GEBZE TECHNICAL UNIVERSITY

COMPUTER ENGINEERING

CSE222-2021

WINTER PROJECT REPORT

AHMET FURKAN KURBAN

1801042674

Problem Definition

It will be a game that can be played in Android environment.The computer (or user2) makes a new move and draws the board again as the same as you did in homeworks. In this project, however, the move should be legal and you must make it “smart”. You may use the minimax algorithm to make the moves smart. The parameters for the minimax algorithm (easy, nominal, difficult, and master) and the parameters for the game (size, colors, who starts first) should be adjustable thru the interface. Note that you will need heuristic board evaluation functions, which will be very important for the whole system.

Problem Solution Approach

Firstly i created interface by using xml part for main activity.I added some necessary buttons like (easy, nominal, difficult, and master) and the parameters for the game (size, colors, who starts first) I did be adjustable thru the interface of main activity.Then i created the interface that game will be played.There are four button (UNDO-SAVE-LOAD-RESET) and Board buttons.Game codes will be explain below.

Code Explanation Part

```
button2.setOnCheckedChangeListener(new  CompoundButton.OnCheckedChangeListener()  {
    public  void  onCheckedChanged(CompoundButton  buttonView,  boolean  isChecked)  {
        button3.setVisibility(VISIBLE);
        button4.setVisibility(VISIBLE);
        button5.setVisibility(VISIBLE);
        button6.setVisibility(VISIBLE);
    }
});
```

Here when click computer vs user button easy medium diffucult expert buttons will be visible .

```
public                  void                  launchgame(View                  v){
    Intent                  i=new                  Intent(this,Gamestart.class);
    String
input=((EditText)findViewById(R.id.editTextTextPersonName)).getText().toString();
    size                        =                        Integer.valueOf(input);
    if(button1.isChecked()){
        gametype=false;
    }
    if(button2.isChecked()){
        gametype=true;
    }
    if                                                    (button7.isChecked()){
        whofirst=false;
    }
    if                                                    (button8.isChecked()){
        whofirst=true;
    }
    i.putExtra("game_size",size);
    i.putExtra("game_type",gametype);
    if                      (button3.isChecked())                      level=1;
    if                      (button4.isChecked())                      level=2;
    if                      (button5.isChecked())                      level=3;
    if                      (button6.isChecked())                      level=4;
    i.putExtra("level_game",level);
    i.putExtra("whofirst",whofirst);
    startActivity(i);
}
```

I create new intet then i get size with input then some checkes which button isClicked i get information with this and lastly i send these informations to the gamestart activity to play.

```
DisplayMetricsdisplayMetrics=newDisplayMetrics();
getWindowManager().getDefaultDisplay().getMetrics(displayMetrics);
maxlength=displayMetrics.heightPixels;
maxwidth=displayMetrics.widthPixels;
intbuttonwidth=2*maxwidth/(3*size-1);
inty=(maxlength-maxwidth)/2;
buttons=newButton[size][size];
buttons1=newButton[4];
for(intj=0;j<size;j++){
    temp=width;
    for(intk=0;k<size;k++){
        buttons[j][k]=newButton(this);
```

```
        buttons[j][k].setLayoutParams(newRelativeLayout.LayoutParams(buttonwidth-
5,buttonwidth-5));
        buttons[j][k].setX(width);
        buttons[j][k].setY(y);
        buttons[j][k].setText("*");
        buttons[j][k].setId(100+10*j+k);
        buttons[j][k].setBackgroundColor(LTGRAY);
        buttons[j][k].setOnClickListener(getOnClick());
        width+=buttonwidth;
        relativeLayout.addView(buttons[j][k]);
    }
    y+=buttonwidth;
    width=temp+buttonwidth/2;
}
```

Here I found from ınternet phone height and width(displayMetrics.heightPixels,displayMetrics.widthPixels) so i design buttons positions to the this dimensions .

```
public void play(int number1,int number2){
    if(getCurrentPlayer()=='x'){//'x'
        if(buttons[number1][number2].getText()=="*") {
            buttons[number1][number2].setText("x");
            buttons[number1][number2].setBackgroundColor(BLUE);
            moves[getlastpoint()][0] = number2;
            moves[getlastpoint()][1] = number1;
            lastpoint++;//counter for moves
            controll=false;
        }
        else{
            controll=true;
            Toast.makeText(getApplicationContext(),"This place is not
empty!",Toast.LENGTH_SHORT).show();
        }
    }
```

here current player can be x or o firstly i check the button that have positions that sended with parameter is empty then i add i change its color and i saved moves arrays to hide previous moves for undo function.

First Part

```
public void undo_game(){
    if(lastpoint!=0 && lastpoint!=1){
        buttons[moves[lastpoint-1][1]][moves[lastpoint-1][0]].setText("*");
        buttons[moves[lastpoint-1][1]][moves[lastpoint-
1][0]].setBackgroundColor(LTGRAY);
        lastpoint--;
        buttons[moves[lastpoint- 1][1]][moves[lastpoint - 1][0]].setText("*");
        buttons[moves[lastpoint-1][1]][moves[lastpoint-
1][0]].setBackgroundColor(LTGRAY);
        lastpoint--;
```

Second Part

```
        for (int i=0;i<size;i++){
            for(int j=0;j<size;j++){
                intcolor(ColorDrawable)buttons[i][j].getBackground()).getColor();
                if(color==MAGENTA){
```

```
                    if(buttons[i][j].getText()=="O") {
                        buttons[i][j].setBackgroundColor(RED);
                        buttons[i][j].setText("o");
                    }
                    else if(buttons[i][j].getText()=="X"){
                        buttons[i][j].setBackgroundColor(BLUE);
                        buttons[i][j].setText("x");
                    }
                }
            }
        }
    }
```

