combination	
Markers Results	Shows the markers and results of the pegs combination that are chosen	Pass
Create Pegs Combination	Let the codemaker (user) to create his/her combination	Pass
Computer Code Generator	Generate the code by the computer system to solve the user's combination	Pass
Logout	This page let the current user to logout and delete all current information from the database.	Pass
History	Shows the Top Ten players of the game and the placing of the current user	Pass
User Profile	Shows the current user's profile	Pass

Table 7.2: Result of the testing field after login

7.4 Summary

Software testing is a critical element of the software quality assurance and represents the ultimate review of specification, design and coding. After doing this part, logic error have been covered and make sure the system confirm to be requirement specified.

Testing is carried out to discover different classes of errors in software development. It demonstrates that the software functions appear to be working according to the specification and performance requirements that have been met.

During the system testing the system start with unit tested. The next level of testing is integration testing, i.e., testing the integration of modules. Eventually, the whole system is tested.

The main testing techniques are:

- Ad Hoc
- Black box
- White box

Successful testing will result in quality software - software with fewer errors and which work according to specification and performance requirements. It will lead o dependable and reliable software.

Chapter 8 – System Evaluation

8.1 Obstacles Faced

As usual, problems always occur during system development and it has been no exception in this project. Throughout this project, many problems have kept unfolding one after another as development work progressed due to many reasons.

1. During Analysis Phase

Determining Scope of the system

Since there was no prior experience in developing a system, it was hard to determine to which extent to define the scope of the system so that it can be completed within the given time frame. However, this was overcome by analyzing and studying all of the capabilities that MySQL and PHP technology can do before determining the scope of the system.

2. During Design Phase

Time Constraint

During the design phase, there was not enough time to study and produce the best solution of design in first semester. This was due to inexperience and insufficient knowledge of designing a system. The best way to learn is to read as many approaches used in previous year students' thesis documentation.

3. During Implementation Phase

Problems on Installation

There were a lot of problems on installing and configuring Red Hat Linux, Apache Server, Windows XP, IIS, MySQL, PHP and other tools before starting coding. The needed software and tools were successfully installed only after a few times of reinstallation. Installing the operating system Linux give me a lot of trouble especially when it corrupted my hard disk partition and deleted all the files contain in the Windows partition. From the experience, it is essential to know the sequence of product installations. This is to ensure smooth execution without system errors. This vital information about installation sequence is not provided in a user-friendly way to guide user.

4. During Testing Phases

Unanticipated appearance of web pages in different web browsers

The appearance of web pages is different on Internet Explorer 5 and Netscape Navigator and Communicator during the testing phase, such as different positioning of graphics, text, and tables on these web browsers. The main cause of these problems couldn't be detected.

8.2 System Strength

Although this system does not have powerful features to some extent, but it still has some strength of its own when compared to some existing system.

- Database maintenance and utilities.

Users are able to do housekeeping for database maintenance. They can create, add, update and delete the association information. Besides that, they can also keep track on the records and view the reports. Further more, the user can backup and restore the database.

- User interface.

This system is featured with an attractive and easy to use graphical interface. Most of the operation can be done by click and point operations. There are only a few input actions required during the user's interaction. Thus, users will spend most of their time using the system without much input effort.

- Transparent.

System is transparent, as users do not need to know where the database resides, how the system is structured, etc. For example, users do not need to know how to retrieve from and insert records in to the database.

- Relatively fast response in document retrieval from server.

Each web page is designed to be lightweight. These pages loaded in a reasonable amount of time to ensure users need not wait too long to view the pages. Heavy graphics is avoided.

- Error messaging.

This is a reliable system as it caters for almost any possible errors encountered. System will generate appropriate feedback to user when an error occurs. For example, a password validation failure or user login ID failure is handling by the system and a user-friendly message will then be generate to inform the user that he or she input the wrong password or user login ID.

8.3 System Limitations

Despite some of the system strengths mentioned previously, there are limitations, which cannot be researched and developed due to time constraint and the lack of resource.

Those limitations are:

- Performance level although can be adjusted in IIS, yet it still depends on Internet connection speed in phone line.

8.4 Future Enhancement

This system provides you with a functional framework from which to evolve. Here are some future enhancement's suggestions.

- Support other browsers. The system requires Microsoft Internet Explorer 4.0 and above for execution. In future, it can be tuned to fulfill other browser requirements such as Netscape for execution. This is because Netscape has a sizeable share in the browser market besides IE.
- Program the system to teach the players to play the mastermind effectively.
- Let the players join a game in multiplayer mode.

Summary

The online Mastermind game is an innovative redevelopment of the original Mastermind game, which is invented since 1970-71. This game was simple but is also a very interesting game. It needs a very critical thinking, experience and careful observation with high concentration.

This game will put online so that people can play it anywhere around the world. The game should give the user an easy approach since it does not need to download or need memory storage allocation for the system. In the other words, this game is played virtually.

The technique used in this system is one of the popular problems solving technique, which is called as Genetic Algorithm. The technique used in the system for solving problems with major probabilities of solutions.

Generally the game is developed as to fulfill the objectives of the system development. This game is not only for the meant of entertainment but also to test the effectiveness of one of the problem solving technique which was found in Artificial Intelligence (AI) research field.

User Manual

Hardware Requirements

The recommended hardware configuration requirement for accessing the online Mastermind game is:

a. Pentium II 233 MHz and above or AMD K6-2 300 MHz (3D now) and above.

b. 64 RAM (Random Access Memory) and above.

c. 15 inch Color SVGA (Super Video Graphics Array) Monitor (with resolution set to 800X600).

d. Standard Keyboard and Mouse.

e. A 14.4 KBPS (Kilo Bits Per Second) modem or Internet connection.

f. 100 MB Free Hard disk space and above.

