GIT Department of Computer Engineering
CSE 241 - Object Oriented Programming
Winter Project

Ahmet Tuğkan Ayhan
1901042692

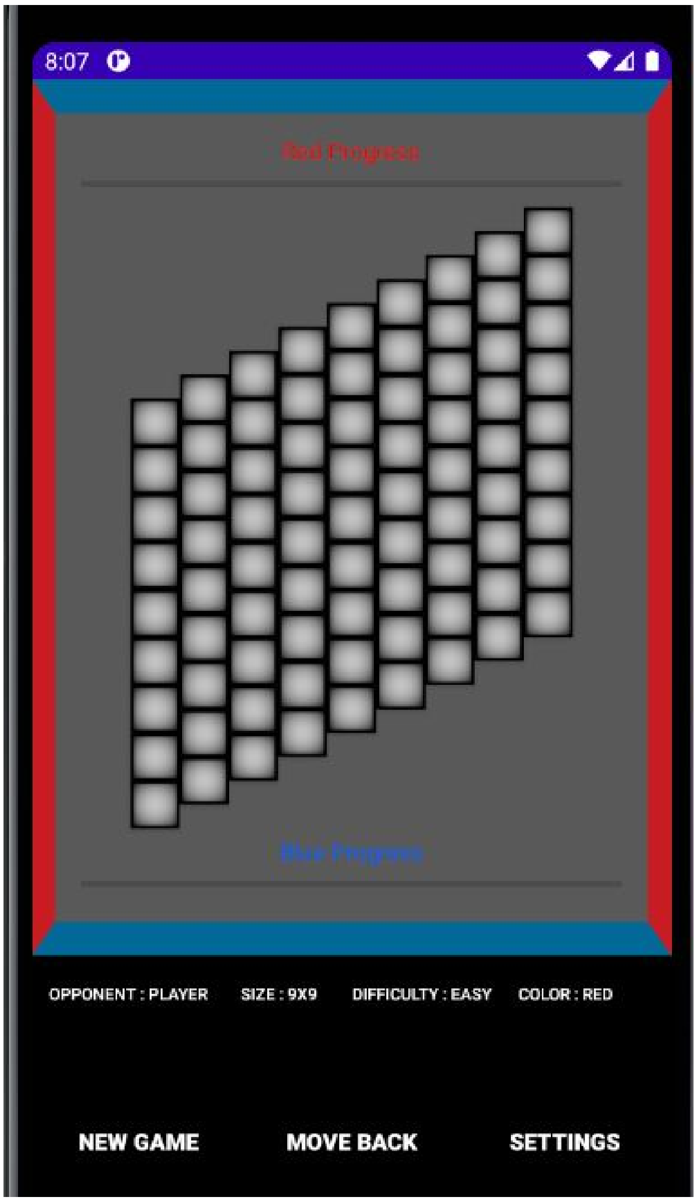
1) What is Hex Game ?

Hex game is a board game which 2 players try to defeat each other by blocking their opponent’s way. Hex usually has a square type board and board size can differ for each developer. And in this winter project I let user to choose 5 different board sizes. If one of the players manages to reach from one side to opposide side without breaking it’s chain then that player wins the game. Customizable settings:

- **Difficulty :** Easy, Nominal, Difficult, Master
- **Board Size :** 5x5, 6x6, 7x7, 8x8, 9x9
- **Opponent :** Player vs Player , Player vs Computer
- **Color :** Red / Blue

