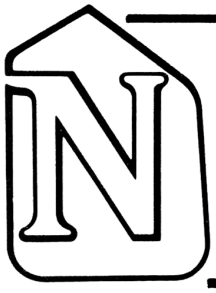


NORTHERN COLLEGE OF APPLIED ARTS AND TECHNOLOGY

LABORATORY REPORT

Graduate/Technical Report

LABORATORY

Computer Chess

EXPERIMENT

Ghislain de Blois

SUBMITTED BY

COM 6215

CLASS

May 6, 1994

DATE

LAB GROUP

EXP. NO.

1.0 **The game of chess**

1.1 **History**

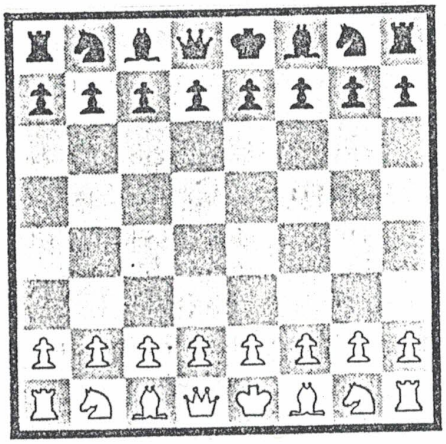
It is generally accepted that chess has a common origin with backgammon, parcheesi, and cribbage. In 500, a form of chess was known in India as chaturanga. The Persians adapted this game and was passed on to the Byzantine empire and eventually made its way to Spain by Muslims. Chess evolved from 1400 to 1600 into the modern version that we are familiar with today.

1.2 **Rules**

Each player begins with sixteen chessmen, eight "pieces" and eight "pawns", which are set up initially as shown on **Diagram 1-1**. The two players are White and Black.

Each different chessman has its own special way of moving and capturing enemy pieces. A capture is done by moving into a square occupied by an enemy piece. All captured pieces are removed from the board.

DIAGRAM 1-1



The most valuable piece, the king (**see Diagram 1-2**) may move in any direction--vertically, horizontally or diagonally--but only one square at a time. The queen (**see Diagram 1-3**) is the most powerful piece. The queen may move any distance in any direction.

DIAGRAM 1-2

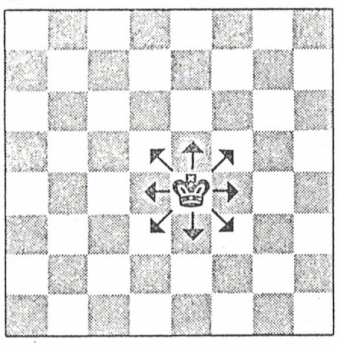
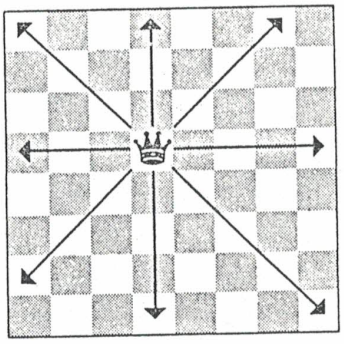


DIAGRAM 1-3



The bishop (**see Diagram 1-4**) moves any distance in any diagonal direction. The rook (**see Diagram 1-5**) moves any distance in any straight (horizontal and vertical) direction.

DIAGRAM 1-4

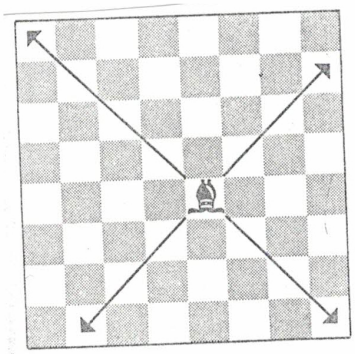
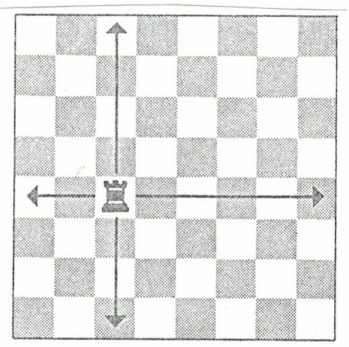


DIAGRAM 1-5



The knight (**see Diagram 1-6**) moves in an unusual way--one square vertically and two squares horizontally or one square horizontally and two squares vertically. The move is a shape of that of an "L". The knight's move is uninterrupted--it may jump over intervening pieces. The pawn (**see Diagram 1-7**) may only move forward one square at a time; unless on its first move, it may advance one or two squares. The pawn can only capture by going one square diagonally forward if there is an enemy piece on that square. If a pawn reaches the end of the chess board, it may be promoted to a Queen, Rook, Knight or Bishop--at the player's choice.

DIAGRAM 1-6

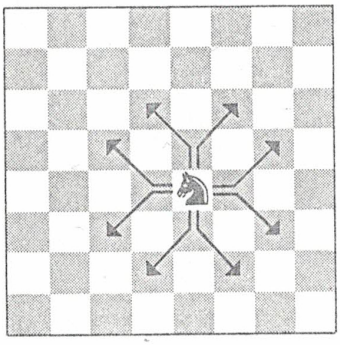
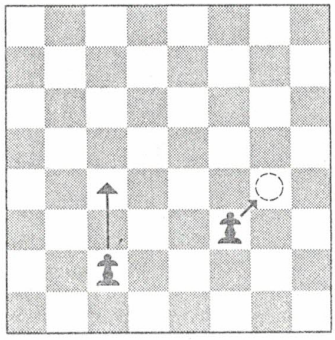


DIAGRAM 1-7



The object of the game is to checkmate the enemy king--to get it into a position where it cannot escape capture. A threat to capture the enemy king places it in "check". The king must then be moved or otherwise protected. If the checked king can't be moved to a free and safe square, if the attacking piece cannot be captured, or if one of the other pieces cannot be placed in front of the king as protection, it is checkmated, and the game is lost.