Software Requirement

The online Mastermind game requires the following software as its running platforms:

a. Microsoft Windows 98, 2000, NT or XP

b. Microsoft Internet Explorer 4.0 and above

c. MySQL

d. PHP

e. Internet Information Services 5.0 (IIS)

Home Page

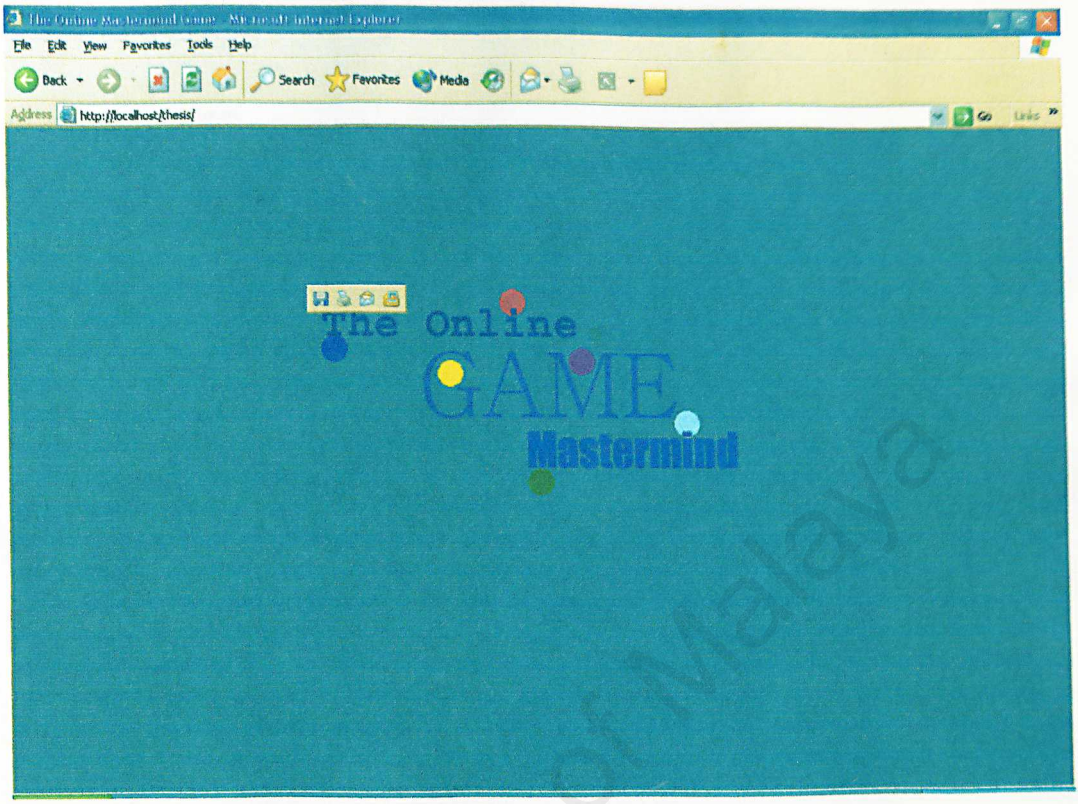

Figure 1: Main page

Page Descriptions:

This is a simple main page for the online Mastermind game. The color circles through out the page represent the pegs used in this game. This page will display for 2 second before it automatically link to the login page.

Login Page

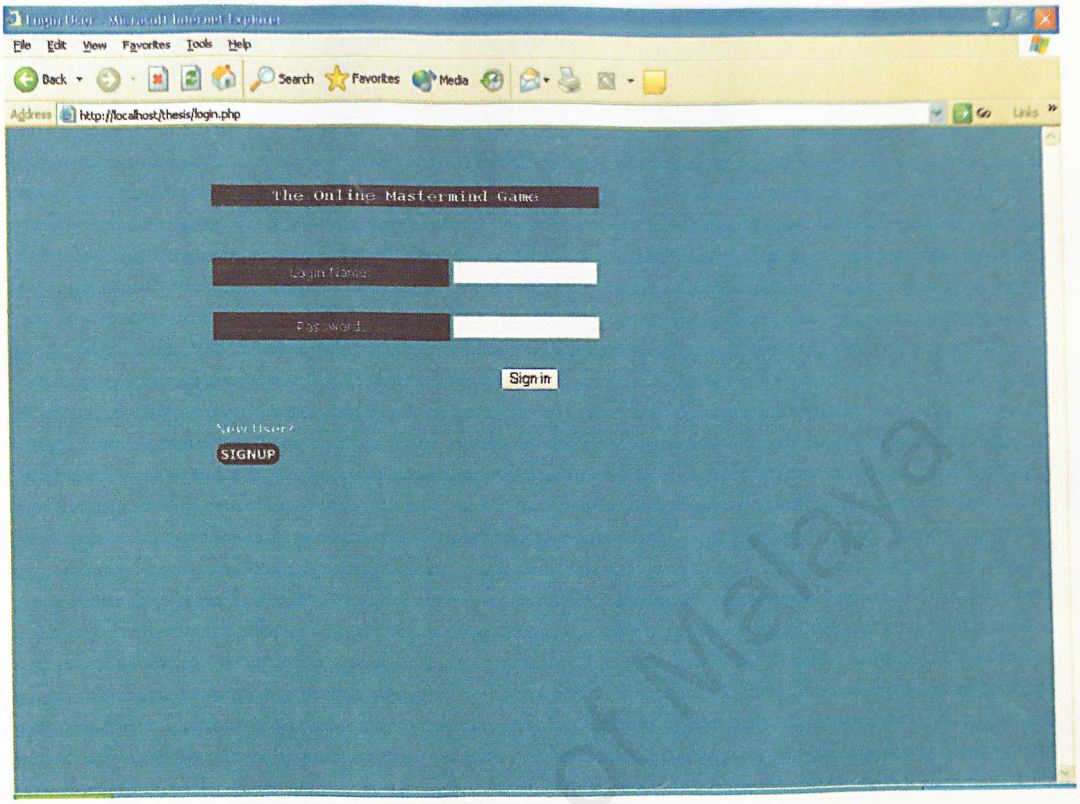

Figure 2: Login page

Page Descriptions:

1. Login Name – Field for users to key in their Login ID.
2. Password – Field for users to key in their Password.
3. Sign in – Users can click here for login to the system.
4. Signup – For new user to sign up as an authorize user of this game.

