```
clc;
clear;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%INITIALIZING%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load Othello.mat;
side_selection='black';

switch side_selection
    case 'white'
        selected_color_disc=whitedisc;
        npc_disc=blackdisc;
        disc_color=0;
    case 'black'
        selected_color_disc=blackdisc;
        npc_disc=whitedisc;
        disc_color=1;
    otherwise
        fprintf('Please select a side.\n');
end

CURRENT_BOARD=[2,2,2,2,2,2,2,2;
               2,2,2,2,2,2,2,2;
               2,2,2,2,2,2,2,2;
               2,2,2,0,1,2,2,2;
               2,2,2,1,0,2,2,2;
               2,2,2,2,2,2,2,2;
               2,2,2,2,2,2,2,2;
               2,2,2,2,2,2,2,2;];
GAME_END=0;
imshow([Board{1,:};Board{2,:};Board{3,:};Board{4,:};Board{5,:};Board{6,:};Board{7,:};↙
Board{8,:}])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%GAMING_PROCESS%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

while(~GAME_END)
    %%%%%%%%%%%%%USER_TURN%%%%%%%%%%%%%
    [user_X,user_Y]=user_turn(CURRENT_BOARD);
    CURRENT_BOARD(user_Y,user_X)=disc_color;
    Board{user_Y,user_X}=selected_color_disc;
    imshow([Board{1,:};Board{2,:};Board{3,:};Board{4,:};Board{5,:};Board{6,:};Board↙
{7,:};Board{8,:}])%refresh play board
    [eat_row,eat_col]=eating_check(CURRENT_BOARD,user_Y,user_X);
    if eat_row<=8&&eat_col<=8
        CURRENT_BOARD(eat_row,eat_col)=~CURRENT_BOARD(eat_row,eat_col);
        if Board{eat_row,eat_col}==blackdisc
            Board{eat_row,eat_col}=whitedisc;
        else
            Board{eat_row,eat_col}=blackdisc;
        end
    end
```

```
    imshow([Board{1,:};Board{2,:};Board{3,:};Board{4,:};Board{5,:};Board{6,:};Board↙
{7,:};Board{8,:}])%refresh play board

    %%%%%%%%%%%%%NPC_TURN%%%%%%%%%%%%%
    [npc_X,npc_Y]=npc_check_eligibility(CURRENT_BOARD,user_X,user_Y);
    CURRENT_BOARD(npc_Y,npc_X)=~disc_color;
    Board{npc_Y,npc_X}=npc_disc;
    imshow([Board{1,:};Board{2,:};Board{3,:};Board{4,:};Board{5,:};Board{6,:};Board↙
{7,:};Board{8,:}])%refresh play board
%     [npc_eat_row,npc_eat_col]=eating_check(CURRENT_BOARD,npc_Y,npc_X);
%     if npc_eat_row<=8&&npc_eat_col<=8
%         CURRENT_BOARD(npc_eat_row,npc_eat_col)=~CURRENT_BOARD(npc_eat_row,↙
npc_eat_col);
%         if Board{npc_eat_row,npc_eat_col}==blackdisc
%             Board{npc_eat_row,npc_eat_col}=whitedisc;
%         else
%             Board{npc_eat_row,npc_eat_col}=blackdisc;
%         end
%     end
    imshow([Board{1,:};Board{2,:};Board{3,:};Board{4,:};Board{5,:};Board{6,:};Board↙
{7,:};Board{8,:}])%refresh play board

    %%%%%%%%%%%%%FINISH_CHECK%%%%%%%%%%%%%
    GAME_END=finish_check(CURRENT_BOARD);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%FUNCTION_DEFINATION%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [x,y]=user_pos()
[user_x,user_y]=ginput(1);
user_y=631-user_y;

if user_x>0&&user_x<=83.875
    x=1;
elseif user_x>83.875&&user_x<=167.75
    x=2;
elseif user_x>167.75&&user_x<=251.625
    x=3;
elseif user_x>251.625&&user_x<=335.5
    x=4;
elseif user_x>335.5&&user_x<=419.375
    x=5;
elseif user_x>419.375&&user_x<=503.25
    x=6;
elseif user_x>503.25&&user_x<=587.125
    x=7;
elseif user_x>587.125&&user_x<=671
    x=8;
end


if user_y>0&&user_y<=78.5
    y=1;
```

```
elseif user_y>78.5&&user_y<=157
    y=2;
elseif user_y>157&&user_y<=235.5
    y=3;
elseif user_y>235.5&&user_y<=314
    y=4;
elseif user_y>314&&user_y<=392.5
    y=5;
elseif user_y>392.5&&user_y<=471
    y=6;
elseif user_y>471&&user_y<=549.5
    y=7;
elseif user_y>549.5&&user_y<=628
    y=8;
end

y=9-y;

end


function [pos_X,pos_Y]=user_turn(board)
state=0;
while ~state
    [pos_X,pos_Y]=user_pos();
    state=check_valid(board,pos_Y,pos_X,1);
end
% board(user_Y,user_X)=1;
fprintf('Row:%d Col:%d\n',pos_Y,pos_X);
end

function state=check_valid(board,row,col,color)%white=0, black=1
state=0;%false
if row==1
    if col>=2&&col<=7
        if board(row,col-1)==~color||board(row,col+1)==~color||board(row-1,col)==~color
            state=1;
        end
    elseif col==1
        if board(row,col+1)==~color||board(row+1,col)==~color
            state=1;
        end
    elseif col==8
        if board(row+1,col)==~color||board(row,col-1)==~color
            state=1;
        end
    end

elseif row>=2&&row<=7
    if col==1
        if board(row+1,col)==~color||board(row-1,col)==~color||board(row,col+1)==~color
            state=1;
        end
    elseif col>=2&&col<=7
        if board(row+1,col)==~color||board(row-1,col)==~color||board(row,col+1)↙
```

```
==~color||board(row,col-1)==~color
            state=1;
        end
    elseif col==8
