



## Report on coursework – Apple Chess Game

### Contents:

1. The basic requirement of the programmer 
2. How does the program work?
3. The difficulties in writing the program 
4. The objects used in the program
5. Testing plan and debug
6. Log book
7. Improvement can be made later

#### 1. The basic requirement of the programmer

- The concept of array  
It is much more easier to write this program with the use of array as the chess representation in the program. As we have to detect where we can put the chess, it can be done easily with the help of array and coordinate in the program.
- The use of nested loop  
In the program, we may have to do the same thing but with different parameter for times, so it is faster by the mean of using nested loop. It is obvious that the use of array and nested loop is very powerful in a chess game.
- A well-developed brain  
In the program, there are many many variables in different usage. There are a number of variable representation different meaning and as a memory. The programmer must have a great memory to remember all the variables in the program.
- The sense of preventing errors in the program  
It is very east to make many errors in the program such as a coordinate (9,9) in a 8x8 chessboard. It may lead to programming error as the variable declaration may not contain the variable (9,9).

#### 2. How does the program work?

The program's structure is quite simple. It can be divided in to several parts.

- Starting – Initial all chess and cursor and the label.
- Detecting – Detect whether it is possible to put a chess while the mouse move.
- Putting a chess – Change the color of the chess after a chess is put.
- Ending part – if either one side can put a chess, the label will show which side should put a chess; if neither one can put a chess, it will shows which side is the winner.

The program first initialize all chess in the game board and draw the lines used in the program. Hence it redraw the chessboard to make the chessboard has 4 chess only. Then the program starts to detect mouse movement and change the coordinate to a form of (1..8,1..8), which is easier for later use. Then a module checks if the user can put a chess and show the result in a label. After that, if the user presses the mouse button that means put a chess, another module will change the setting in the variable storing the state of the chess. Then the program redraws all the chess. After each chess is put, the program decides which side should put a chess. If both side cannot put a chess, the program will end and show the result. A button of restart will then shows.

#### 3. The difficulties in writing the program

- The definition of “END OF GAME”  
Many people may think that when the total number of chess on the chessboard equal to 64, then it is the end of the game. But the correct definition should be that no more chess can be put in both color's situation. I met this problem while I was writing this program. At last, I use the concept of the latter definition, so I am able to write the program that will end the game when there is no chess can be put.
- The concept of (X, Y)  
While I was writing the program, I met a problem was that the concept of X, Y coordinate problem. I first write the program with the coordinate in (X, Y) form. But later I continued my work, I found that I had written the program in the form of (Y, X). I nearly deleted half of the program to convert all the wrong (Y, X) coordinate into (X,Y) coordinate.
- How to test where can we put the chess