Here we can make undo using positons at moves array the at first part,at second part if ended game is saved to the file then i load it if i make undo it returns old situations and you can make move to end the game again it is work i load screen shot too.

```
public void save_file();
outputFile.write(getSize()+"\n");
outputFile.write("\n");
outputFile.write("USERVSUSER\n");
outputFile.write("USERVSCOMPUTER\n");
```

Codes is too much so i just will explain a little bit ,here i create file then i write size to the file buttons text user or computer vs lastly i write moves to the file.

```
public void is_valid(){
    boolean valid=false;
    while(valid==false){
        if(getSize()<6){
            Toast.makeText(getApplicationContext(),"Please enter the board size
,the size can be bigger than 5*5 ",Toast.LENGTH_SHORT).show();
            setSize(6);
        }
        valid=true;
    }
}
```

Check size is not lower than 6.

```
public void boardArray(){
    hexCells=new char[size][size];
    for(int row=0;row<size;row++){
        for(int col=0;col<size;col++){
            CharSequence text = buttons[row][col].getText();
            if ("*".equals(text)) {
                hexCells[row][col] = '.';
            } else if ("x".equals(text)) {
                hexCells[row][col] = 'o';
            } else if ("o".equals(text)) {
                hexCells[row][col] = 'x';
            }
        }
    }
}
```

This function for playing against AI ,after every moves of AI i create a new array and ı add to the array the current table status to check the game and to make good decision.

```
public void load_file();
end_game1=end_game_User1();
if(end_game1==true){
    AlertDialog.Builder builder1 = new AlertDialog.Builder(Gamestart.this);
    builder1.setTitle("HEXGAME");
    builder1.setMessage("User1 is win restart again or check moves with dismiss");
    builder1.setNegativeButton("DISMISS", null);
    builder1.setPositiveButton("NEW GAME", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialogInterface, int i) {
            reset_game();
        }
    });
    builder1.show();
}
```

I read the datas saved before then upper i check if loaded game from file is already ended i check it before and i send alert dialog to new game or dissmis to control board and maybe reset or undo shortly whatever you want.

```
public boolean end_game_User1(){
    int i,j=0;
    boolean control=false;
    for(i=0;i<getSize();i++){
        if(buttons[i][j].getText()=="x"){
            iterate_func1();
            setTempi(i);
            setTempj(j);
            control=control_moves_x();
            if(control){
                up_words_x();
                return true;
            }
        }
    }
    return false;
}
```

If there is one x at first column i start control because the game was ended only all column is full then i check moves with control moves ,control moves function is starting to check with founded first column positions then itontrolling 6 places around if there is one x around it calls this function again (recursion) after ended last column.lastly with upwords function change the color and text that reaching the result and connected buttons.upwords function is working like control moves function logis is same(recursion).

```
private void playAI(){
    boardArray();
    int [] move = BestMove(MaximizingPlayer);
    hexCells[move[0]][move[1]]='x';
    buttons[move[0]][move[1]].setBackgroundColor(RED);
    buttons[move[0]][move[1]].setText("o");
```

```
    moves[lastpoint][1]=move[0];
    moves[lastpoint][0]=move[1];
    lastpoint++;
}
```

Here the positions returned from best move function is attended to buttons and ı saved it moves array for undo.

```
private int [] BestMove(int player){
    if (player == MaximizingPlayer) {
        for (i = 0; i < size; i++)
            for (j = 0; j < size; j++) {
                if (hexCells[i][j] == '.') {
                    hexCells[i][j] = 'x'; //
                    moveVal=minimax(level,maxval,minval,false);
                    hexCells[i][j] = '.';
                    if (moveVal > maxval) {
                        bestMove[0] = i;
                        bestMove[1] = j;
                        maxval = moveVal;
                    }
                }
            }
    }
    else {
        for (i = 0; i < size; i++)
            for (j = 0; j < size; j++) {
                if ( hexCells[i][j] == '.') {
                    hexCells[i][j] = 'o';
                    moveVal=minimax(level,minval,maxval,true);
                    hexCells[i][j] = '.';
                    if (moveVal < minval) {
                        bestMove[0] = i;
                        bestMove[1] = j;
                        minval = moveVal;
                    }
                }
            }
    }

    return bestMove;
}
```

Here function access all buttons with hexcells array and find best move with minimax function i will explain it below.

```
int minimax(int depth, int alpha,int beta,boolean maxTurn) {
    if (maxTurn) {
        int maxEval = -Integer.MAX_VALUE;
        for (int x = 0; x < size; x++){
            for (int y = 0; y < size; y++){
                if (hexCells[x][y] == '.') {
                    hexCells[x][y] = 'x';
                    int eval =minimax(depth - 1,alpha,beta, false);
                    maxEval = maximum(maxEval, eval);
                    alpha = maximum(alpha,eval);
                    hexCells[x][y] = '.';
                    if(beta <= alpha)
                        break;
                }
```

```
            }
            if(beta<=alpha)
                break;
        }
        return maxEval;
    } else {/* minimizing player */
        int minEval = Integer.MAX_VALUE;
        for (int x = 0; x < size; x++){
            for (int y = 0; y < size; y++) {
                if (hexCells[x][y] == '.') {
                    hexCells[x][y] = 'o';
                    int eval = minimax(depth - 1,alpha,beta, true);
                    minEval = minimum(minEval, eval);
                    beta = minimum(beta,eval);
                    hexCells[x][y] = '.';
                    if(beta<=alpha) break;
                }

            }
            if(beta<=alpha) break;

        }
        return minEval;
    }
}
```

with Alpha-Beta Minimax algorithm. We will create an agent that can successfully compete with humans in the classic Hex game. After the end of this article, you will be able to create adversarial search agents that can competitively play zero-sum and perfect information games

Initially we check if we reached the final depth or if the board is full. If so, we return the heuristic score (we'll get back to it later).Minimax core. We iterate through possible moves and evaluate them recursively computing minimax scores. If it's still unclear to you, check this out.