Login Fail

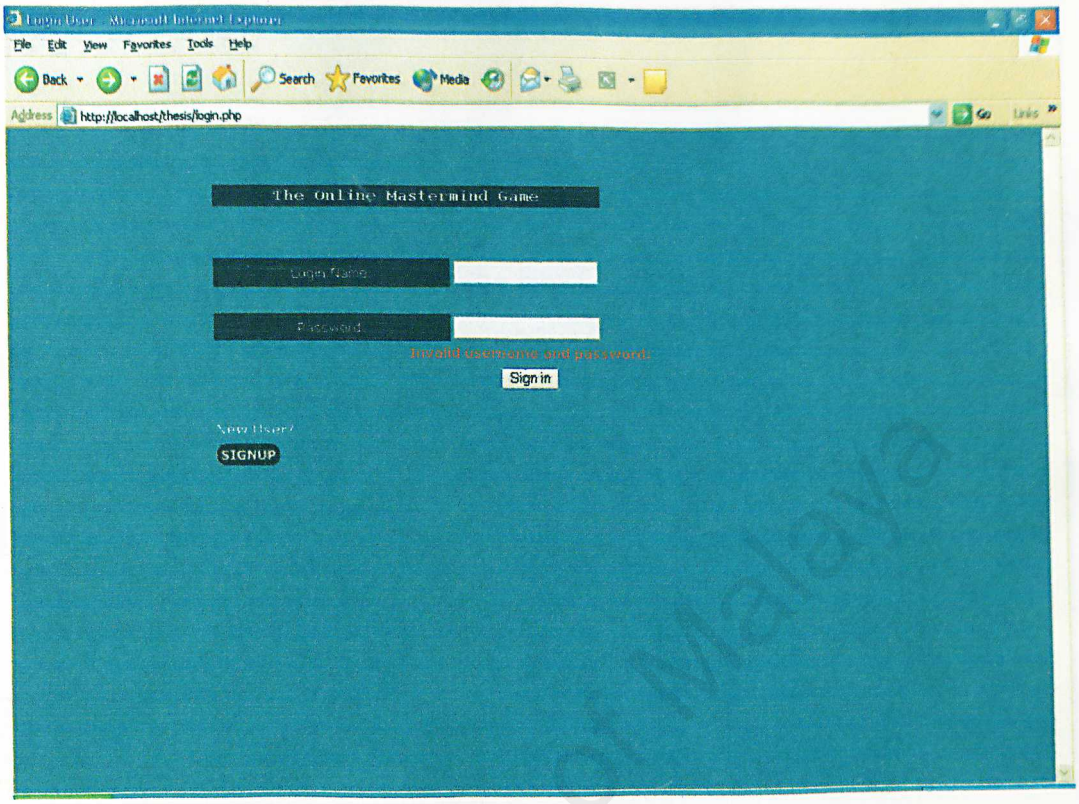

Figure 3: Login failed

Page Descriptions:

- If the user enters a username or password that is not found in the database, the system will ask him/her to re-enter the value.
- Otherwise, the system will encrypt the password and decrypt to check for hacker. If authorized, the system will go to the game page.

Registration page

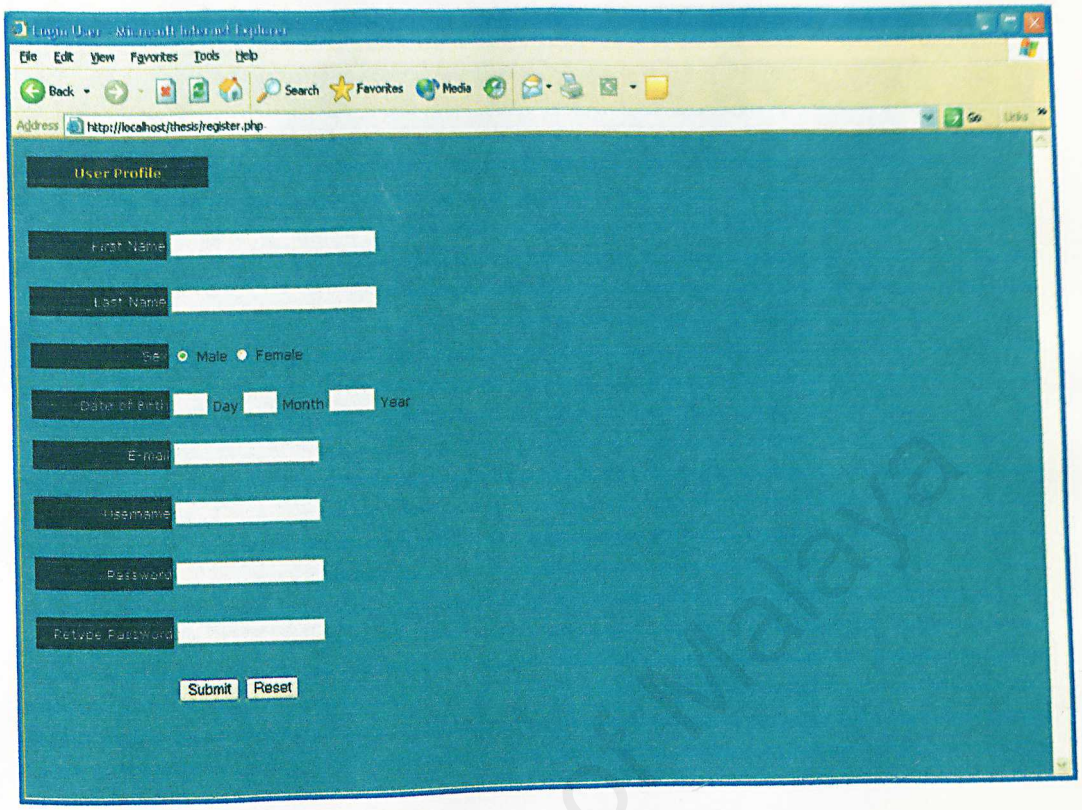

Figure 4: Registration page (before)

Page Descriptions:

User will fill in the text field provided. When the user fill up all the text field, the user has to click on the '**Submit**' button or press '**Enter**' to store into the database. Otherwise, the user can click on the '**Reset**' button to refresh the page.