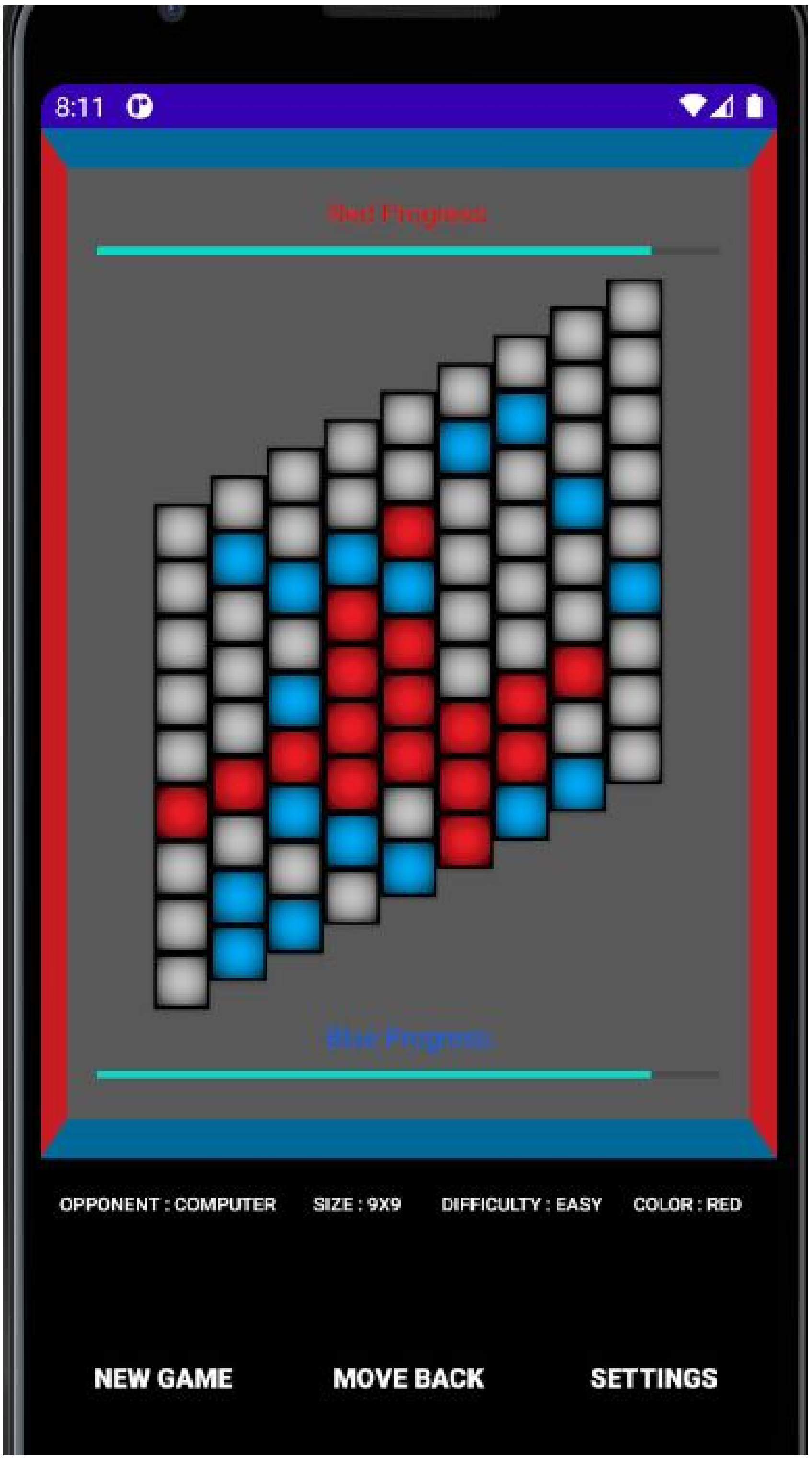
2) Screenshots

- **screen when game first opened**



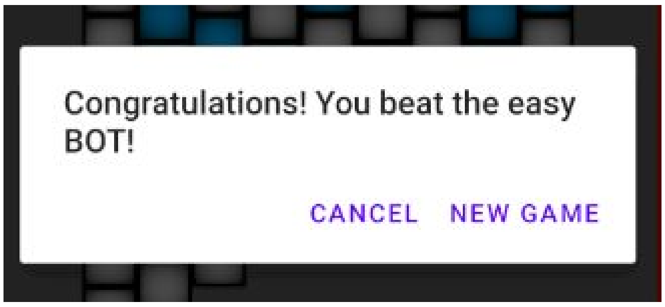
OPPONENT : PLAYER   SIZE : 9X9   DIFFICULTY : EASY   COLOR : RED