For maximizing player we set

alpha = max(alpha, bestValue)

and for a minimizing player we set

beta = min(beta, bestValue)

For both cases we then evaluate

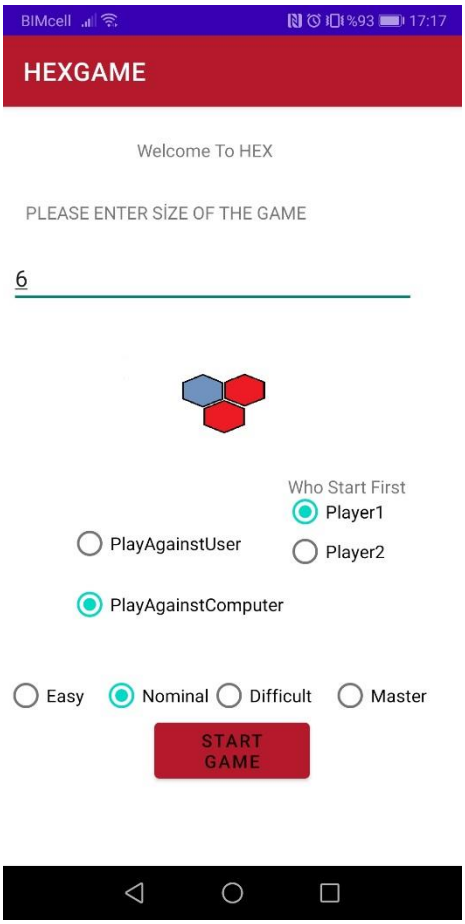
```
if beta <= alpha {
    break
}
```

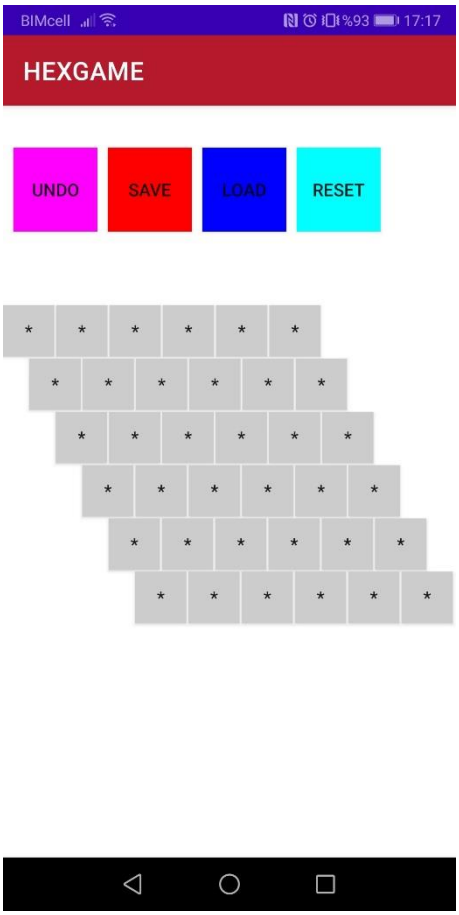
If we break from the loop, it means that we don't have to evaluate further branches (α-β cutoff). This is where we optimize the whole process.

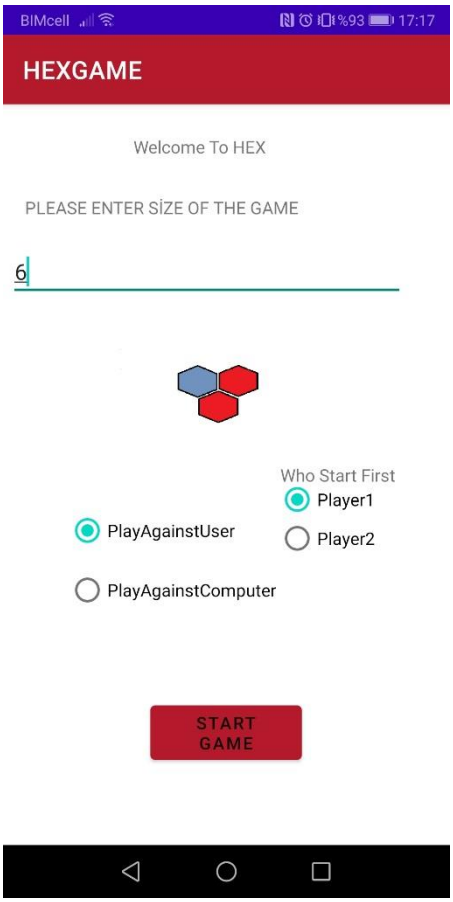
If we don't find any possible moves, we return the current heuristic score.

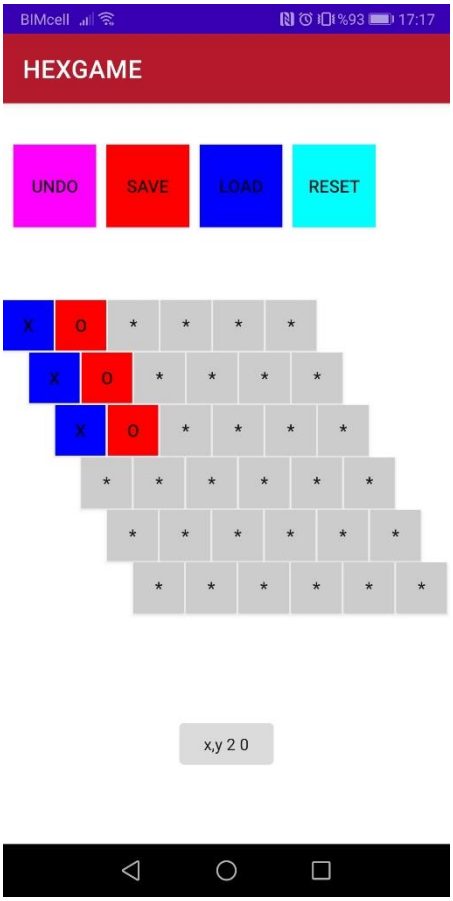
But there is one thing missing - heuristic score i.e score returned for a given board state.