        if board(row+1,col)==~color||board(row-1,col)==~color||board(row,col-1)==~color
            state=1;
        end
    end
elseif row==8
    if col>=2&&col<=7
        if board(row,col+1)==~color||board(row,col-1)==~color||board(row-1,col)==~color
            state=1;
        end
    elseif col==1
        if board(row-1,col)==~color||board(row,col+1)==~color
            state=1;
        end
    elseif col==8
        if board(row-1,col)==~color||board(row,col-1)==~color
            state=1;
        end
    end
end
end

function [eat_row,eat_col]=eating_check(board,row,col)
if row<=2
    if col<=2%(0,0)
        if board(row,col)~=board(row+1,col)&&board(row+1,col)~=2
            if board(row,col)==board(row+2,col)
                eat_row=row+1;
                eat_col=col;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        elseif board(row,col)~=board(row,col+1)&&board(row,col+1)~=2
            if board(row,col)==board(row,col+2)
                eat_row=row;
                eat_col=col+1;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;

        end
    elseif col>=7
        if board(row,col)~=board(row+1,col)&&board(row+1,col)~=2
            if board(row,col)==board(row+2,col)
                eat_row=row+1;
                eat_col=col;
            else
```

```
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        elseif board(row,col)~=board(row,col-1)&&board(row,col-1)~=2
            if board(row,col)==board(row,col-2)
                eat_row=row;
                eat_col=col-1;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    else%(,2-6)
        if board(row,col)~=board(row+1,col)&&board(row+1,col)~=2
            if board(row,col)==board(row+2,col)
                eat_row=row+1;
                eat_col=col;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        elseif board(row,col)~=board(row,col+1)&&board(row,col+1)~=2
            if board(row,col)==board(row,col+2)
                eat_row=row;
                eat_col=col+1;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        elseif board(row,col)~=board(row,col-1)&&board(row,col-1)~=2
            if board(row,col)==board(row,col-2)
                eat_row=row;
                eat_col=col-1;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    end
elseif row>=7%bottom
    if col<=2
        if board(row,col)~=board(row-1,col)&&board(row-1,col)~=2
            if board(row,col)==board(row-2,col)
                eat_row=row-1;
                eat_col=col;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
```

```
    elseif board(row,col)~=board(row,col+1)&&board(row,col+1)~=2
        if board(row,col)==board(row,col+2)
            eat_row=row;
            eat_col=col+1;
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    else
        eat_row=999; %arbitary numbers if no eating happened
        eat_col=999;
    end

elseif col>=7
    if board(row,col)~=board(row-1,col)&&board(row-1,col)~=2
        if board(row,col)==board(row-2,col)
            eat_row=row-1;
            eat_col=col;
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    elseif board(row,col)~=board(row,col-1)&&board(row,col-1)~=2
        if board(row,col)==board(row,col-2)
            eat_row=row;
            eat_col=col-1;
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    else
        eat_row=999; %arbitary numbers if no eating happened
        eat_col=999;
    end
else
    if board(row,col)~=board(row-1,col)&&board(row-1,col)~=2
        if board(row,col)==board(row-2,col)
            eat_row=row-1;
            eat_col=col;
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    elseif board(row,col)~=board(row,col-1)&&board(row,col-1)~=2
        if board(row,col)==board(row,col-2)
            eat_row=row;
            eat_col=col-1;
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    elseif board(row,col)~=board(row,col+1)&&board(row,col+1)~=2
        if board(row,col)==board(row,col+2)
            eat_row=row;
            eat_col=col+1;
```

```
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    end
end

if col<=2
    if row>=3&&row<=6
        if board(row,col)~=board(row+1,col)&&board(row+1,col)~=2
            if board(row,col)==board(row+2,col)
                eat_row=row+1;
                eat_col=col;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        elseif board(row,col)~=board(row-1,col)&&board(row-1,col)~=2
            if board(row,col)==board(row-2,col)
                eat_row=row-1;
                eat_col=col;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        elseif board(row,col)~=board(row,col+1)&&board(row,col+1)~=2
            if board(row,col)==board(row,col+2)
                eat_row=row;