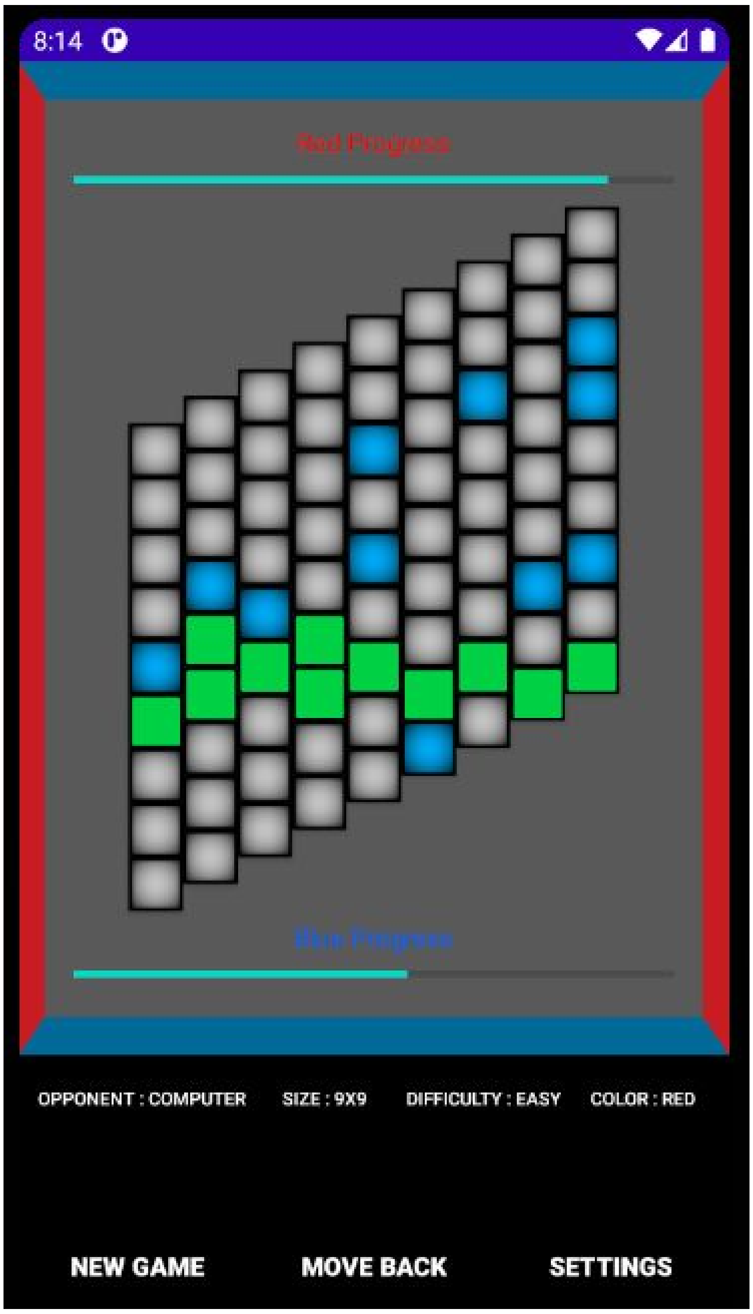
- **after both players make some move**



OPPONENT : COMPUTER SIZE : 9X9 DIFFICULTY : EASY COLOR : RED

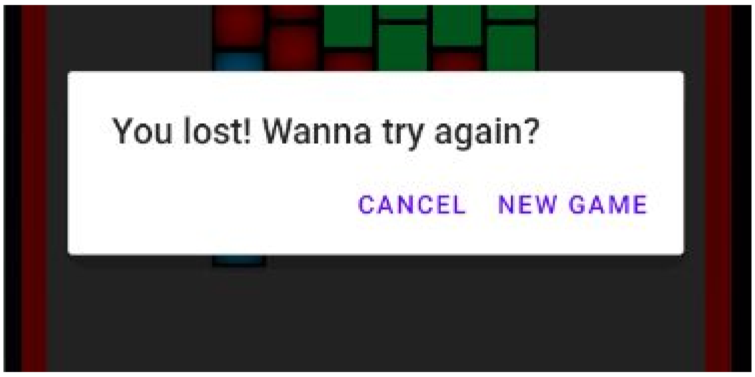
- **After Red wins the game**


Congratulations! You beat the easy BOT!