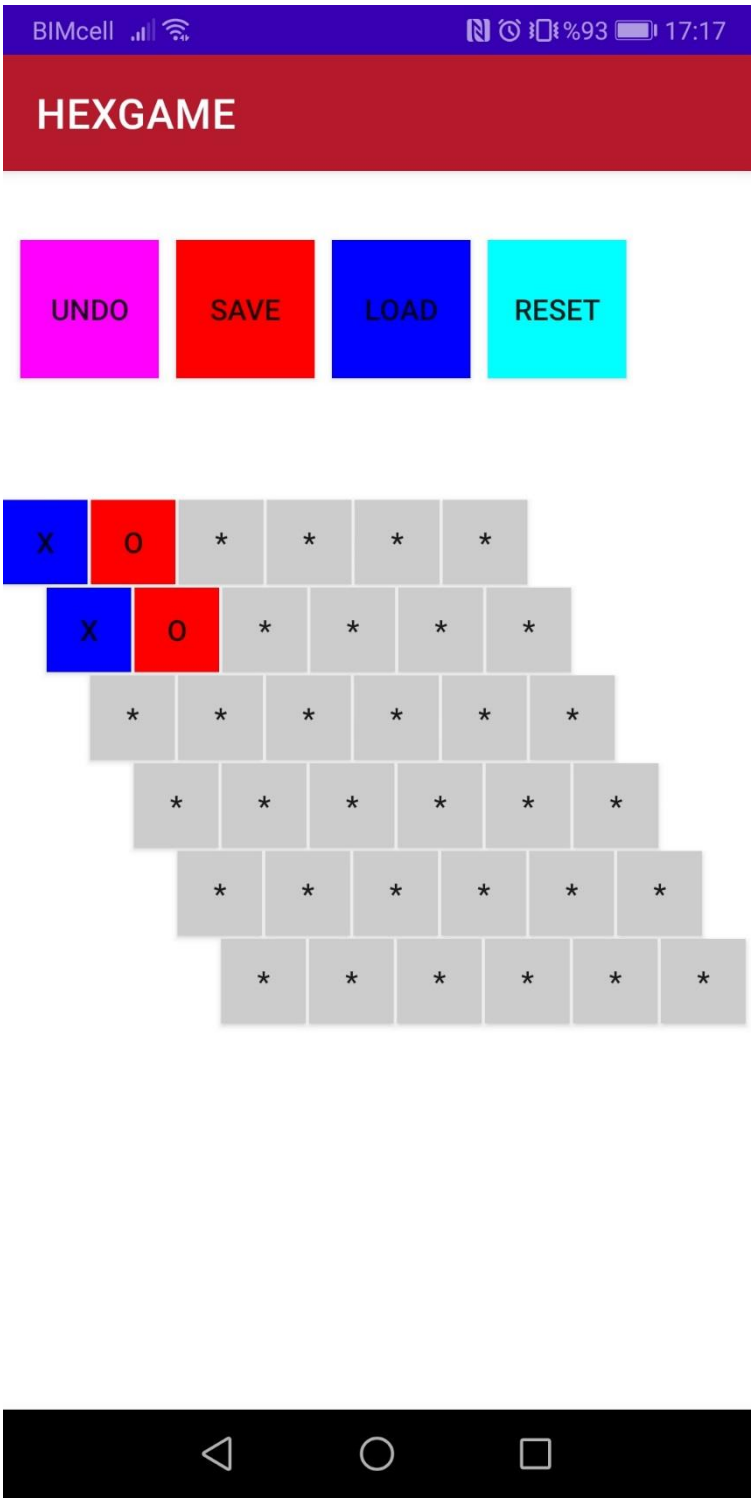
SCREEN SHOOTS PART


BIMcell
%93
17:17
HEXGAME
Welcome To HEX
PLEASE ENTER SİZE OF THE GAME
6
Who Start First
Player1
Player2
PlayAgainstUser
PlayAgainstComputer
Easy
Nominal
Difficult
Master
START GAME