A "stalemate" occurs when a player, although not in check, cannot move any of his pieces without bringing his king into check. The game is then a draw. A draw can also occur when neither player has enough men left on the

board to checkmate the other. "Perpetual check" occurs when a player who cannot checkmate the other king with the king having no possibility of escape. This too is a draw. A game may be also declared a draw by mutual agreement of the players.

Castling is a special move and may only be made once per game. The move is executed by moving the king two squares toward the rook and the placing on the square passed over by the kind. Either rook may be used in castling. The move is only allowed only if neither the rook being castled with has yet to be moved from its original square. The squares where the castle is being must be vacant. The king's trajectory towards the rook while castling must be free of enemy attack (no enemy piece must be attacking the square which the king passes through while castling). The king may not castle while in check.

"En-passant" is another special move. If a pawn makes the double advance for its first move, an enemy pawn that could have captured it had the first pawn moved only one square may capture it "en passant". But the en passant capture may only be made on the immediate turn, not after.

1.3 **Concepts**

Though checkmate is the object of the game, a player can rarely gain it without first getting a good advantage over the opponent in some conjunction of the elements of space, time, and force.

Space. The player who controls the greater number of squares can maneuver his men more freely and restrict his opponent's moves. When pieces can reach most squares from the center of the board, it is most important to control the squares in the center.

Time. The player who places his chessmen most effectively, first "develops" a position from which to strike effectively. So it's important to get each chessman into its ideal place without loss of value of one turn to play (or half-move).

Force. The capture of an enemy chessman is the most "tangible" of advantages. With the advantage of one piece, a good player will win' a master can win with the advantage of one pawn. The exchange of a man for a more valuable one can also be very useful.

The opening. Mobilizing the chessmen for attack and defense. There are many ways to "open" a chess game. The most common opening moves in chess are e2-e4 for white then followed by e7-e5 for black

Middle game. The heart of chess. It is the means of winning after the mobilizing of the chessmen and before the "end-game" sets in. There are tactical concepts such as "pins" and "forks" in which a player attacks two enemy men or points at the same time.

End game. After the reduction of forces, the thing left to do is either how to checkmate or how to force a pawn through to promotion of queen.

2-0 **Teaching a computer how to play chess**

The program I wrote which I am describing in this section was written in BASIC. The following sections, **2-1 Representing rules,** and **2-2 The algorithm** describe in detail how the program works. **2-3 Running the program** is a short user's guide on how to run the program and use it.

2-1 **Representing rules**

The first thing we need in a game of chess is an 8*8 squared board. Normally a two-dimensional array would be used to represent this chess board but I've decided that a one-dimensional chess board would simplify things greatly because only one number would be needed to represent a direction (see **Diagram 2-3** to see why). Note that the board (see **Diagram 2-1**) is 12*12. This is because knights can jump up to 2 spaces and a "border" is put there to preventing pieces from going one side of the board to the other and "jumping" off the board. The numbers 1-16 represent the white pieces and the numbers 25-40 represent the black pieces. The 99's represent the

DIAGRAM 2-1

99	99	99	99	99	99	99	99	99	99	99	99
99	99	33	34	35	36	37	38	39	40	99	99
99	99	25	26	27	28	29	30	31	32	99	99
99	99	00	00	00	00	00	00	00	00	99	99
99	99	00	00	00	00	00	00	00	00	99	99
99	99	00	00	00	00	00	00	00	00	99	99
99	99	00	00	00	00	00	00	00	00	99	99
99	99	01	02	03	04	05	06	07	08	99	99
99	99	09	10	11	12	13	14	15	16	99	99
99	99	99	99	99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99	99	99	99	99

DIAGRAM 2-2

PIECE	N	S	W	E	NW	NE	SE	SW	NWW	NNW	NNE	NEE	SEE	SSE	SSW	SWW	TOTAL
PAWN	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
ROOK	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	15
BISHOP	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	240
QUEEN	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	255
KING	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	255
KNIGHT	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	65280

```
V = (# OF WHITE PAWNS)*1 - (# OF BLACK PAWNS)*1
    + (# OF WHITE KNIGHTS AND BISHOPS)*3
    - (# OF BLACK KNIGHTS AND BISHOPS)*3
    + (# OF WHITE ROOKS)*5 - (# OF BLACK ROOKS)*5
    + (# OF WHITE QUEENS)*9 - (# OF BLACK QUEENS)*9
    + (WHITE KING)*1000 - (BLACK KING)*1000
    + (# SQUARES CONTROLLED BY WHITE)*0.1
    - (# SQUARES CONTROLLED BY BLACK)*0.1
    + (# CENTER SQUARES CONTROLLED BY WHITE)*0.1
    - (# CENTER SQUARES CONTROLLED BY BLACK)*0.1
    + (# BLACK UNPROTECTED PAWNS)*0.25
    - (# WHITE UNPROTECTED PAWNS)*0.25
    + (# BLACK UNPROTECTED KNIGHTS AND BISHOPS)*0.75
    - (# WHITE UNPROTECTED KNIGHTS AND BISHOPS)*0.75
    + (# BLACK UNPROTECTED ROOKS)*1.25
    - (# WHITE UNPROTECTED ROOKS)*1.25
    + (# BLACK UNPROTECTED QUEENS)*2.25
    - (# WHITE UNPROTECTED QUEENS)*2.25
    + (BLACK KING IN "CHECK")*6.25
    - (WHITE KING IN "CHECK")*6.25
```