When the user press the '**Submit**' button, the system will automatic go to another link. (see Figure 5).

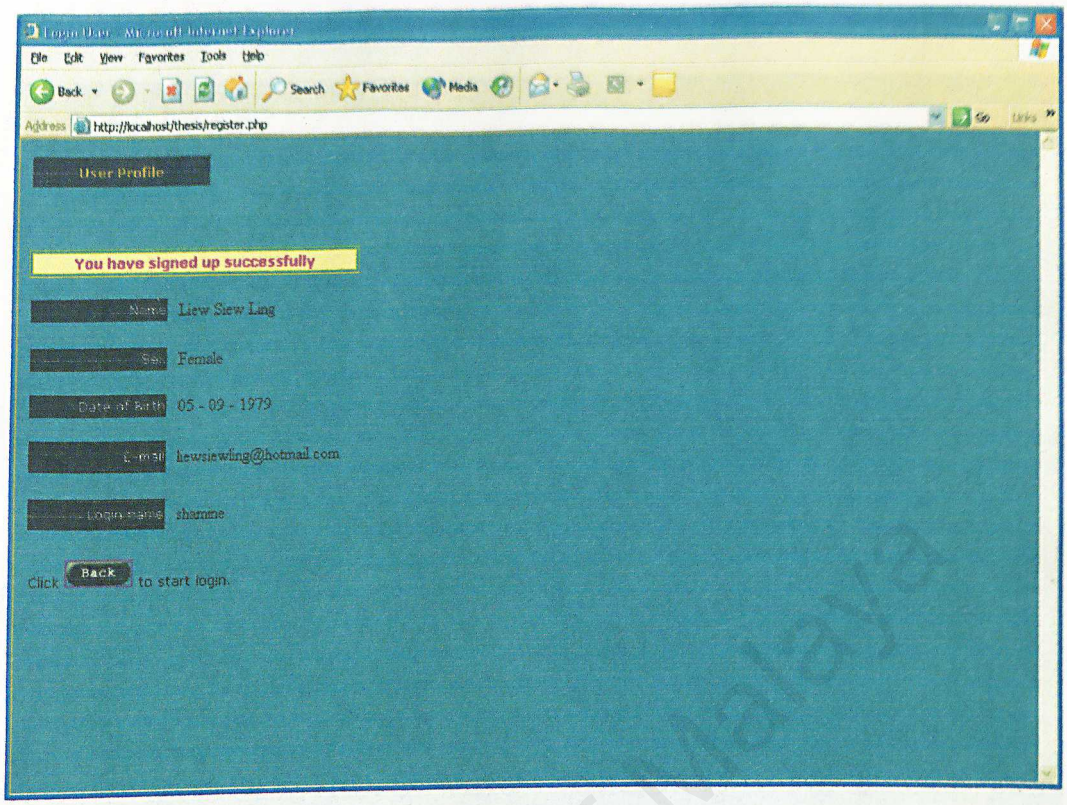

Figure 5: User registration (after press submit button)

Game page

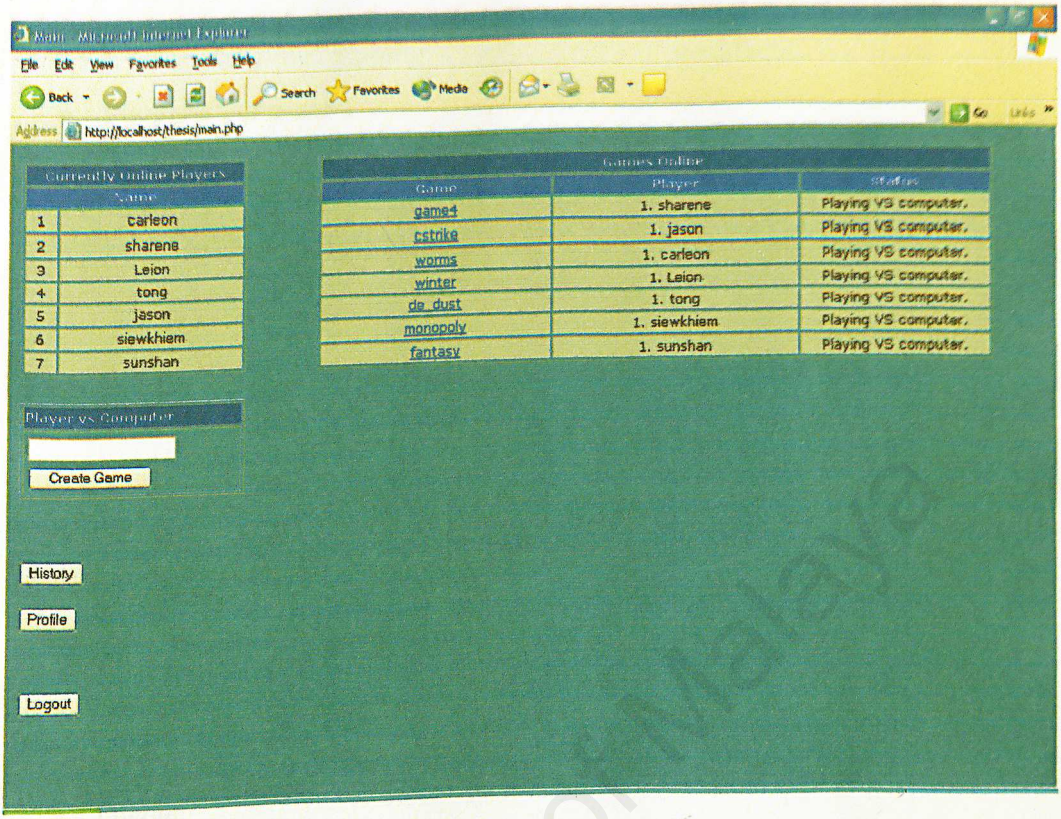

Figure 6: Game page

Page Descriptions

This page will automatically refresh for every 2 second to check for new registered game and online users. Besides, it contains some feature: -

- Currently Online Players
    - Check for current users that are login and play the game.
    - State out the list of online players in a table.