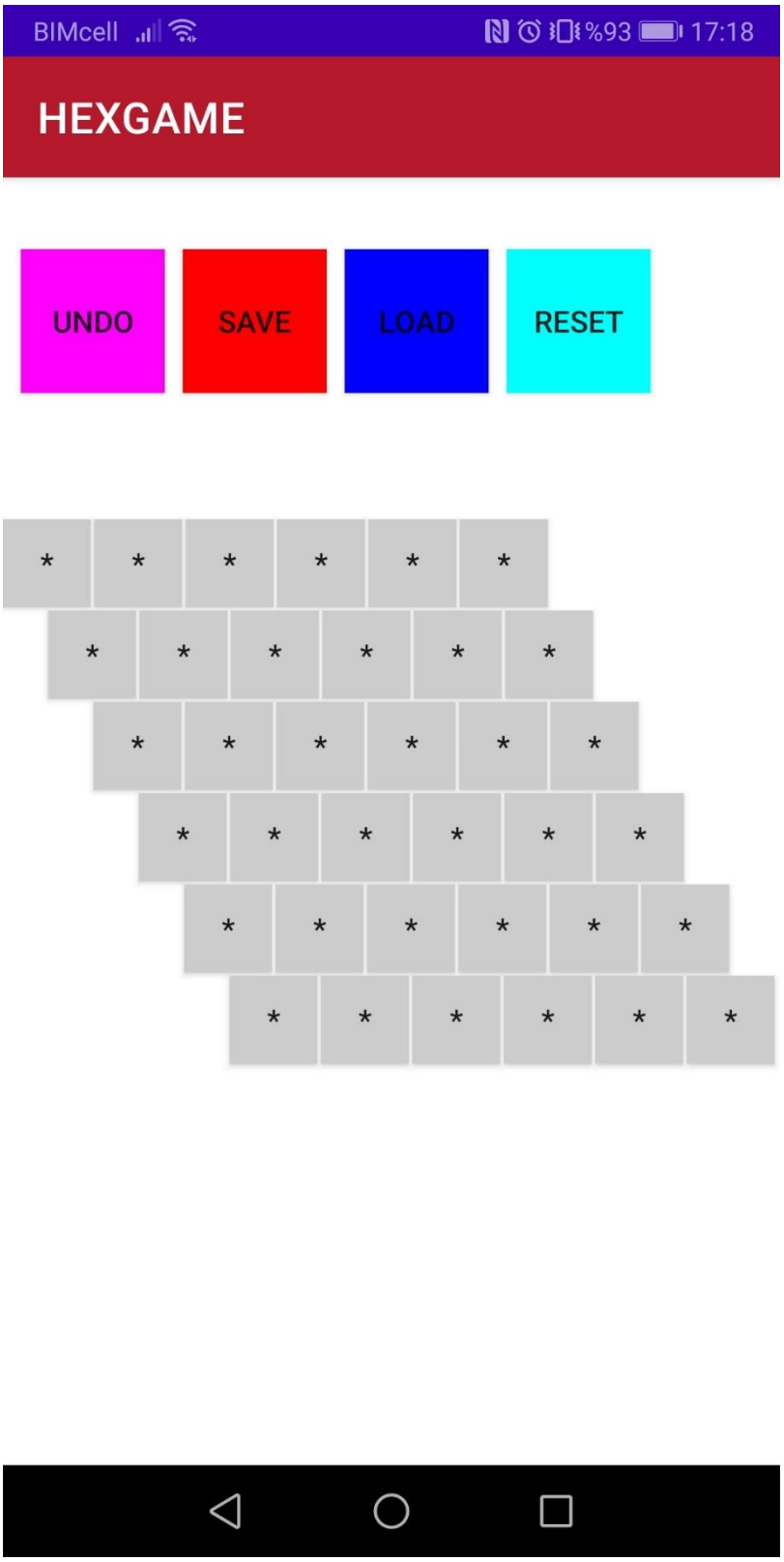

BIMcell
%93
17:17
HEXGAME
UNDO
SAVE
LOAD
RESET


BIMcell
%93
17:17
HEXGAME
Welcome To HEX
PLEASE ENTER SİZE OF THE GAME
6
Who Start First
Player1
Player2
PlayAgainstUser
PlayAgainstComputer
START GAME

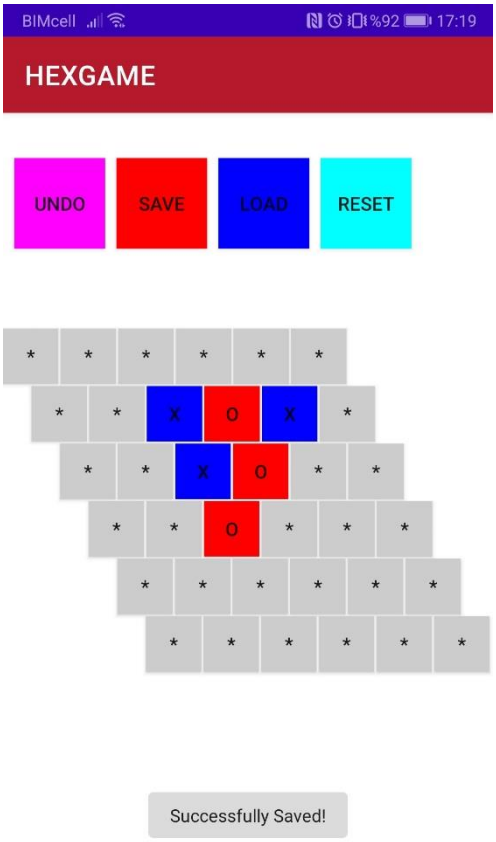

BIMcell
%93
17:17
HEXGAME
UNDO
SAVE
LOAD
RESET
x,y 2 0



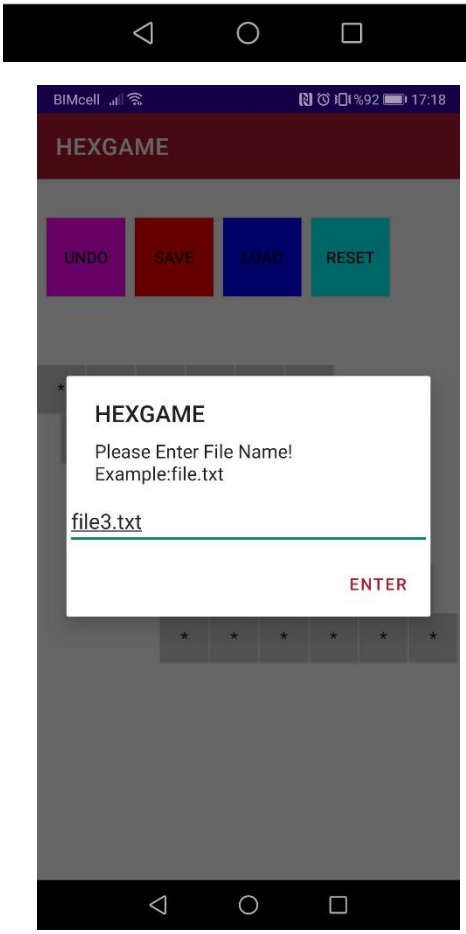
Here ı press the undo button and make undo one move.



