

***THE
AMAZING
CHESS***

BY – BATCH 532

Project Report

Topic : Chess

Game type: 2-Player, played on a board

Version : 1.2

Features:

- ~Includes all the basic rules of the game
- ~Includes special features like castling
- ~Introduces a new graphical interface developed on EzWindows
- ~Displays messages like “Illegal move”
- ~Display of the moves of the players on the terminal using Standard Notation of Chess

Scope for future work :

~The graphic interface could be improved by showing the range of possible moves to the user on the board.

~The game is open to extension to 1-Player.

~In the case of 1-player, it is possible to include the different levels of difficulty.

~The game could include different types like Timed Chess, Limited-Round Chess.

~Debugging of enpassant and pawn promotion is left to be done.

~Possible graphic modifications include :

1. Inclusion of the toolbar for starting a new game, saving/loading a game.
2. Introducing buttons for undo-move.
3. Display of the moves on a window integrated with the game.

PROJECT TEAM WORKING METHODOLOGY

The team divided the work according to the interest level and the understanding of each member

The meetings were held initially regularly in the lab slot of Friday, extending about 30 min over the allotted time almost every time.

The basic coding of the pieces was decided upon by the Diwali Break.

Yet, after the Diwali Break, started the problems of coding-graphic integration, along with logical errors and obviously syntax-errors.

The meetings started becoming more irregular and unorthodox, with meetings starting at night and easily extending to pre-morning time.

They were held almost 4 days a week for the last two weeks.

Basic Design and Graphic related to the game has been completed, with many possible errors being removed after extensive testing(thanks to many of our friends, who consented to play the game and gave their views on its working).

A few major errors surfacing(and removed) during this process were:

1. The main piece of the board – the king – was found to be unable to move, thanks to an error in its legal-moves.
2. The special move - castling and en passant – were found to be dysfunctional, again in this process.
3. The problem of the overwriting of array in the legal moves of the pieces was found after one long match.
4. And quite a few small ones....

DESIGN AND ALGORITHMS

Use of "Virtual Function"

- >We have declared a class with name Type.
- >It has the virtual function legal_moves as its member.
- >Then we declared classes for every type of piece as inherited from the Type class which have the actual definition of legal move function.
- >And the respective definitions will be called on calling the function.

Algorithm for setting up the board:

- >We have managed all the data according to positions.
- >Object "chess" of class "board" has arrays of colours ,pointer of pieces and ranges which respectively store the colour , type and range of the piece at a particular position.

Algorithm for movement :

->As soon as a click is made, we will call firstclick function and check if the click made is significant or not. When significant, piece is highlighted.

->Now the initial and final co-ordinates are passed to movement function which will check whether the move is legal.

->If allowed the move is made, otherwise the message "Illegal move" is displayed.

Algorithm for checking legality of a move:

->First we check whether the place selected is inside range or not.

->If inside we internally assume the move.

->Recalculate the ranges.

->Check if it leads to "check to self king".

->If not, the move is allowed.

->Otherwise board is reverted to its initial state.

Modified Algorithms

For calculating legal moves:

Earlier the algorithm used was -

->Check if the place being considered was empty or opponent , continue adding places in that direction to range array.

Though it seems fine at first sight, it included all the boxes in that direction even if it had to jump over an opponent's piece.

So the algorithm was changed as

->Continue adding places to range as long as it is empty and if the terminating block is occupied by opponent ,include it otherwise ignore it.

So all the functions were accordingly modified.

The trouble of images:

->First we “googled” cropped images of all pieces hoping it to work nice.

->But when the image was loaded , it automatically got a black background thanks to "EzWindows".

->So we had to leave the idea of loading cropped images.

->And we “googled” images with a white background and used them.

->But everything was not fine even now.

->Whenever we moved a piece , it left a white patch on the board again thanks to our "EzWindows".

->So we are drawing the image of cream or brown box as desired whenever a move is made.

CSE Lab Project 5-32 Meeting Minutes

1. Friday the 31st September, 2011 9:20 - 10:50 PM
Attended by : All

~Discussion on Different Topics for Project; Came up with 3 different topics

~Decision to think over the different topics involving ground-work and research

2. Saturday the 1st October, 2011 7:00 – 8:30 PM
Attended by : Mohit, Manish, Venkatesh, Ishaan, Prince

~Project Topic Discussion going on

3. Friday the 7th October, 2011 8:50 - 10:40 PM
Attended by : All

~Discussion on Project Topic; Finalised as "Chess"

~Group work divided; algorithm writing for different chess pieces decided upon

4. Sunday the 9th October, 2011 10:50 - 12:10 PM
Attended by : Mohit, Kshitij, Prince

~Discussion on integration of the codes for different pieces

~Discussion on the Basic Layout of the Board for the game -
using inheritance and virtual functions