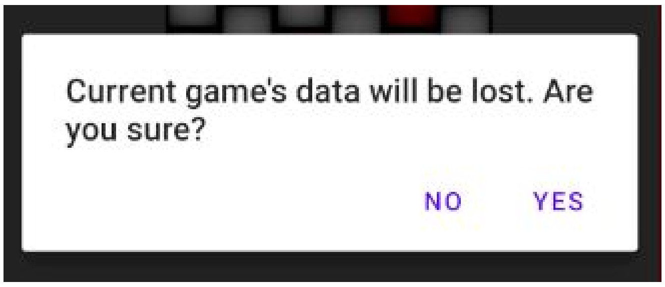
OPPONENT : COMPUTER SIZE : 9X9 DIFFICULTY : EASY COLOR : RED

- After Blue wins the game (against bot)



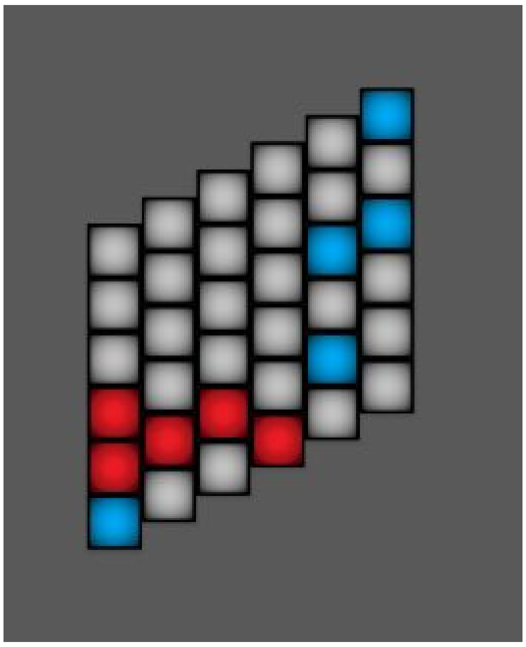
You lost! Wanna try again?