- Games Online
    - Check for current online games that are created by the players.

- State out the list of online game name in a table.

- Player vs Computer
  - To let the users to enter the game name and start a game playing with the system in single mode.

- History
  - Link to the rating page.

- Profile
  - Link to the current user's profile page.

- Logout
  - To let the current user to log out and go off from the current online player list.

User profile

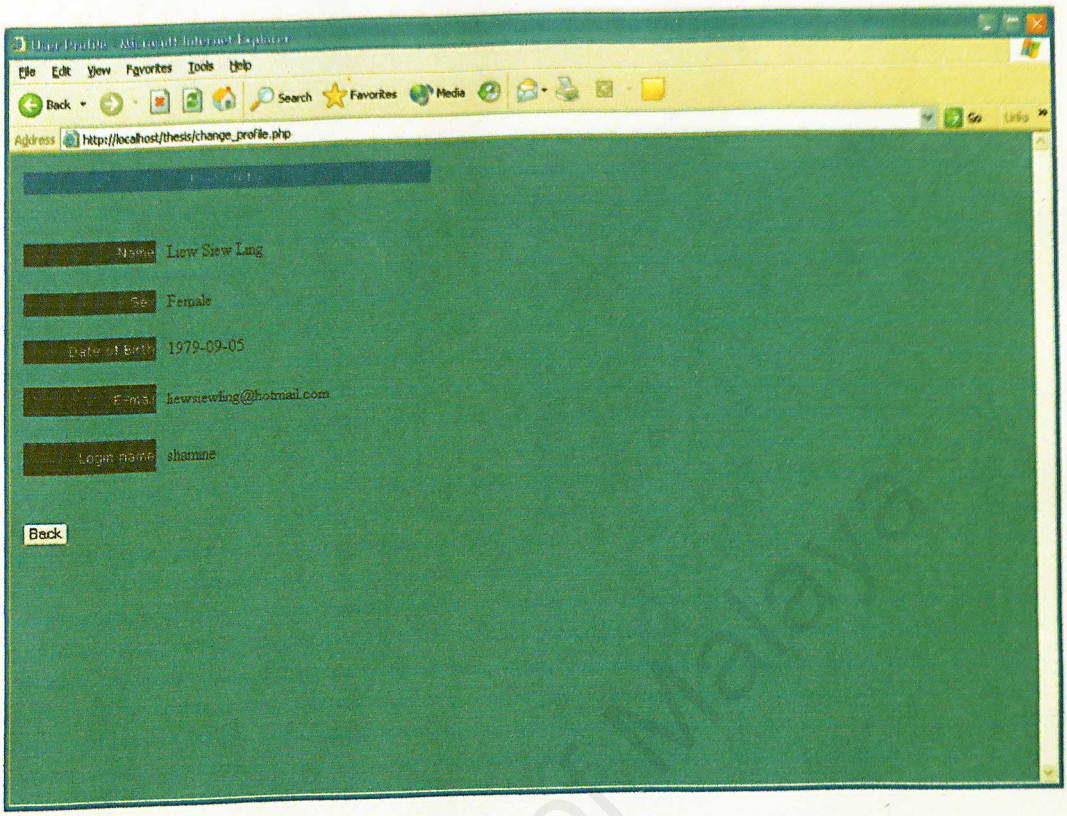

Figure 7: User profile

Page Descriptions:

When the user click on the '**User Profile**' button at the game page, it link to this page.

This page shows the details of the current user.

History page

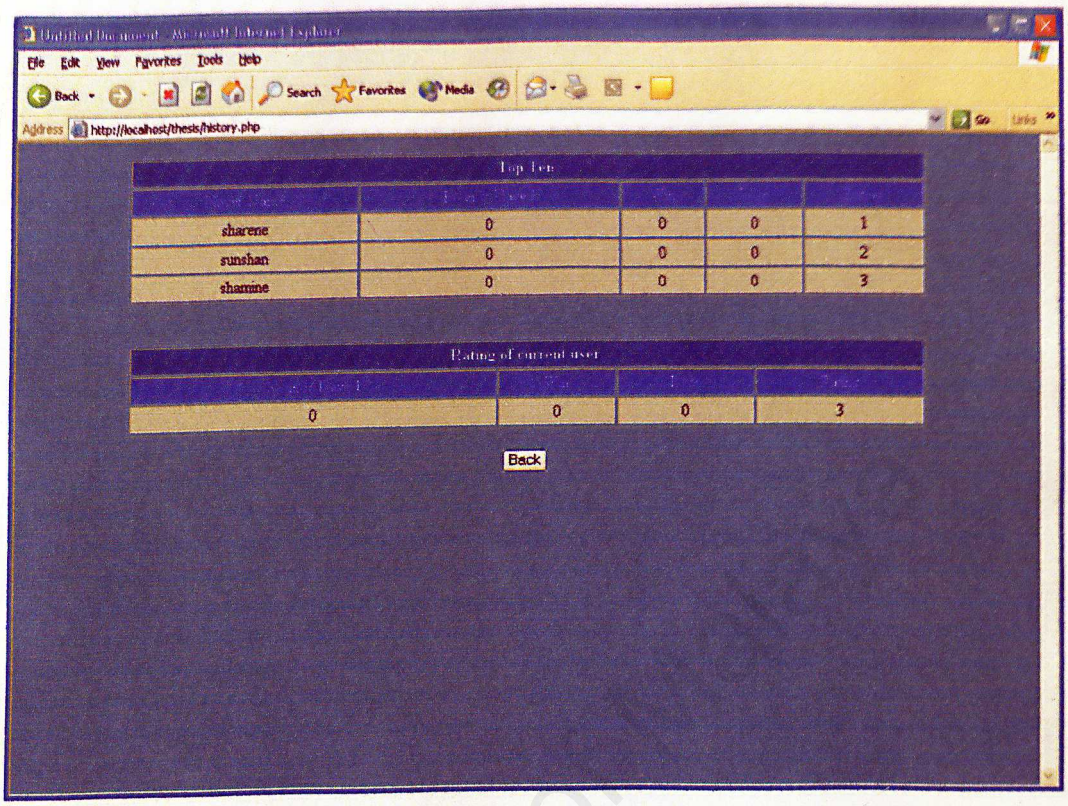

Figure 8: History or record of the players

Page Descriptions:

This page shows the rating of the players.