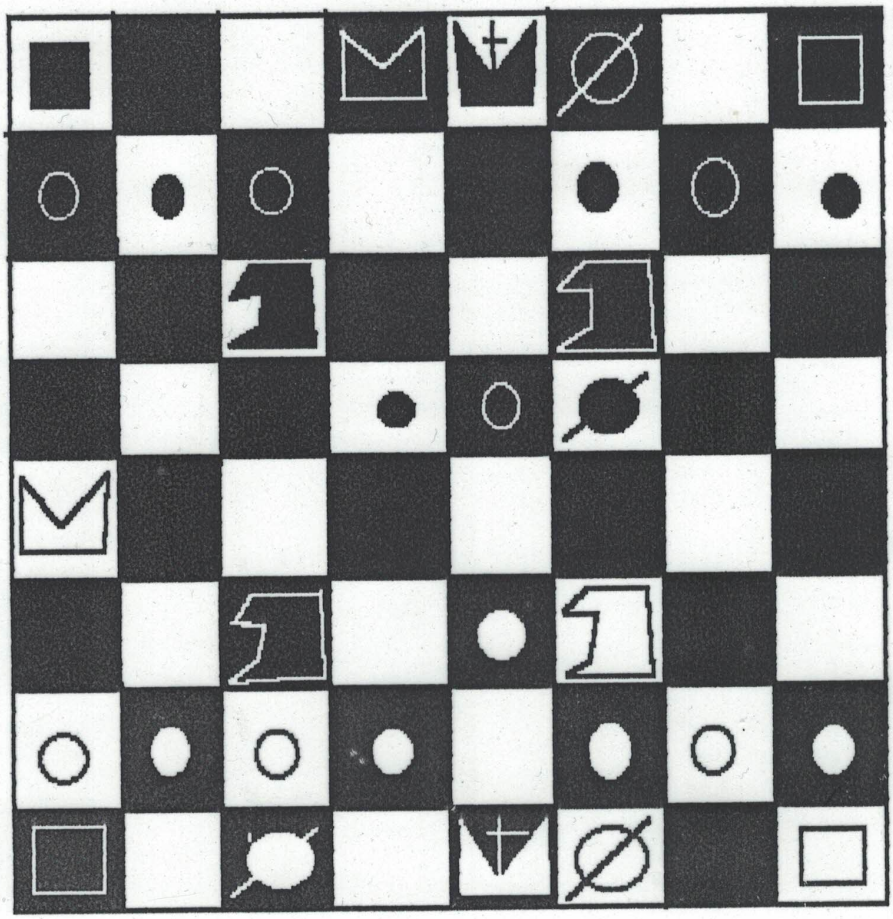


DIAGRAM 2-3

N	S	W	E	NW	NE	SE	SW	NWW	NNW	NNE	NEE	SEE	SSE	SSW	SWW
12	+12	-1	+1	-13	-11	+13	+11	-14	-25	-23	-10	+14	+25	+23	+10

DIAGRAM 2-4

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	2	4	8	16	32	64	128
513	2	516	8	16	16416	64	16512
256	1024	2048	7680	30720	4096	8192	32768
0	256	2048	4096	2048	4096	32768	0

```
BIT 0:PAWN#1      8:ROOK#1       16:QUEEN#2
    1:PAWN#2      9:KNIGHT#1     17:QUEEN#3
    2:PAWN#3     10:BISHOP#1     18:QUEEN#4
    3:PAWN#4     11:QUEEN#1      19:QUEEN#5
    4:PAWN#5     12:KING         20:QUEEN#6
    5:PAWN#6     13:BISHOP#1     21:QUEEN#7
    6:PAWN#7     14:KNIGHT#2     22:QUEEN#8
    7:PAWN#8     15:ROOK#2       23:QUEEN#9
```

2:PAWN#3	10:BISHOP#1	18:QUEEN#4
3:PAWN#4	11:QUEEN#1	19:QUEEN#5
4:PAWN#5	12:KING	20:QUEEN#6
5:PAWN#6	13:BISHOP#1	21:QUEEN#7
6:PAWN#7	14:KNIGHT#2	22:QUEEN#8
7:PAWN#8	15:ROOK#2	23:QUEEN#9

This is the initial "think move board" for white at the start of the game.

2-2 **The algorithm**

The wonderful thing about computers is that they can be easily programmed to do exact problems such as simple mathematical problems. The sad thing about computers is that it's not so easy to program them to do inexact (or "common sense") problems. Although to some, chess may not be "common sense", to the experienced chess player, "common sense" may mean that it is foolish to exchange a queen for a pawn, for example. How do you represent this in computer language?

Let's assume that the computer plays white in a chess game. First we must construct a "think move board" for white (discussed in 2-1) for the computer to know which moves will be available to it. Then the computer tries every possible move on a "think board" (a board duplicated in memory from the original, real board to do "think" moves on without altering the original board--not

to be confused with the "think **move** board"), and when it tries a move on its "think board", the computer will try to simulate black's best possible move (by first constructing a black "think move board" of course) to see the outcome of the tested move. The computer does this for every possible move for white and then chooses the best move. The program I wrote for this report thinks in two half-moves ahead (or one full move ahead).

How does the program determine the best possible move? By the use of an algorithm--a special calculation to give a "score" or value of a move. The scoring system for the algorithm in my chess program is as follows: one point for every pawn, 3 for every knight or bishop, 5 for every rook, 9 for every queen (remember--pawns may be promoted) and a very high value for the king (since when you lose a king in chess, you lose the game. But I give the king in my program a 25 for other uses). These are just "tangible" values of a position in chess. There are also "intangible" values such as, control of squares, the rank of a pawn (or how far down the board a pawn is) and control of the center. The calculation for the value of a move (for white) is done as follows:

V = (# OF WHITE PAWNS)*1 - (# OF BLACK PAWNS)*1
+ (# OF WHITE KNIGHTS AND BISHOPS)*3

```
- (# OF BLACK KNIGHTS AND BISHOPS)*3
+ (# OF WHITE ROOKS)*5 - (# OF BLACK ROOKS)*5
+ (# OF WHITE QUEENS)*9 - (# OF BLACK QUEENS)*9
+ (WHITE KING)*1000 - (BLACK KING)*1000
+ (# SQUARES CONTROLLED BY WHITE)*0.1
- (# SQUARES CONTROLLED BY BLACK)*0.1
+ (# CENTER SQUARES CONTROLLED BY WHITE)*0.1
- (# CENTER SQUARES CONTROLLED BY BLACK)*0.1
+ (# BLACK UNPROTECTED PAWNS)*0.25
- (# WHITE UNPROTECTED PAWNS)*0.25
+ (# BLACK UNPROTECTED KNIGHTS AND BISHOPS)*0.75
- (# WHITE UNPROTECTED KNIGHTS AND BISHOPS)*0.75
+ (# BLACK UNPROTECTED ROOKS)*1.25
- (# WHITE UNPROTECTED ROOKS)*1.25
+ (# BLACK UNPROTECTED QUEENS)*2.25
- (# WHITE UNPROTECTED QUEENS)*2.25
+ (BLACK KING IN "CHECK")*6.25
- (WHITE KING IN "CHECK")*6.25
```

To translate this formula for black, just replace white with black and black with white. Also not mentioned in this formula are values involved in castling. In my program, moving an uncastled king costs a move value of 0.9 and "castling" a king adds to the move value by 0.9.

All the official rules of chess are used in my chess program except one; my program cannot "en-passant". To see my program play chess, see the following section.

2-3 **Running the program**

To run the chess program you require:

-an IBM PC home computer with a 486 microprocessor

-a disk drive

Insert the disk into a floppy disk drive.

Set the DOS prompt default drive to the drive with the disk with the chess program on it.

Type **chess** and hit the **ENTER** key.

After going through he title screen you should see:

DIAGRAM 2-5

```
  A B C D E F G H
 ****************
8*r n b q k b n r
7*p p p p p p p p
6*. . . . . . . .
5*. . . . . . . .
4*. . . . . . . .
3*. . . . . . . .
2*P P P P P P P P
1*R N B Q K B N R
```

The capital letters represent the white pieces and the lower-case letters represent the black pieces. **P** represents pawn, **N** represents knight (it can't use K because king begins with a K), **B** represents bishop, **R** represents the rook, **Q** represents the queen and **K** represents the king. If this appearance of a chess board

appears odd to you, use a real chess board to assist you.

The computer will prompt you to hit the enter key after the completion of every move. On a 486, the program takes approximately 2 to 2.5 minutes to do a move.

To stop the program, the program must be running while it is thinking of a move. To do this, hit the letter "A" (if you're not in **CAPS LOCK** mode, hold down the **SHIFT** key and hit the letter "A").

Conclusion

In this report I've accomplished what I've set out to do: I wrote a chess program. It is not as sophisticated as the commercial chess programs out there but I think it plays at a fair level.

I've learned many things in writing this chess program. For one, I quickly realised how important it is to use structured programming. Without structured programming, I would not have completed my chess program as fast as I would have used non-structured programming because with structured programming it is very easy to "debug" a program. I've also learned that it is **very** wise to write down every single variable you use in a program and what it does. In doing this, it is far easier to understand your own programs when you refer to them later.

If I could change one thing in my program I would make it more efficient in its way for checking all possible moves. Instead of scanning all possibilities, I would have excluded some of them when analyzing the opponent's best possible move to a possible move. What I mean is, on the first move for example, there are 20 possible moves for white initially. What I could have done is instead of scanning all 20 available moves for black's response, 4 of white's best available moves are scanned for black's response.

APPENDIX A: **PSEUDO CODE**

MAIN

```
START
DO INITIALIZATION
DO UNTIL CHESS GAME IS LOST OR DRAW
     DISPLAY CHESS BOARD
     DO WHITE
     MOVE IS WHITE'S MOVE
     DO CWHITE
     CHANGE DATA IN ARRAYS FOR WHITE'S MOVE
     DISPLAY CHESS BOARD
     DO BLACK
     MOVE IS BLACK'S MOVE
     DO CBLACK
     CHANGE DATA IN ARRAYS FOR BLACK'S MOVE
END DO
END
```

WHITE

```
START
CLEAR WHITE THINK MOVE BOARD ARRAY
DO UNTIL ALL SPACES ON BOARD HAVE BEEN EXAMINED
     EXAMINE NEXT SPACE
     IF WHITE PIECE ON SPACE
          TURN ON BITS ON WHITE THINK MOVE BOARD ACCORDING TO RULES
     END IF
END DO
END
```

BLACK

```
START
CLEAR BLACK THINK MOVE BOARD ARRAY
DO UNTIL ALL SPACES ON BOARD HAVE BEEN EXAMINED
     EXAMINE NEXT SPACE
     IF BLACK PIECE ON SPACE
          TURN ON BITS ON BLACK THINK MOVE BOARD ACCORDING TO RULES
     END IF
END DO
END
```

CWHITE

```
START
COPY REAL BOARD TO THINK BOARD
```

```
COPY REAL WHITE THINK MOVE BOARD TO TEMPORARY WHITE THINK MOVE
BOARD
DO UNTIL ALL POSSIBLE MOVES FOR WHITE HAVE BEEN EXAMINED
     EXAMINE NEXT POSSIBLE MOVE
     DO POSSIBLE MOVE ON THINK BOARD
     DO WHITE
     DO BLACK
     CALCULATE WHITE SCORE ACCORDING TO ALGORITHM
     IF IT IS WHITE'S MOVE
          DO CBLACK
          TOTAL SCORE=WHITE SCORE-BLACK SCORE
     END IF
     IS IT THE BEST MOVE SO FAR?
          RECORD CURRENT MOVE AS BEST POSSIBLE MOVE FOR WHITE
     END IF
END DO
END
```

CBLACK

```
START
COPY REAL BOARD TO THINK BOARD
COPY REAL BLACK THINK MOVE BOARD TO TEMPORARY BLACK THINK MOVE
BOARD
DO UNTIL ALL POSSIBLE MOVES FOR BLACK HAVE BEEN EXAMINED
     EXAMINE NEXT POSSIBLE MOVE
```

```
     DO POSSIBLE MOVE ON THINK BOARD
     DO BLACK
     DO WHITE
     CALCULATE BLACK SCORE ACCORDING TO ALGORITHM
     IF IT IS BLACK'S MOVE
          DO CWHITE
          TOTAL SCORE=BLACK SCORE-WHITE SCORE
     END IF
     IS IT THE BEST MOVE SO FAR?
          RECORD CURRENT MOVE AS BEST POSSIBLE MOVE FOR BLACK
     END IF
END DO
END
```