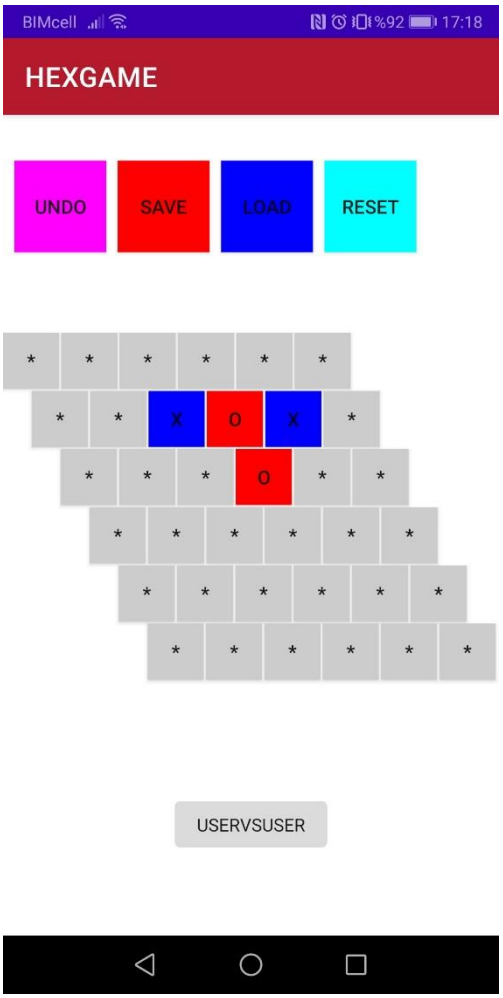
Here I press the reset button and all private variable and game buttons have been empty moves array is deleted.



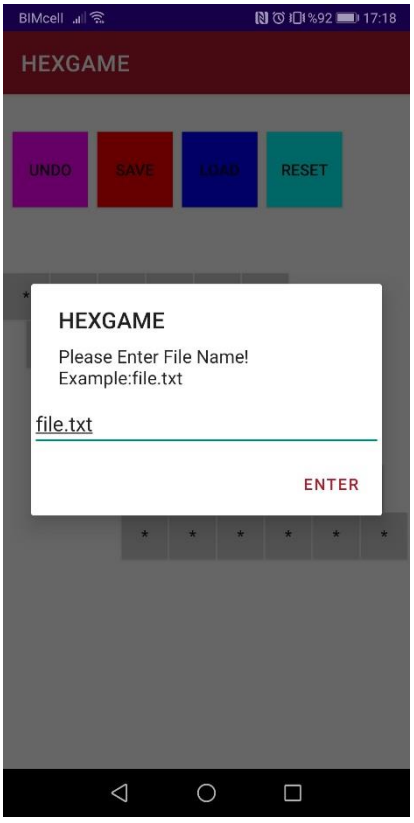
Here ı press the save button and saved to the file current board situation datas and private varible used at the game.

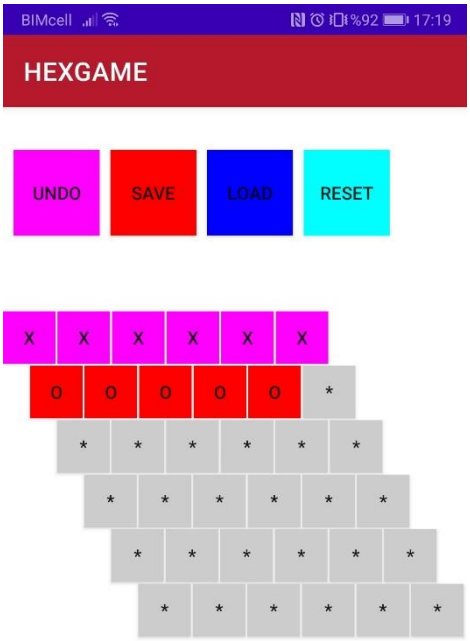


Here i write the file name and saved to this file after pressing the save button.

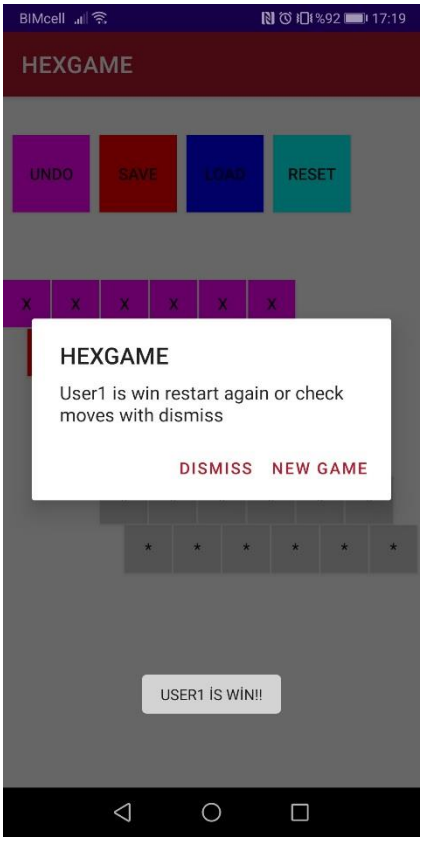


Here ı press the load button and loaded from the file current board situation datas and private varible used at the game.