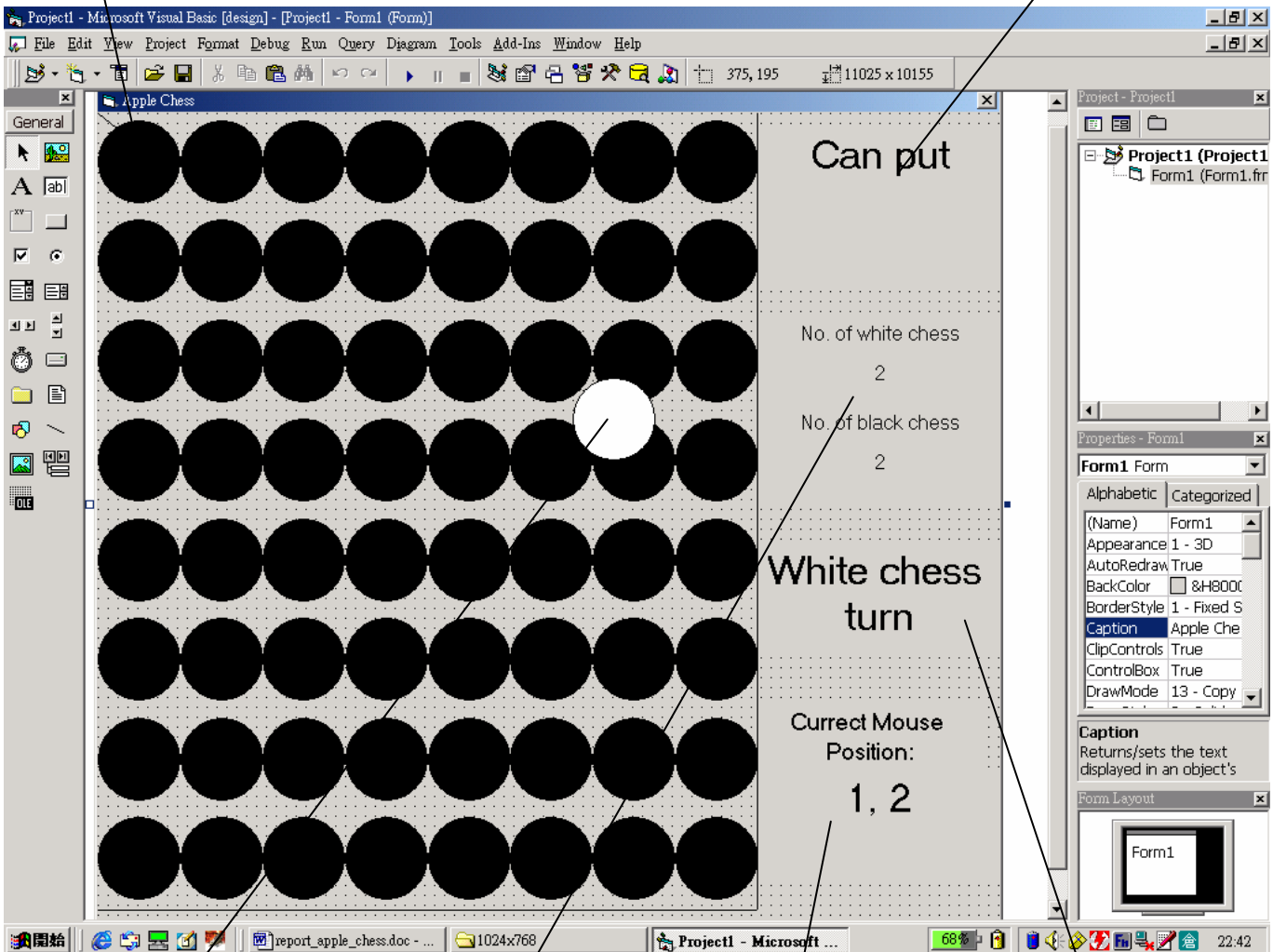
It is the most difficult problem I have met. I have never thought of this question. But later I try to write the detecting method in North, East, South and West four directions. I found that it is quite simple in the programming idea, but very difficult in programming. Later I finished that four directions and then use the concept of it to write the other four directions.

#### 4. The objects used in the program

- There are many components in the program. Let discuss it with the help of a diagram.

Totally 64 circles act as the chesses by changing it visibility and color.

It can let us know where we can put the chess when we move the mouse, the words change while the coordinate of the mouse has moved.



It is the cursor of the game. It can be white or black in color that can tell which chess should be put. It follows the mouse movement.