INITIALIZATION

```
START
     CREATE ARRAY FOR MOVE CODES
     CREATE ARRAY FOR MOVE DIRECTION CODES
     CREATE ARRAY FOR CHESSMEN STATUS (NOT MOVED, MOVED, CAPTURED)
     CREATE ARRAY FOR CHESSMEN POSITIONS FOR BLACK AND WHITE
     CREATE ARRAY FOR REAL BOARD
     CREATE ARRAY FOR THINK BOARD
     CREATE ARRAY FOR WHITE AND BLACK THINK MOVE BOARDS
END
```

APPENDIX B: **CODING**

```
'* COMPUTER CHESS
'* TECHNICAL REPORT
'* by GHISLAIN DE BLOIS

START:
     SCREEN 1
     CLS
     FOR X=0 TO 75
          LINE (X,X)-(319-X,X),1
          LINE (319-X,X)-(319-X,199-X),2
          LINE (319-X,199-X)-(X,199-X),3
          LINE (X,199-X)-(X,X),2
     NEXT X
     LOCATE 12,14
     PRINT"COMPUTER CHESS"
     LOCATE 13,11
     PRINT"by Ghislain de Blois"
     LOCATE 14,13
```

```
     PRINT"TECHNICAL REPORT"
     GOSUB KEYP

ASK:
     CLS
     GOTO ASK1
     PRINT"IS WHITE (H)UMAN OR (C)OMPUTER?"
     GOSUB KEYP
     PW=1
     IF A$="C" THEN PW=0
     IF PW=1 THEN PRINT"HUMAN"
     IF PW=0 THEN PRINT"COMPUTER"
     PRINT
     PRINT"IS BLACK (H)UMAN OR (C)OMPUTER?"
     GOSUB KEYP
     PB=0
     IF A$="H" THEN PB=1
     IF PB=1 THEN PRINT"HUMAN"
     IF PB=0 THEN PRINT"COMPUTER"
     PRINT
     PRINT"HIT ANY KEY"
     GOSUB KEYP
ASK1:
     GOSUB INIT

MLOOP:
```

```
     MOVE=MOVE+1
     GOSUB BOARD
     PRINT"WHITE'S MOVE"
     INPUT"ENTER MOVE";A$
     IF A$="X" THEN GOTO ENDEND
     GOSUB WHITE
     JAA=1
     GOSUB CWHITE
     GOSUB BOARD
     PRINT"BLACK'S MOVE"
     INPUT"ENTER MOVE";A$
     GOSUB BLACK
     JAA=0
     GOSUB CBLACK
     GOTO MLOOP
BOARD:
     FOR X=2 TO 9
          FOR Y=2 TO 9
               A(0,Y*12+X)=0
          NEXT Y
     NEXT X
     FOR Z=0 TO 23
          X=PW(Z)
          Y=PB(Z)
          IF PM(Z,0)<>2 THEN A(0,X)=Z+1
          IF PM(Z,1)<>2 THEN A(0,Y)=Z+25
```

```
     NEXT Z
     CLS
BOARD2:
     PRINT"MOVE #";MOVE
     PRINT
     PRINT"  A B C D E F G H"
     PRINT"*****************"
     FOR Y=9 TO 2 STEP -1
     PRINT MID$("12345678",Y-1,1);"*";
          FOR X=2 TO 9
               PRINT MID$(BB$,A(0,Y*12+X)+1,1);" ";
          NEXT X
          PRINT
     NEXT Y
     RETURN

KEYP:
     A$=INKEY$
     IF A$="" THEN KEYP
     RETURN

WHITE:
     FOR X=0 TO 143
          A(1,X)=0
     NEXT X
     FOR X=24 TO 119
```

```
          Z=A(0,X)
          IF Z>0 AND Z<25 THEN GOSUB WHITE1
     NEXT X
     RETURN
WHITE1:
     FOR A=0 TO 15
          B=MC(Z-1,1)
          IF B>255 AND A>7 THEN WHITE2
          IF B<256 AND A<8 THEN IF ((2^A) AND B)=(2^A) THEN WHITE2
          GOTO WHITE3
WHITE2:
     FOR B=1 TO MC(Z-1,0)
          IF Z<9 AND PM(Z-1,0)=0 AND A(0,61+Z)=0 AND A(0,49+Z)=0
THEN
               A(1,61+Z)=A(1,61+Z)+2^(Z-1):S1=S1+J
          END IF
          X0=X-MDU(A)*B
          IF A(0,X0)=99 THEN B=0:GOTO WHITE3
          IF A(0,X0)=0 THEN A(1,X0)=A(1,X0)+2^(Z-1):S1=S1+J
          IF A(0,X0)>0 AND Z>8 THEN
               A(1,X0)=A(1,X0)+2^(Z-1):B=0:S1=S1+J:GOTO WHITE3
          END IF
          IF Z<9 THEN
               IF A(0,X+11)<>99 THEN
                    IF A(0,X+11)>0 THEN
                         A(1,X+11)=A(1,X+11)+2^(Z-1):S1=S1+J
```

```
                    END IF
               END IF
               IF A(0,X+13)<>99 THEN
                    IF A(0,X+13)>0 THEN
                         A(1,X+13)=A(1,X+13)+2^(Z-1):S1=S1+J
                    END IF
               END IF
          END IF
          IF X0=65 OR X0=66 OR X0=77 OR X0=78 THEN
               IF A(1,X0)>1 THEN S1=S1+.1
          END IF
     NEXT B
 WHITE3:
     NEXT A
     RETURN

BLACK:
     FOR X=0 TO 143
          A(2,X)=0
     NEXT X
     FOR X=24 TO 119
          Z=A(0,X)
          IF Z>24 AND Z<49 THEN GOSUB BLACK1
     NEXT X
     RETURN
BLACK1:
```

```
     FOR A=0 TO 15
          B=MC(Z-25,1)
          IF B>255 AND A>7 THEN BLACK2
          IF B<256 AND A<8 THEN IF ((2^A) AND B)=(2^A) THEN BLACK2
          GOTO BLACK3
BLACK2:
     FOR B=1 TO MC(Z-25,0)
          IF  Z-24<9  AND  PM(Z-25,1)=0  AND  A(0,49+Z)=0  AND
A(0,61+Z)=0 THEN
               A(2,49+Z)=A(2,49+Z)+2^(Z-25):S1=S1+J
          END IF
          X0=X+MDU(A)*B
          IF A(0,X0)=99 THEN B=0:GOTO BLACK3
          IF A(0,X0)=0 THEN
               A(2,X0)=A(2,X0)+2^(Z-25):S1=S1+J
          END IF
          IF A(0,X0)>0 AND Z-24>8 THEN
               A(2,X0)=A(2,X0)+2^(Z-25):B=0:S1=S1+J:GOTO BLACK3
          END IF
          IF Z-24<9 THEN
               IF A(0,X-13)<>99 THEN
                    IF A(0,X-13)>0 THEN
                         A(2,X-13)=A(2,X-13)+2^(Z-25):S1=S1+J
                    END IF
               END IF
               IF A(0,X-11)<>99 THEN
```

```
                    IF A(0,X-11)>0 THEN
                        A(2,X-11)=A(2,X-11)+2^(Z-25):S1=S1+J
                    END IF
                END IF
            END IF
            IF X0=65 OR X0=66 OR X0=77 OR X0=78 THEN
                IF A(2,X0)>0 THEN S1=S1+.1
            END IF
        NEXT B
BLACK3:
        NEXT A
        RETURN

CWHITE:
        WOO=0
        WOOO=0
        IF PM(12,0)=0 THEN
            GOSUB BLACK
            IF PM(8,0)=0 THEN
                II=0
                FOR X=27 TO 29
                    IF A(0,X)>0 THEN II=1
                NEXT X
                FOR X=28 TO 30
                    IF A(2,X)>0 THEN II=1
                NEXT X
```

```