~Discussion on integration of graphics with the main
program

~Division for Board and King-Check Functions

5. Wednesday the 12th October, 2011

11:10 PM - 2:00 AM

Attended by : All

~Discussion on basic idea of doing the board movement

~Discussion on the layout of the board

~Discussion on the new types of coordinate-based piece-
movements

6. Friday the 14th October, 2011

8:40 - 11:00 PM

Attended by : All

~Discussion and compilation on the algorithms of all the
chess pieces

~Division of work of king-and-pawn algorithms

7. Wednesday the 19th October, 2011

1:20 AM– 3:45 AM

Attended by : Mohit, Venkatesh, Kshitij

~Discussion on future work division with the holiday work
being divided

~Compilation of the SRS

8. Wednesday the 2nd November, 2011

9:00 P M - 10:45 PM

Attended by : All except Manas

~Discussion on the integration of the codes

~A few changes made to the codes

9. Friday the 4th November, 2011

8:40PM - 1:05 AM

Attended by : All except Manas

~Coding of the pieces

~Few more confusions on design resolved

~Graphics finalised

10. Sunday the 6th November, 2011

2:00 PM - 4:30 PM (Mohit & Venkatesh) Final Coding of pieces

4:30 PM - 6:00 PM (+ Manish & Kshitij) Images Cropping

6:00 PM-7:20 PM(-Manish) Simple Testing by Mohit,Kshitij

9:00 PM - 2:00 AM (+ Manish) Integration of graphics

11. Monday the 7th November, 2011

5:30 PM - 11:00 PM (Mohit,Kshitij,Venkatesh,Manish) Debugging

11:00 PM - 1:00 AM (- Manish) Seeking TA's help in debugging

12. Wednesday the 9th November, 2011

5:30 PM-8:30 PM(Mohit & Venkatesh)Error Identification

9:00PM-7:00 AM (+ Kshitij & Manish)Finalising special moves

7:00 AM - 8:30 AM(only Manish)Graphics Debugging

13. Thursday the 10th November, 2011

5:30 pm-8:30pm & 9:00pm-2:45am

Attended by: Mohit, Venkatesh

~Making of checkmate function

14. Friday the 11th November, 2011

8:55 PM – 3 AM

Attended by : All

~Finalisation of special moves

~Peer Review Conducted

Breakdown of time spent by the group members on the project

NAME	DISCUSSION	PROGRAMMING	TESTING	DOCUMENTATION	MISC.
Mohit	9hr	6hr 10min	44hr 25min	3hr 05min	3hr 20min
Kshitij	7hr 30min	6hr 10min	40hr 10min	5hr	2hr 10min
Manish	7hr 40min	8hr	40hr	2hr	6hr 30min
Venkatesh	7hr 40min	6hr 10min	42hr	3hr 05min	2hr
Ishaan	7hr 40min	2hr 5min	1hr	2hr	0hr
Prince	9hr	2hr 5min	1hr	2hr	0hr
Manas	6hr 10min	0hr	0hr	0hr	2hr

Note – We have included time of “debugging” in “testing”

Total time spent by group –

Mohit : 66 hrs.

Kshitij : 61 hrs.

Manish : 64hr 10min

Venkatesh : 60hr 55min

Ishaan : 12hr 45min

Prince : 14hr 5min

Manas : 8hr 10min

Final Work Division

Coding:

Pawn : Kumar Ishaan

Bishop: Love Prince Saini

Rook: Venkatesh

King: Venkatesh

Checkmate: Venkatesh

Knight: Kshitij Singh

Castling : Kshitij Singh

En Passant : Mohit Gupta

Board : Mohit Gupta

Graphics:

Manish Keshri

Testing:

Graphics : Manish Keshri

Debugging:

Coding : Mohit Gupta, Venkatesh, Kshitij Singh

Graphics : Manish Keshri, Mohit Gupta

PEER EVALUATION SYSTEM

A. Self Evaluation :

Mohit – 5

Venkatesh – 4.5

Kshitij – 4.5

Manish – 4

Ishaan – 2

Prince – 2

Manas – 0.5

B. Peer Evaluation :

Mohit – 5

Venkatesh – 4.5

Kshitij – 5

Manish – 5

Ishaan – 2

Prince – 3

Manas – 0.5

Special Credits:

Special Thanks to our Teaching Assistant Agam Sarup for his advice on the design of the game along with help in a special phase of debugging(when every member had failed to detect the error).

Bibliography:

- ➔ Introduction to OOP, By Davidson and Cohoon.
- ➔ www.google.com for chess images.
- ➔ Adobe Photoshop for editing images.