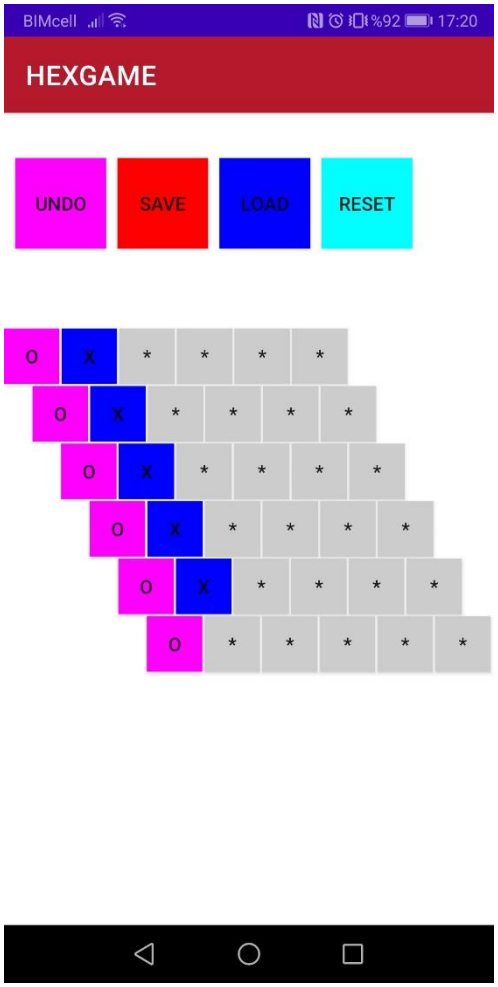


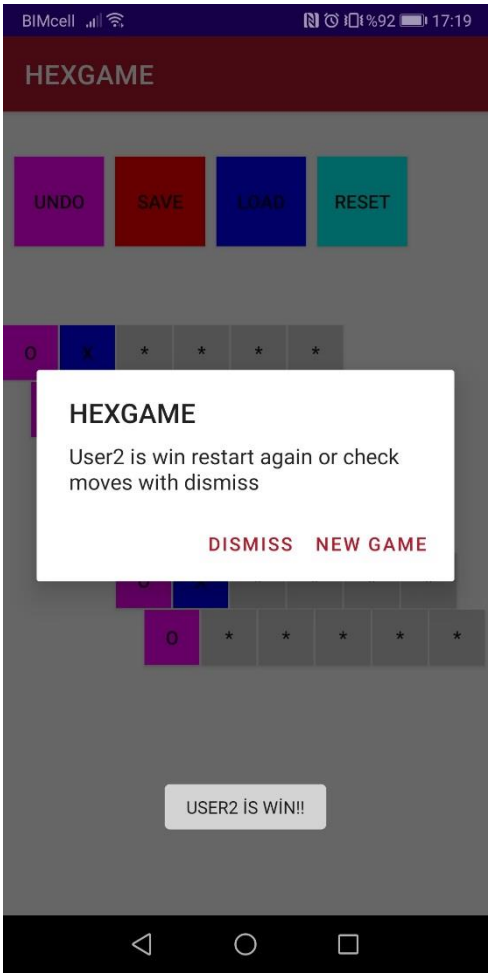


User 1 is win and X buttons color are change.

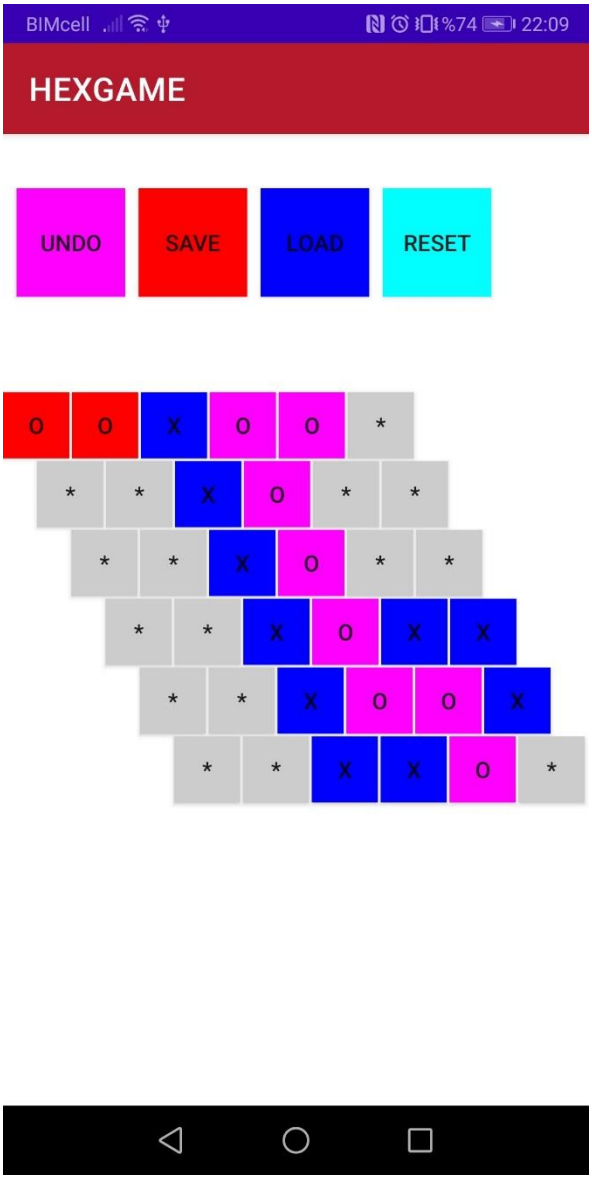


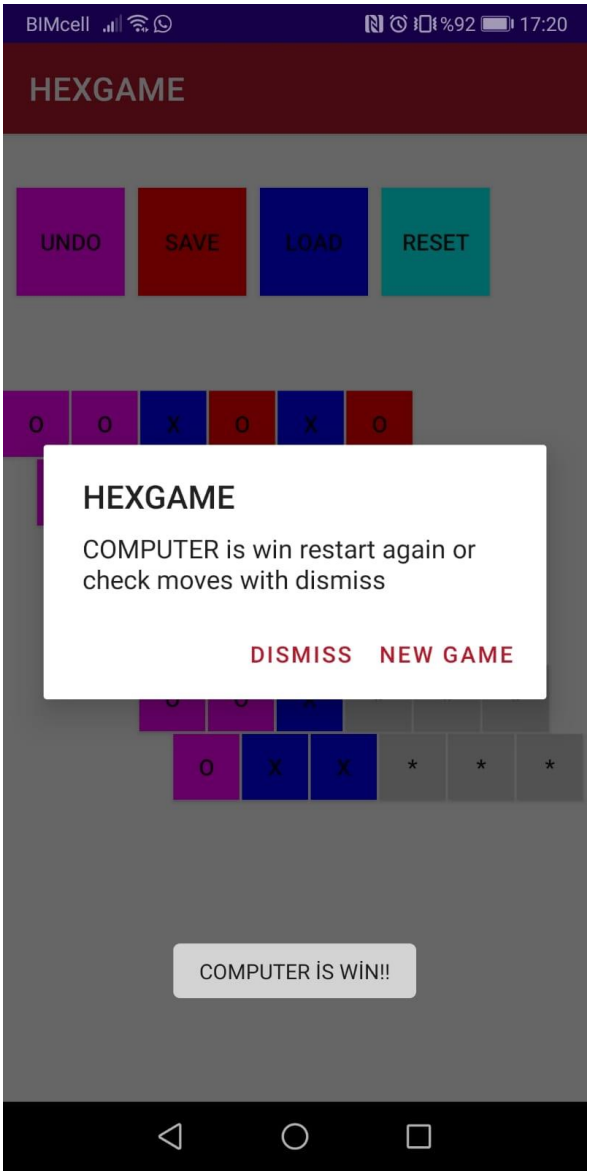
When user1 is win alert dialog is appears. İf you press new game game is starting again if you press dıssmıs button you can make move on the board you can undo reset save load.