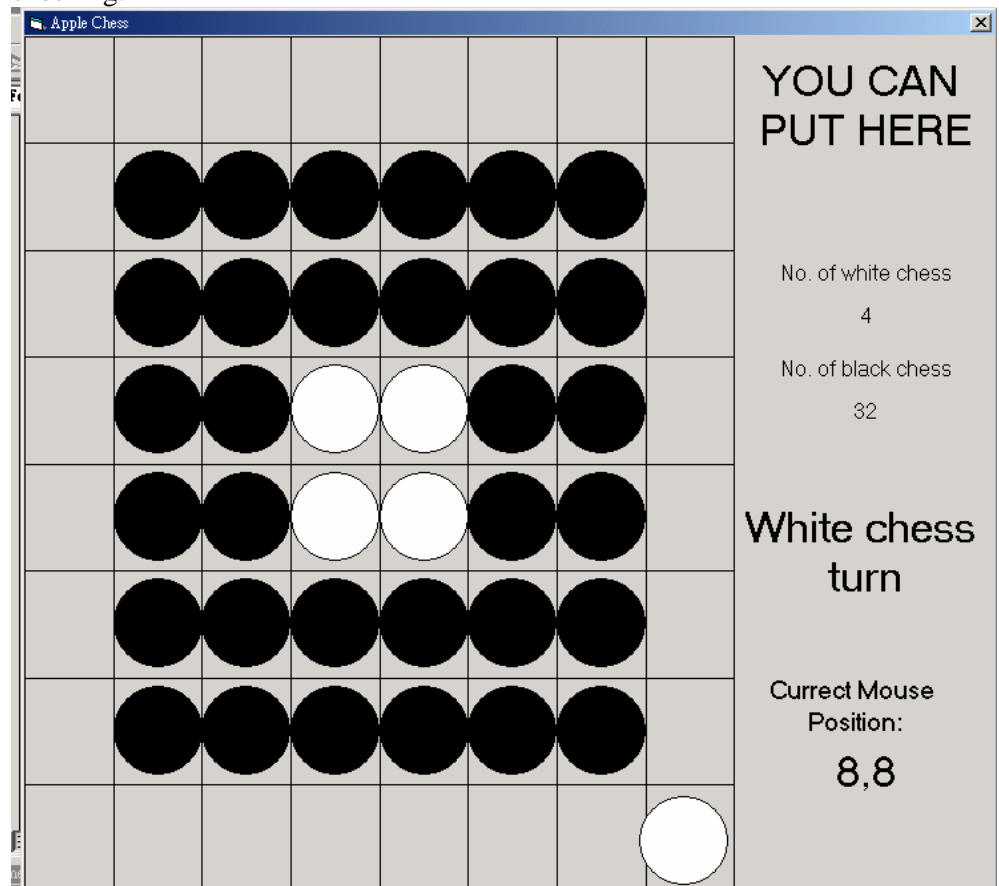
It indicates the number of chess in both white and black, updates after chess is put.

Indicate the mouse coordinate in the chessboard

Tell the user which side should put the chess now and which one win.

## 5. Testing plan and debug

- Each program part must have to pass through a testing before I do the other part. Some are listed below.
- The ability of showing the correct position of the chess
  - I changed the code to some specific arrangement. Such as I make a black chess at position 3,4, after the drawing of all chess. I can know whether the module of redraw if function and also the coordinate arrangement in the chessboard.
- The detecting module when the mouse moves.
  - A special chessboard is made by changing the source code. This arrangement is very useful as it contain all the possible direction for checking.



- Printing some variables' value at the corner of the program.
  - Some module may not function by some minor mistake that we cannot find them out easily. It can be solved by printing out some variable's value once it has changed.
- Changing color of chess
  - I tested this part of the program by the special arrangement mentioned above as it contains all the direction possible. It can also test for a chess in multi-direction after a chess is put.
- Black and white, one by one
  - I first write a procedure for checking whether the opponent can put a chess. If the opponent can, the Boolean type variable changed. I check by playing it several times. That's easy to find out mistake in this part.
- Win or lost
  - The ending of the game is another difficult part in the program. As I mentioned before, the ending means that both black and white cannot put any more chess means the end of the game. I tested by making some 60 chess's game. So I just have to put 4 chess and can do the procedure for ending.
- Debug procedure



- I divide the program into several parts, so that after running a part, the program will end automatically. By this method, I can know where the errors are and hence the debug process is much more easy.

## 6. Log book

Event	Date	Detail
Starting Date	23 December 2002 (3 hours)	Finished the structure of chessboard and declare all variables in the program.
Find data	24 December 2002 (2 hours)	Find data about how to detect whether user can put chess.
	26 December 2002	Ask my teacher about the question above in ICQ. He guided me how to do it through ICQ.
	31 December 2002 (2 hours)	Finish the sub-program of checking.
End of writing program Debug and testing.	1 December 2003 (3.5 hours)	Finish the program and debug.
	2 December 2003	Discussion of the contents of the report, writing of the report.
	3 December 2003	Discussion of the log book. Finish this log book.
	4 December 2003	The program with this report sent to my teacher.
	5 December 2003	Suggestion from my teacher to improve the report. Rewrite the report and finish it.

## 7. Improvement can be made later

- User VS Computer.
  - As the program now only support 2 players. If more playing method is available, that will be better. It is possible that make a apple chess which is user VS computer. This may increase the interest of the user.
- Network play
  - By using winsock in windows, 2 players can play in different country if they have the IP of their Internet connection.
- Animation
  - It will be more attractive if animation or some 3D chess is added. Children always like playing game with great graphic or animation.
- A web page
  - Users can play with other that is browsing the page. It can be done by using java language.