     END IF
     IF II=0 THEN WOOO=1
     IF PM(15,0)=0 THEN
          II=0
          FOR X=31 TO 32
               IF A(0,X)>0 THEN II=1
          NEXT X
          FOR X=30 TO 31
               IF A(2,X)>0 THEN II=1
          NEXT X
     END IF
     IF II=0 THEN WOO=1
END IF
FOR X=0 TO 143
     A(3,X)=A(0,X)
     A(4,X)=A(1,X)
NEXT X
BESTVAL1=-1000
BESTPIECE1=0
BESTPOS1=0
FOR X8=26 TO 117
     IF A(3,X8)>0 AND A(3,X8)<25 THEN CWHITE3
     IF A(4,X8)=0 OR A(3,X8)=99 THEN CWHITE3
     YY1=A(4,X8)
     FOR Y8=23 TO 0 STEP-1
          FOR X=0 TO 143
```

```
     A(0,X)=A(3,X)
NEXT X
II=0
IF WOO=1 THEN
     A(0,30)=0
     A(0,31)=16
     A(0,32)=13
     A(0,33)=0
     WOO=2
     II=1
     GOTO CWHITEA
END IF
IF WOOO=1 THEN
     A(0,26)=0
     A(0,27)=0
     A(0,28)=13
     A(0,29)=9
     A(0,30)=0
     WOOO=2
     II=2
     GOTO CWHITEA
END IF
          ZZ1=INT(YY1/(2^Y8))
          IF ZZ1<1 THEN GOTO CWHITE2
           YY1=YY1-2^Y8
A(0,PW(Y8))=0
```

```
                A(0,X8)=Y8+1
CWHITEA:
                J=.1
                S1=0
                GOSUB WHITE
                J=-.1
                GOSUB BLACK
                FOR KJ=24 TO 119
                     IF A(0,KJ)=0 OR A(0,KJ)=99 THEN CWHITE4
                     JK=A(0,KJ)
                     IF JK>24 THEN JK=JK-25:GOTO CWHITE5
                     JK=JK-1
                     IF PM(JK,0)<>2 THEN
                          IF A(0,KJ)>0 AND A(0,KJ)<25 THEN
                          S1=S1+PV(JK)
                          IF A(2,KJ)>0 THEN
                               S1=S1-PV(JK)*.25
                               IF A(1,KJ)>0 THEN
                                    S1=S1+PV(JK)*.25
                               END IF
                          END IF
                          END IF
                     END IF
                     GOTO CWHITE4
CWHITE5:
                     IF PM(JK,1)<>2 THEN
```

```
                    IF A(0,KJ)>24 THEN
                    S1=S1-PV(JK)
                    IF A(1,KJ)>0 THEN
                         S1=S1+PV(JK)*.25
                         IF A(2,KJ)>0 THEN
                              S1=S1-PV(JK)*.25
                         END IF
                    END IF
                    END IF
               END IF
CWHITE4:
          IF  A(0,KJ)=13  AND  A(2,KJ)>0  OR  PM(12,0)=2  THEN
S1=-1000
          NEXT KJ
          FOR HJ=0 TO 7
               GH=0
               FOR ZI=38+HJ TO 110+HJ STEP 12
               GH=GH+.05
               IF ZI>109 THEN GH=9
               IF A(0,ZI)>0 AND A(0,ZI)<9 THEN S1=S1+GH
               NEXT ZI
          NEXT HJ
          IF JAA=0 THEN GOTO CWHITE9
          FOR FUL=0 TO 143
               A(5,FUL)=A(3,FUL)
               A(6,FUL)=A(4,FUL)
```

```
                NEXT FUL
                BB2=Y8
                PP2=X8
                SS2=S1
                GOSUB CBLACK
                FOR FUL=0 TO 143
                     A(3,FUL)=A(5,FUL)
                     A(4,FUL)=A(6,FUL)
                NEXT FUL
                S1=SS2-BESTVAL
                Y8=BB2
                X8=PP2
                LOCATE 18,1
                PRINT INT(S1*100)/100;Y8;X8
                IF Y8=12 AND PM(12,0)=0 AND II=0 THEN S1=S1-.9
                IF II>0 THEN S1=S1+.9
CWHITE9:
                IF S1>BESTVAL1 THEN
                     BESTVAL1=S1
                     BESTPIECE1=Y8
                     BESTPOS1=X8
                     IF II=1 THEN
                          BESTPOS1=7000
                     END IF
                     IF II=2 THEN
                          BESTPOS1=8000
```

```
                    END IF
                END IF
CWHITE2:
          A$=INKEY$:IF A$="A" THEN GOTO ENDEND
          NEXT Y8
CWHITE3:
     GOTO WHJ
WHJ:
     NEXT X8
     IF JAA=0 THEN RETURN
     IF BESTPOS1=7000 OR BESTPOS1=8000 THEN CWHITEB
     PW(BESTPIECE1)=BESTPOS1
     PM(BESTPIECE1,0)=1
     IF A(3,BESTPOS1)>24 THEN
          L2=A(3,BESTPOS1)-25
          PM(L2,1)=2
          PB(L2)=0
     END IF
     IF A(0,BESTPOS1)<8 AND BESTPOS1>109 THEN
          PW(BESTPIECE1)=0
          PM(BESTPIECE1,0)=2
          PM(BESTPIECE1+16,0)=1
          PW(BESTPIECE1+16)=BESTPOS1
     END IF
CWHITEB:
     IF BESTPOS1=8000 THEN
```

```
        PW(8)=29
        PW(12)=28
        PM(8,0)=1
        PM(12,0)=1
    END IF
    IF BESTPOS1=7000 THEN
        PW(12)=32
        PW(15)=31
        PM(12,0)=1
        PM(15,0)=1
    END IF
    RETURN

CBLACK:
    BOO=0
    BOOO=0
    IF PM(12,1)=0 THEN
        GOSUB WHITE
        IF PM(8,1)=0 THEN
            III=0
            FOR X=111 TO 113
                IF A(0,X)>0 THEN III=1
            NEXT X
            FOR X=112 TO 114
                IF A(1,X)>0 THEN III=1
            NEXT X
```

```
     END IF
     IF III=0 THEN BOOO=1
     IF PM(15,0)=0 THEN
          III=0
          FOR X=115 TO 116
               IF A(0,X)>0 THEN III=1
          NEXT X
          FOR X=114 TO 115
               IF A(1,X)>0 THEN III=1
          NEXT X
     END IF
     IF III=0 THEN BOO=1
END IF
FOR X=0 TO 143
     A(3,X)=A(0,X)
     A(4,X)=A(2,X)
NEXT X
BESTVAL=-1000
BESTPIECE=0
BESTPOS=0
FOR X9=26 TO 117
     IF A(3,X9)>24 OR A(4,X9)=0 THEN CBLACK3
     YY=A(4,X9)
     FOR Y9=23 TO 0 STEP -1
          FOR X=0 TO 143
               A(0,X)=A(3,X)
```

```
NEXT X
III=0
IF BOO=1 THEN
     A(0,114)=0
     A(0,115)=40