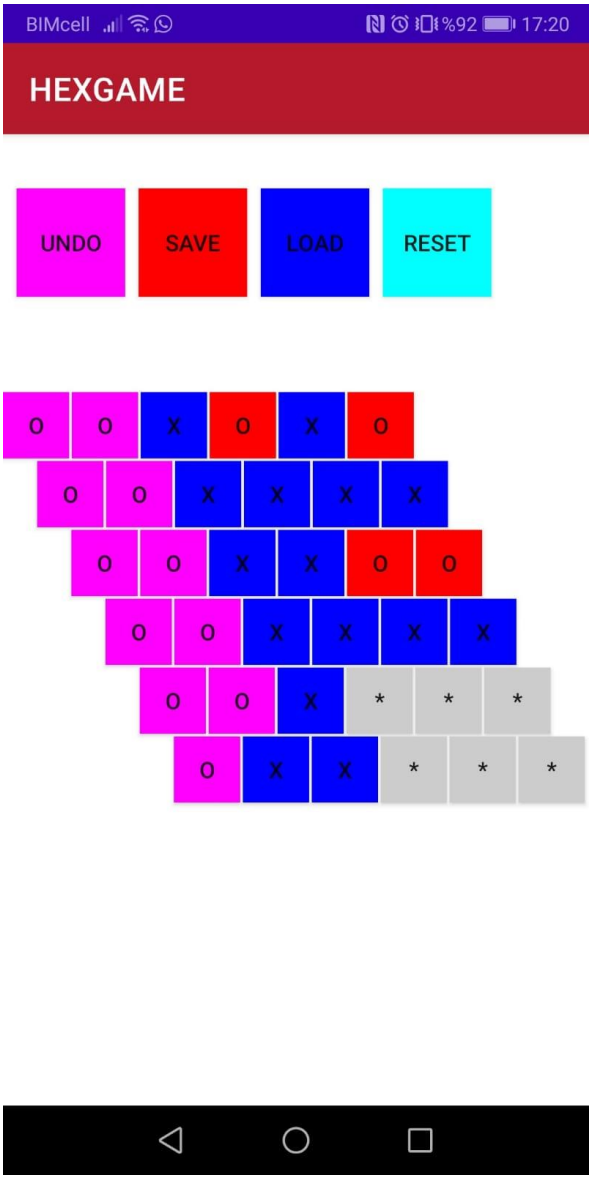


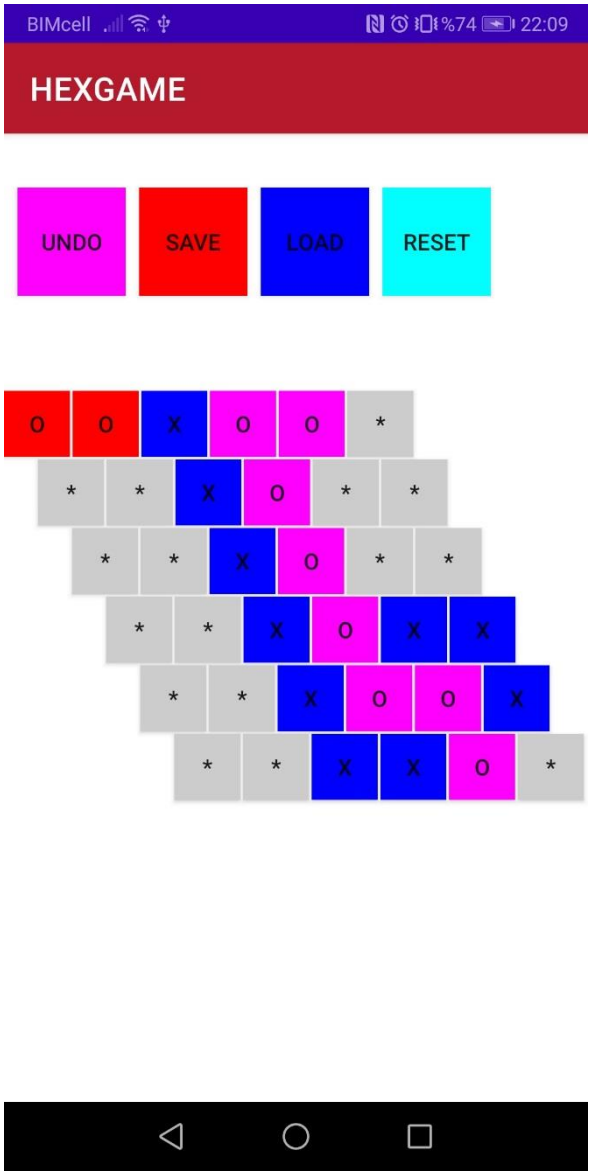


User 2 win.same situations with user 1 i explain it upper.


BIMcell
%74
22:09
HEXGAME
UNDO
SAVE
LOAD
RESET


BIMcell
%92
17:20
HEXGAME
UNDO
SAVE
LOAD
RESET
HEXGAME
COMPUTER is win restart again or check moves with dismiss
DISMISS
NEW GAME
COMPUTER İS WİN!!





SOME COMPUTER WINS SITUATION.