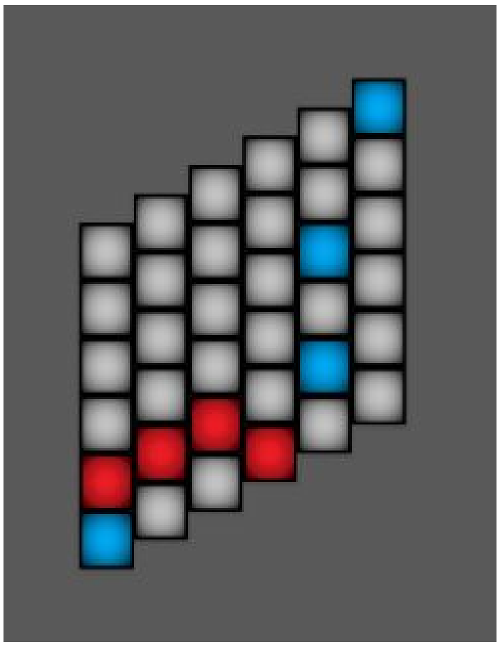
- New Game button

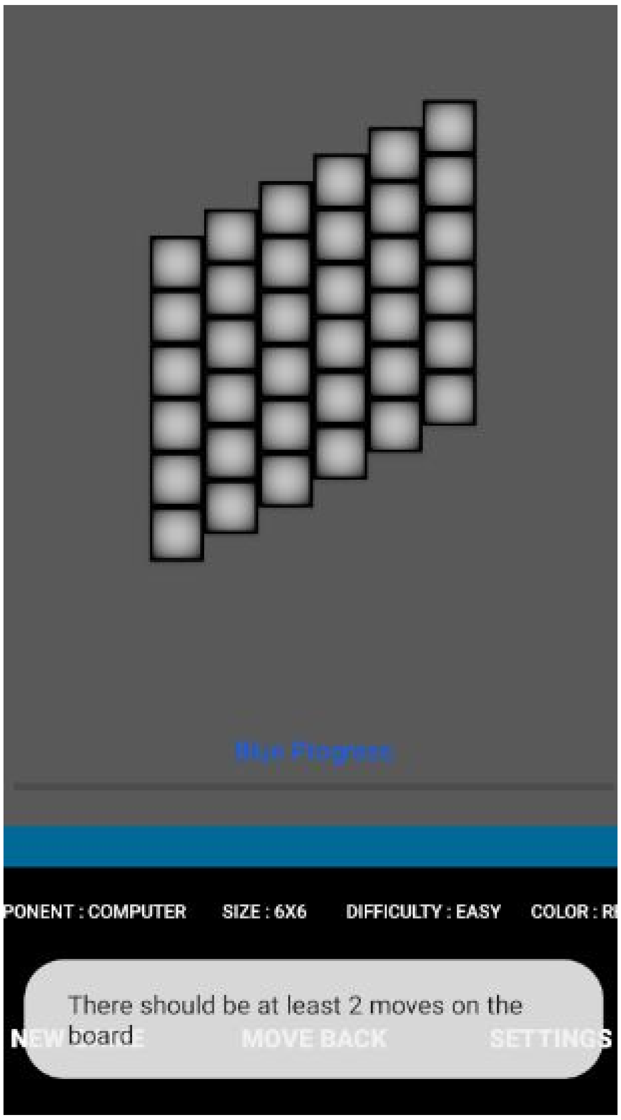


Current game's data will be lost. Are you sure?