Upper table – Shows the top ten of the users.

Bottom table – shows the rating of the current user.

Game Setting

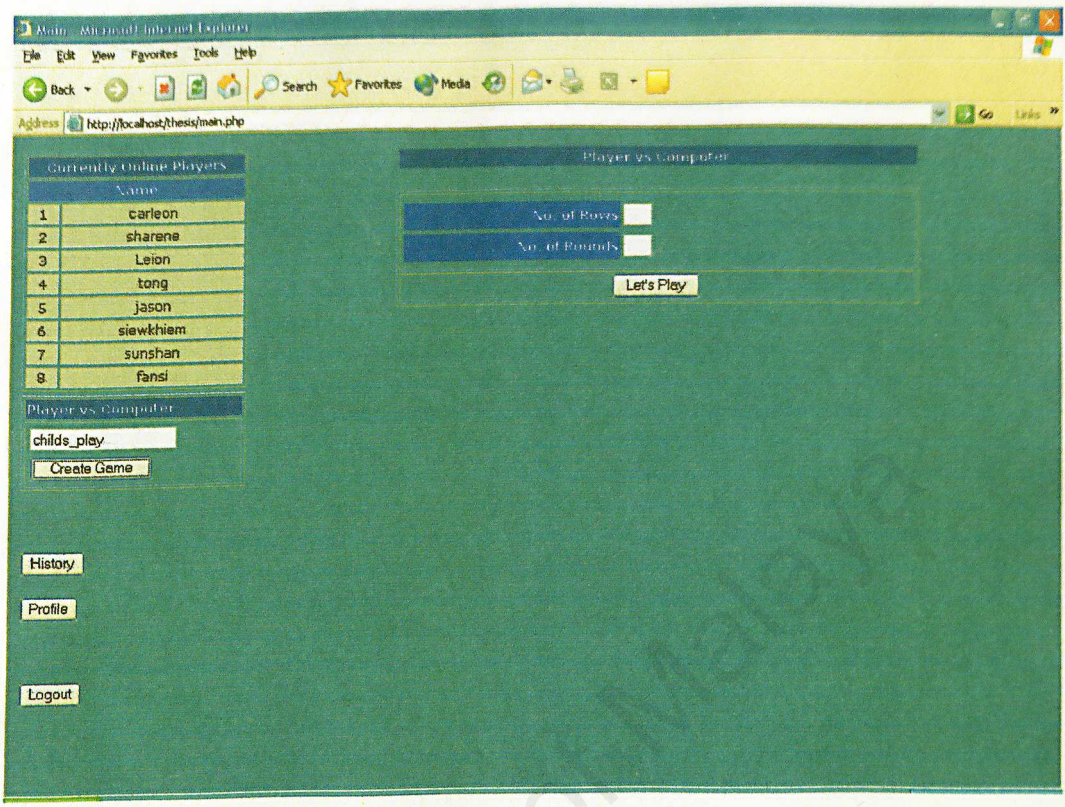

Figure 9: Game setting

Page Descriptions:

To let the user to set the number of rows and rounds of the game that is created.

- No. of Rows – set the number of rows in a round.
- No. of Rounds – set the number of rounds of the game.

When the '**Let's Play**' button is clicked, the game starts.

Play mode (Player as the code breaker)

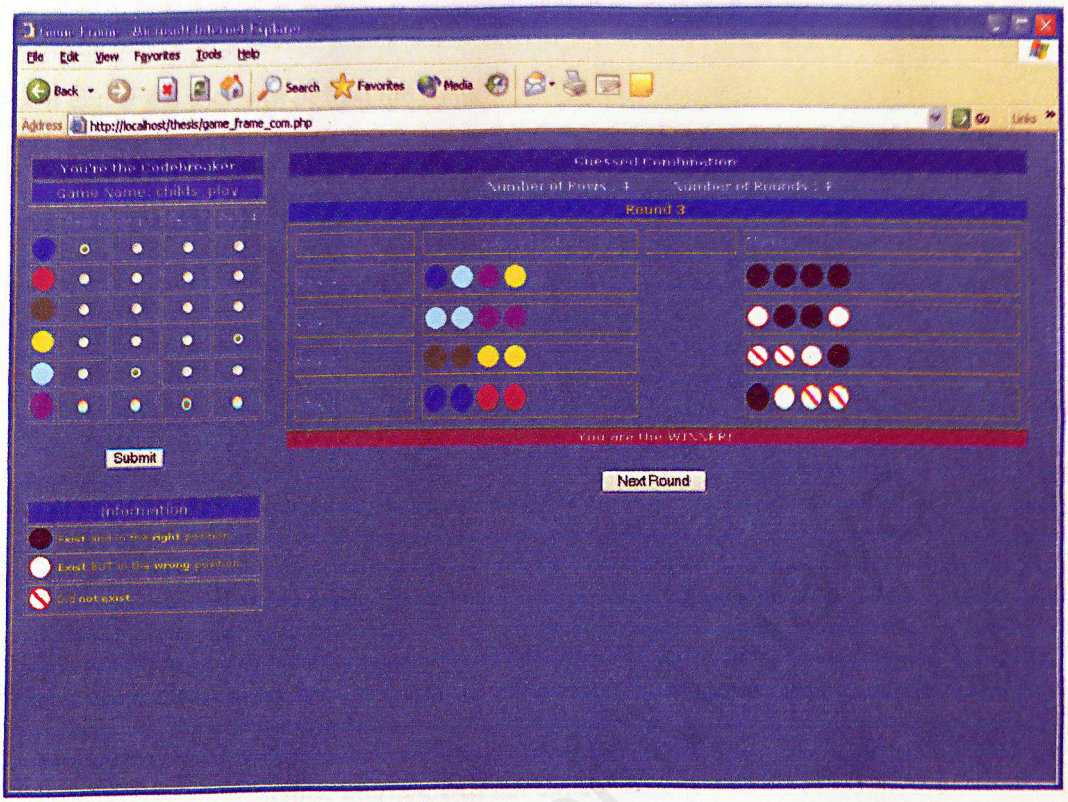

Figure 10: Play mode (user as the code breaker)

Page Descriptions:

In this page, the user (as code breaker) will input the code combination by using the radio button. When the user clicks on the '**Submit**' button or press the '**Enter**', the color combination will be created according to the radio input on the upper left table. The combination code that is created will be shown at the upper right table of the page and automatically, the system will place the markers for the user as reference.