     A(0,116)=37
     A(0,117)=0
     BOO=2
     III=1
     GOTO CBLACKA
END IF
IF BOOO=1 THEN
     A(0,110)=0
     A(0,111)=0
     A(0,112)=37
     A(0,113)=33
     A(0,114)=0
     BOOO=2
     III=2
     GOTO CBLACKA
END IF
ZZ=INT(YY/(2^Y9))
IF ZZ<1 THEN GOTO CBLACK2
YY=YY-2^Y9
A(0,PB(Y9))=0
A(0,X9)=Y9+25
```

```
CBLACKA:
                J=.1
                S1=0
                GOSUB BLACK
                J=-.1
                GOSUB WHITE
                FOR KJ=24 TO 119
                     IF A(0,KJ)=0 OR A(0,KJ)=99 THEN CBLACK4
                     JK=A(0,KJ)
                     IF JK<25 THEN JK=JK-1:GOTO CBLACK5
                     JK=JK-25
                     IF PM(JK,1)<>2 THEN
                          IF A(0,KJ)>24 THEN
                          S1=S1+PV(JK)
                          IF A(1,KJ)>0 THEN
                               S1=S1-PV(JK)*.25
                               IF A(2,KJ)>0 THEN
                                    S1=S1+PV(JK)*.25
                               END IF
                          END IF
                          END IF
                     END IF
                     GOTO CBLACK4
CBLACK5:
                     IF PM(JK,0)<>2 THEN
                          IF A(0,KJ)>0 AND A(0,KJ)<25 THEN
```

```
                    S1=S1-PV(JK)
                    IF A(2,KJ)>0 THEN
                        S1=S1+PV(JK)*.25
                        IF A(1,KJ)>0 THEN
                            S1=S1-PV(JK)*.25
                        END IF
                    END IF
                    END IF
                END IF
CBLACK4:
            IF A(0,KJ)=37 AND A(1,KJ)>0 OR PM(12,1)=2 THEN
                S1=-1000
            END IF
            NEXT KJ
            FOR HJ=0 TO 7
                GH=0
                FOR ZI=98+HJ TO 26+HJ STEP -12
                GH=GH+.05
                IF ZI<34 THEN GH=9
                IF A(0,ZI)>24 AND A(0,ZI)<33 THEN S1=S1+GH
                NEXT ZI
            NEXT HJ
            IF JAA=1 THEN GOTO CBLACK9
            FOR FUL=0 TO 143
                A(5,FUL)=A(3,FUL)
                A(6,FUL)=A(4,FUL)
```

```
                NEXT FUL
                SS2=S1
                BB2=Y9
                PP2=X9
                GOSUB CWHITE
                FOR FUL=0 TO 143
                     A(3,FUL)=A(5,FUL)
                     A(4,FUL)=A(6,FUL)
                NEXT FUL
                S1=SS2-BESTVAL1
                Y9=BB2
                X9=PP2
                LOCATE 18,1
                IF Y9=12 AND PM(12,1)=0 AND III=0 THEN S1=S1-.9
                IF III>0 THEN S1=S1+.9
                PRINT INT(S1*100)/100;Y9;X9
CBLACK9:
                IF S1>BESTVAL THEN
                     BESTVAL=S1
                     BESTPIECE=Y9
                     BESTPOS=X9
                     IF III=1 THEN
                          BESTPOS=7000
                     END IF
                     IF III=2 THEN
                          BESTPOS=8000
```

```
                    END IF
               END IF
CBLACK2:
          A$=INKEY$:IF A$="A" THEN GOTO ENDEND
          NEXT Y9
CBLACK3:
          GOTO ZZB
ZZB:
          NEXT X9
          IF JAA=1 THEN RETURN
          IF BESTPOS=7000 OR BESTPOS=8000 THEN CBLACKB
          PB(BESTPIECE)=BESTPOS
          PM(BESTPIECE,1)=1
          IF A(3,BESTPOS)>0 AND A(3,BESTPOS)<25 THEN
               L2=A(3,BESTPOS)-1
               PM(L2,0)=2
               PW(L2)=0
          END IF
          IF A(0,BESTPOS)>24 AND A(0,BESTPOS)<33 AND BESTPOS<34
THEN
               PB(BESTPIECE)=0
               PM(BESTPIECE,1)=2
               PM(BESTPIECE+16,1)=1
               PB(BESTPIECE+16)=BESTPOS
          END IF
CBLACKB:
```

```
          IF BESTPOS=8000 THEN
               PB(8)=113
               PB(12)=112
               PM(8,1)=1
               PM(12,1)=1
          END IF
          IF BESTPOS=7000 THEN
               PB(12)=116
               PB(15)=115
               PM(12,1)=1
               PM(15,1)=1
          END IF
          RETURN

INIT:
     REM *** INDIVIDUAL MOVE CODES FOR PIECES
     DIM MC(23,1)
     FOR X=0 TO 23
          READ MC(X,0)
          READ MC(X,1)
     NEXT X
        REM *** MOVE DIRECTIONS FOR MOVE CODES
        DIM MDU(15)
        FOR X=0 TO 15
          READ MDU(X)
        NEXT X
```

```
REM *** SET UP BOARD
DIM PM(23,1)
DIM PW(23)
DIM PB(23)
FOR X=0 TO 7
  PW(X)=38+X
  PB(X)=98+X
NEXT X
FOR X=8 TO 15
  PW(X)=18+X
  PB(X)=102+X
NEXT X
FOR X=16 TO 23
  PM(X,0)=2
  PM(X,1)=2
NEXT X
REM *** DEFINE BOARD
DIM A(6,143)
FOR X=0 TO 143
  A(0,X)=99
NEXT X
REM *** PIECE VALUES
DIM PV(23)
FOR X=0 TO 23
  READ PV(X)
NEXT X
```

```
      BB$=".PPPPPPPPRNBQKBNRQQQQQQQQpppppppprnbqkbnrqqqqqqqq"


      DATA 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
      DATA 7,15,1,65280,7,240,7,255,1,255,7,240,1,65280,7,15
      DATA 7,255,7,255,7,255,7,255,7,255,7,255,7,255,7,255

      DATA -12,12,-1,1,-13,-11,13,11,-14,-25,-23,-10,14,25,23,10

      DATA 1,1,1,1,1,1,1,1,5,3,3,9,25,3,3,5,9,9,9,9,9,9,9,9

      RETURN

CHECK:
      SCREEN 2
      FOR X=2 TO 9:FOR Y=2 TO 9
      PRINT A(2,Y*12+X);:NEXT Y:PRINT:NEXT X
      GOSUB KEYP
      SCREEN 1
      RETURN
CHECK2:
      SCREEN 2
      FOR X=2 TO 9:FOR Y=2 TO 9
      PRINT A(1,Y*12+X);:NEXT Y:PRINT:NEXT X
      GOSUB KEYP
      SCREEN 1
```

```
      RETURN
ENDEND:
      STOP
```