- Move Back button



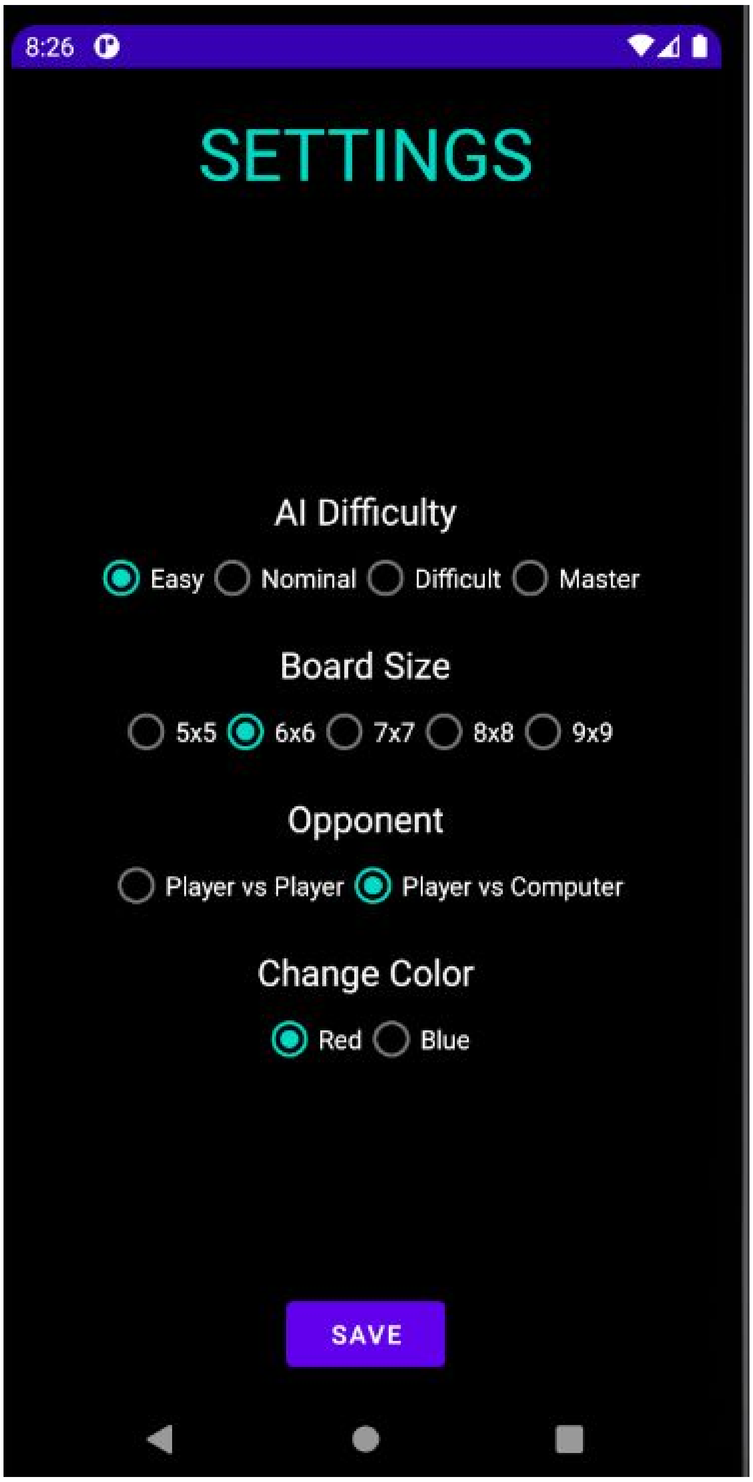




PONENT : COMPUTER SIZE : 6X6 DIFFICULTY : EASY COLOR : R

There should be at least 2 moves on the board

- **Settings Button**



8:26

SETTINGS

AI Difficulty

[✓ Easy] Nominal Difficult Master

Board Size

5x5 [✓ 6x6] 7x7 8x8 9x9

Opponent

Player vs Player [✓ Player vs Computer]

Change Color

[✓ Red] Blue

3) Pseudo Code

When program is executed:

```
startGame method executed
    set initial informations (size, color, opponent, etc.)
    allocate memory for board and other necessary fields
```

After clicked on a cell:

```
onClick method calls
    get id of the click
    if clicked id is equal to new game button
        call newGame method
    else if clicked id is equal to move back button
        call moveBack method
    else if clicked id is equal to settings button
        call settingActivity method
    else
        get clicked cell's name
        search every cell's name until finding clicked cell's
        after getting cell look for played cell is empty or not
        also look for which opponent user is playing against
        if computer and cell is empty
            play that cell and let ai make a move
        else if player and cell is empty
            play that cell
        else if game is over
            give a toast message that say's game is already over
        else
            give a toast message that say's this place is not empty
```

Win condition check:

```
after a player makes a move: call checkEnd method
check end method calls searchRoute method with first row or column’s index as a parameter
inside searchRoute method
if isPathCompleted returns true
    change cell’s into green (valid win condition cells)
    call a win dialogue menu for the user
    set isEnd variable as true
else if given parameter is first row or column and if game is not ended
    look for every column or row according to player
    if any cell is selected within first column or row
        set it’s coordinates as a possible route
        call this method recursively but increasing index by 1

else if given parameter is not first row or column and not passes max possible move and not ended game
    check for neighbors
    for every neighbor
        if any neighbors is same colors as user
            call this method with +1 of current parameter
        throw exception if neighbor is null
```

AI algorithms:

<u>easy:</u>

```
after user makes a move: call playAI method with “easy” parameter
playAI method executes AI_easy method inside a switch case
AI_easy method:
    set a seed for Random class
    until selected coordinates is not empty
        select x and y values randomly

    after valid x and y values are selected
    call play method with these coordinates
play:
    change color and tag of selected cell
```

```
assign played cell's coordinates for later use (moveBack method)
increase move count by one
check if game is over or not
change turn if it is not over
```

nominal:

```
if computer is playing blue
    for every row (from biggest to lowest)
        for every column (from biggest to lowest)
            if state of the cell is blue
                try
                    if selected row+1 , column is empty play
                    that cell and finish
                    else if selected row+1, column+1 is empty
                    play that cell and finish
                    else if selected row, column-1 is empty
                    play that cell and finish
                throw nullpointerexception if failed
    try to find empty first column and play that
else if computer is playing red
    for every column(from biggest to lowest)
        for every row(from biggest to lowest)
            if state of the cell is red
                try
                    if selected row, column+1 is empty play
                    that cell and finish
                    else if selected row-1, column+1 is empty
                    play that cell and finish
                throw nullpointerexception if failed
    try to find empty first row and play that
```

difficult:

not implemented

master:

not implemented