                eat_col=col+1;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    end

elseif col>=3&&col<=6
    if row>=3&&col<=6
        if board(row,col)~=board(row+1,col)&&board(row+1,col)~=2
            if board(row,col)==board(row+2,col)
                eat_row=row+1;
                eat_col=col;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        elseif board(row,col)~=board(row-1,col)&&board(row-1,col)~=2
            if board(row,col)==board(row-2,col)
```

```
                eat_row=row-1;
                eat_col=col;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        elseif board(row,col)~=board(row,col+1)&&board(row,col+1)~=2
            if board(row,col)==board(row,col+2)
                eat_row=row;
                eat_col=col+1;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        elseif board(row,col)~=board(row,col-1)&&board(row,col-1)~=2
            if board(row,col)==board(row,col-2)
                eat_row=row;
                eat_col=col-1;
            else
                eat_row=999; %arbitary numbers if no eating happened
                eat_col=999;
            end
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    end
elseif col>=7
    if row>=3&&col<=6
        if board(row,col)~=board(row+1,col)&&board(row+1,col)~=2
            if board(row,col)==board(row+2,col)
                eat_row=row+1;
                eat_col=col;
            end
        elseif board(row,col)~=board(row-1,col)&&board(row-1,col)~=2
            if board(row,col)==board(row-2,col)
                eat_row=row-1;
                eat_col=col;
            end
        elseif board(row,col)~=board(row,col-1)&&board(row,col-1)~=2
            if board(row,col)==board(row,col-2)
                eat_row=row;
                eat_col=col-1;
            end
        else
            eat_row=999; %arbitary numbers if no eating happened
            eat_col=999;
        end
    end
end

end

function state=finish_check(board)
k=0;
```

```
for i=1:1:8
    for j=1:1:8
        if board(i,j)~=2
            k=k+1;
        end
    end
end

if k==64
    state=1;
else
    state=0;
end
fprintf("There are totally %d disc on the board\n",k);
end



function [npc_x,npc_y]=npc_check_eligibility(board,user_x,user_y)
x=rand();
x=x*4;
y=round(x);
if y<1
    y=1;
end

switch y
    case 1
        if board(user_y+1,user_x)~=board(user_y,user_x)&&board(user_y+1,user_x)==2
            npc_x=user_x;
            npc_y=user_y+1;
        elseif board(user_y-1,user_x)~=board(user_y,user_x)&&board(user_y-1,user_x)==2
            npc_x=user_x;
            npc_y=user_y-1;
        elseif board(user_y,user_x+1)~=board(user_y,user_x)&&board(user_y,user_x+1)==2
            npc_x=user_x+1;
            npc_y=user_y;
        elseif board(user_y,user_x-1)~=board(user_y,user_x)&&board(user_y,user_x-1)==2
            npc_x=user_x-1;
            npc_y=user_y;
        end
    case 2
        if board(user_y-1,user_x)~=board(user_y,user_x)&&board(user_y-1,user_x)==2
            npc_x=user_x;
            npc_y=user_y-1;
        elseif board(user_y+1,user_x)~=board(user_y,user_x)&&board(user_y+1,user_x)==2
            npc_x=user_x;
            npc_y=user_y+1;
        elseif board(user_y,user_x+1)~=board(user_y,user_x)&&board(user_y,user_x+1)==2
            npc_x=user_x+1;
            npc_y=user_y;
        elseif board(user_y,user_x-1)~=board(user_y,user_x)&&board(user_y,user_x-1)==2
            npc_x=user_x-1;
            npc_y=user_y;
        end
```

```
    case 3
        if board(user_y,user_x+1)~=board(user_y,user_x)&&board(user_y,user_x+1)==2
            npc_x=user_x+1;
            npc_y=user_y;
        elseif board(user_y-1,user_x)~=board(user_y,user_x)&&board(user_y-1,user_x)==2
            npc_x=user_x;
            npc_y=user_y-1;
        elseif board(user_y+1,user_x)~=board(user_y,user_x)&&board(user_y+1,user_x)==2
            npc_x=user_x;
            npc_y=user_y+1;
        elseif board(user_y,user_x-1)~=board(user_y,user_x)&&board(user_y,user_x-1)==2
            npc_x=user_x-1;
            npc_y=user_y;
        end
    case 4
        if board(user_y,user_x-1)~=board(user_y,user_x)&&board(user_y,user_x-1)==2
            npc_x=user_x-1;
            npc_y=user_y;
        elseif board(user_y-1,user_x)~=board(user_y,user_x)&&board(user_y-1,user_x)==2
            npc_x=user_x;
            npc_y=user_y-1;
        elseif board(user_y-1,user_x)~=board(user_y,user_x)&&board(user_y-1,user_x)==2
            npc_x=user_x;
            npc_y=user_y-1;
        elseif board(user_y,user_x+1)~=board(user_y,user_x)&&board(user_y,user_x+1)==2
            npc_x=user_x+1;
            npc_y=user_y;
        end
end
end
```