APPENDIX C: **SAMPLE GAME**

When you run the chess program, these will be the following moves you see when my chess program plays against itself. I've recorded the first 12 moves in the standard algebraic chess notation. Also included are notes on why the chess program reacted in the way it did.

	WHITE	**BLACK**
1.	e2-e3	e7-e6
2.	d1-f3	d7-d5
3.	f3-f4	b8-c6
4.	b1-c3	e6-e5
5.	f4-a4	c8-f5
6.	g1-f3	g8-f6
7.	f3-e5(*1)	f8-b4
8.	e5xc6	b7xc6
9.	a4xb4(*2)	a7-a5
10.	b4-f4	f5xc2
11.	f1-e2	d8-e7
12.	O-O (castle)	

(*1) Black thought that pawn on e5 was protected by the knight on c6. It did not have the foresight to see that if it moved the

knight on c6, it would have put its king in check

(*2) Black didn't have the foresight to see that with loss of knight on c6 that it would leave its bishop on b4 unprotected.

APPENDIX D: **BIBLIOGRAPHY**

Botvinnik, M.M., Computers, chess and long-range planning, New York, Springer-Verlag, 1970.

Halsey, William, Collier's Encyclopaedia, volume 6, pages 190-191, New York, Macmillan Educational Company, 1991.

University of Chicago, Compton's Encyclopaedia, volume 4, pages 305-306, Chicago, Encyclopaedia Britannica, 1990.