The user will continue his/her game until he/she succeeded guess the code combination that is generated by the system or when the declared rows has been filled up. This round will end if the user win or lose the round.

After the user has finished this round as a code breaker, the use will continue to the next round as a code maker. The game will continue until the user has finish all the rounds that declared in the beginning of the game.

Upper left table – user input the code combination

Bottom left table – legend or information about the color markers.

Black marker: Exist and in the right position

White marker: Exist but in the wrong position

None marker: Did not exist

Upper right table – show the results of the code combination that the user created.

Play mode (Player as the code maker)

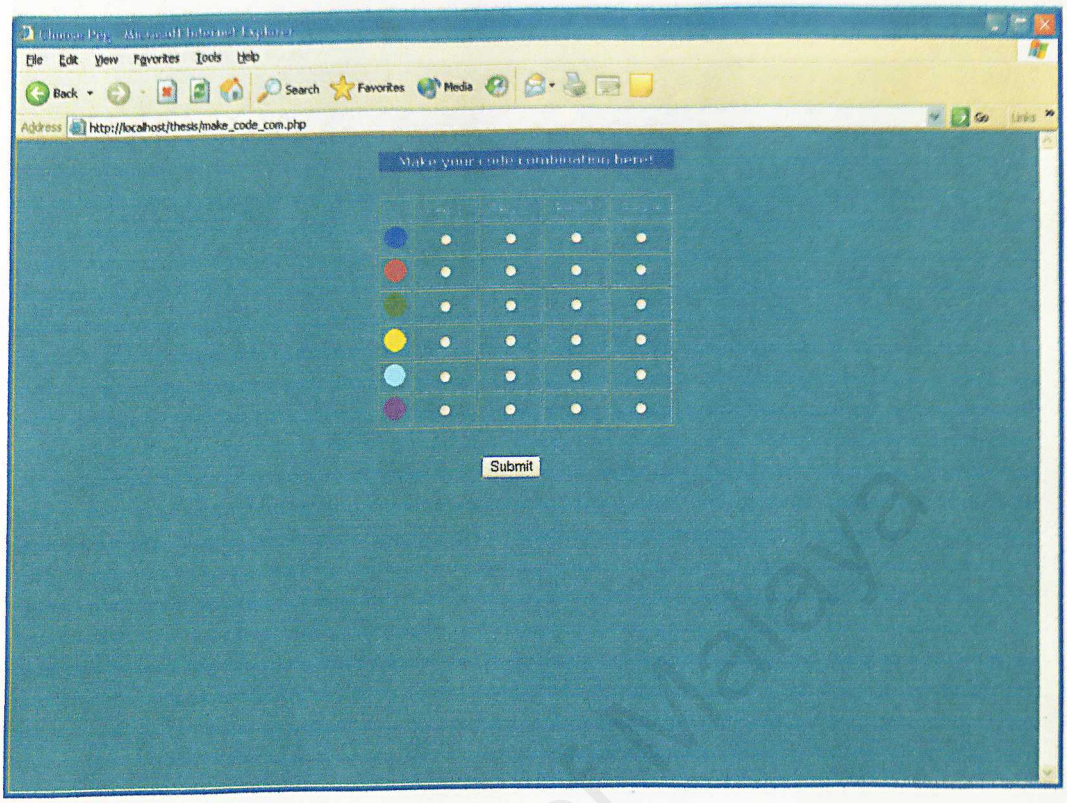

Figure 11: Play mode (user as the code maker)

Page Descriptions:

User will create his/her code combination from this table. The user will create the code combination by using the radio button. When the user click on the '**Submit**' button or press the '**Enter**', the code has been made.

The user can edit the code combination before he/she start the game. (see **Figure 12**). When the user click on the '**Edit**' button, the page will go back to the previous page to edit the code combination. If the user click on the '**Start**' button, the game is started.

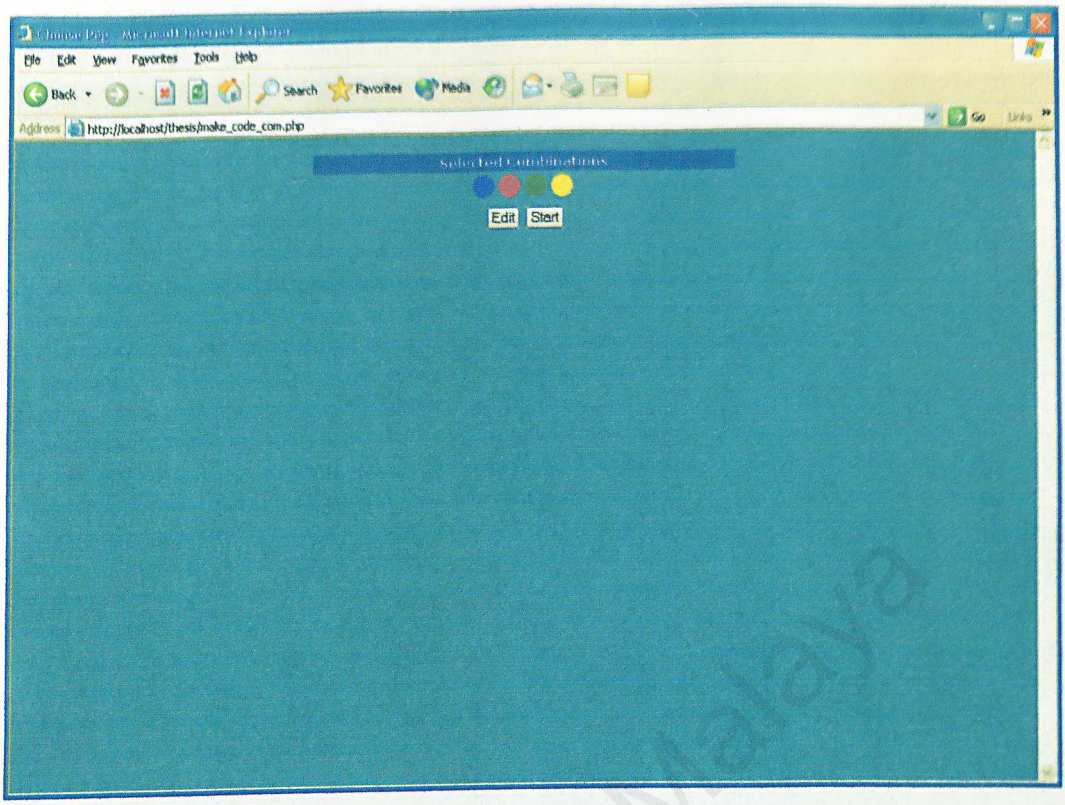

Figure 12: Play mode (confirmation of the code)

Play mode (Computer as the code breaker)

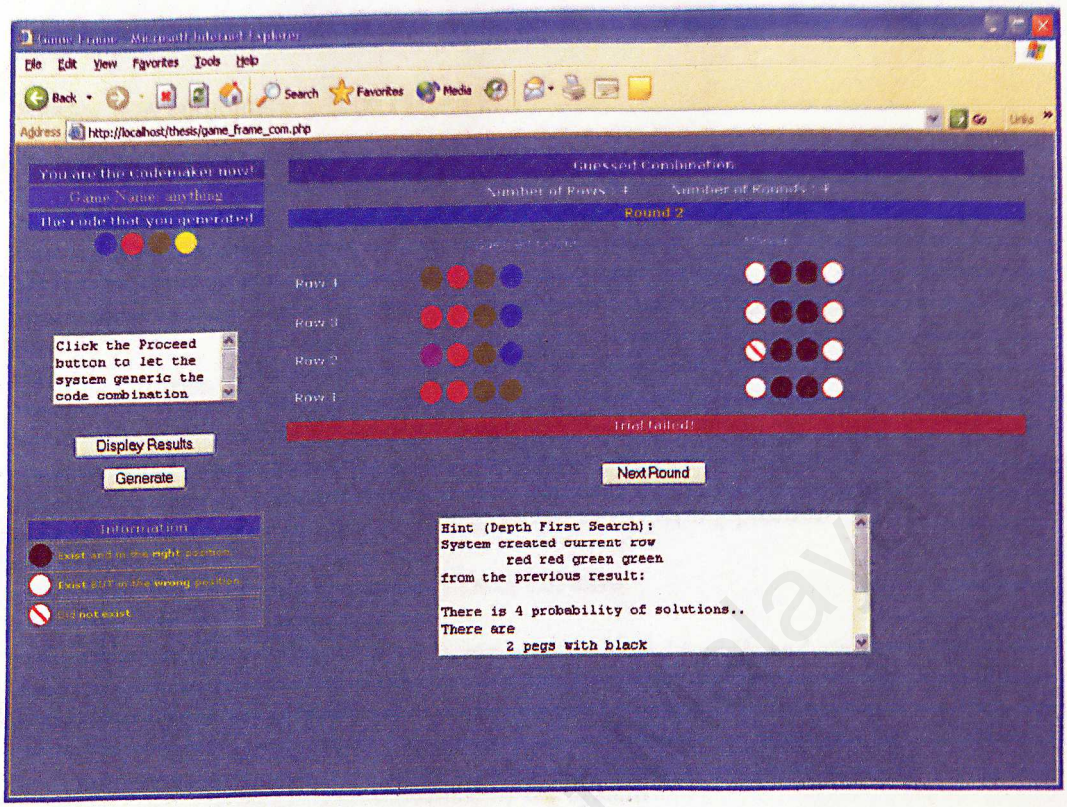

Figure 13: Play mode (computer as the code breaker)

Page Descriptions:

In this page, the user can see on how the system makes its combination by using one of the AI techniques, depth first search. As the first move of the system, it will randomly create a combination of two colors. (In this example, the system created a combination of 2 colors, red and green). From the research, the probability value after using two colors as the first move is at most 256 upon 1296 ways.

Each time the user click on the '**Display Results**' button, the system will show the results of the code that is created. Hints will be given at the bottom right of the page.

The user has to press the '**Generate**' button to give chance to the system to create another code combination. System will store the previous result as the result to build another new combination. (See **Figure 9**).

- Upper left – Shows the color combination that is created by the user
- Bottom left – Shows the legend or information about the markers used in this game.
    - Black marker – Exist and in the right position
    - White marker – Exist but in the wrong position
    - None marker – Does not exist
- Upper right – Shows the number of rows and rounds and also the result of the game.
- Bottom right – Hints showed by the system

Bibliography

1 Simon, H. A. 1981. The Sciences of the Artificial, 2nd Ed. Cambridge, MA: MIT Press

2 Kirk Patrick, S., Gelatt, Jr., C. D., and M. P. Vecchi. 1983, Optimization by simulated annealing. Science 220 (4598)

3 Bowden, B. V. 1953. Fater Than Thought. London: Pitman

4 Shannon, C. E. 1950. Programming a computer for playing chess. *Philosophical Magazine.* [series 7] 41: 256-257

5 Samuel, A. L. 1962. Some studies in machine learning using the game of checkers. In Computers And Thought, Ed. E. A. Geigenbaum and J. Feldman. New York: McGraw-Hill