https://www.khanacademy.org/computer-programming/connect4/5649496171036672

- If we somehow deconstruct the tic tac toe game and make it 4 across instead of 3 across.
- Use the tile prototype functions to make the checkwins easier
- Need to figure out how to make the function within the array to display the red and yellow chips
- Redesign game board to how it was designed in the original project idea and add the splash screen and stuff
- Change all of the checkwins

https://www.khanacademy.org/computer-programming/c4/5280386278965248

- Example without the checkwin

```
var redChip = function(xPos, yPos)
//{ mouseClicked = function()
{ noStroke();
fill(255, 0, 0);
ellipse(xPos, yPos, 45, 45);
//};
};
var yellowChip = function(xPos, yPos)
//{ mouseClicked = function()
{ noStroke();
fill(255, 242, 0);
ellipse(xPos, yPos, 45, 45);
//};
};
var playerTurn = 0;
var NUM_COLS= 6;
var NUM_ROWS= 7;
var SYMBOLS= [redChip, yellowChip]; //somehow make this work !!!!!
var currentScene =0; //0=splash scene, 1=game scene

var Button = function(config) {
   this.x = config.x || 0;
   this.y = config.y || 0;
   this.width = config.width || 150;
   this.height = config.height || 50;
   this.label = config.label || "Click";
   this.onClick = config.onClick || function() {};
};

Button.prototype.draw = function() {
   fill(0, 234, 255);
   rect(this.x, this.y, this.width, this.height, 5);
   fill(0, 0, 0);
   textSize(19);
   textAlign(LEFT, TOP);
   text(this.label, this.x+10, this.y+this.height/4);
};

Button.prototype.isMouseInside = function() {
   return mouseX > this.x &&
          mouseX < (this.x + this.width) &&
          mouseY > this.y &&
          mouseY < (this.y + this.height);
};

Button.prototype.handleMouseClick = function() {
   if (this.isMouseInside()) {
       this.onClick();
   }
};

var btn1 = new Button({ //this is how to make new button
   x: 137,
   y: 310,
   label: "START GAME!!!",
   onClick: function() {
       currentScene = 1;
   }
});


var splash = function()
{
   fill(255, 0, 0);
   rect(0,0,400,400);
   fill(0, 0, 0);
   textSize(40);
   text("Connect 4", 102,38);
   textSize(15);
   text("c", 127,91);
   text("Instructions: ", 107, 247);

   btn1.draw();
};
```

```
var tiles = []; //an array of the objects

/* check for winner, determine if 4 in a row - boolean */
/*  0 8  17 26
    1 9  18 27
    2 10 19 28
    3 11 20 29
    4 12 21 30
    5 13 22 31
    6 14 23 32
    7 15 24 33
    5 16 25 34
    */
 var checkWin = function() {
   /* if (tiles[0].label === tiles[4].label && tiles[4].label === tiles[8].label && !tiles[8].empty())
//check left to right diagonal
    {
    return true;
    }
    if (tiles[0].label === tiles[3].label && tiles[3].label === tiles[6].label && !tiles[6].empty()) //top
row
    {
        return true;
    }
    if (tiles[0].label === tiles[1].label && tiles[1].label === tiles[2].label && !tiles[2].empty()) //left
column
    {
        return true;
    }
    if (tiles[3].label === tiles[4].label && tiles[4].label === tiles[5].label && !tiles[5].empty())
//middle column
    {
        return true;
    }
    if (tiles[6].label === tiles[7].label && tiles[7].label === tiles[8].label && !tiles[8].empty()) //right
column row
    {
        return true;
    }
    if (tiles[1].label === tiles[4].label && tiles[4].label === tiles[7].label && !tiles[7].empty())
//middle row
    {
        return true;
    }
    if (tiles[2].label === tiles[5].label && tiles[5].label === tiles[8].label && !tiles[8].empty())
//bottom row
    {
        return true;
    }
    if (tiles[6].label === tiles[4].label && tiles[4].label === tiles[2].label && !tiles[2].empty()) //right
to left diagnoal
    {
        return true;
    }
    return false;
    */
};

var Tile = function(x, y) {
    this.x = x;
    this.y = y;
    this.size = width/NUM_COLS;
    this.label = "";
};

Tile.prototype.draw = function() {
    fill(214, 247, 202);
    strokeWeight(2);
    rect(this.x, this.y, this.size, this.size, 10);
    //textSize(10);
    textAlign(CENTER, CENTER);
    fill(0, 0, 0);
    text(this.label, this.x+this.size/2, this.y+this.size/2);
};

Tile.prototype.empty = function() {
    return this.label === "";
};
/* Onclick metod of the tile object class */
Tile.prototype.onClick = function() {
    // If the tile is not empty, exit the function
    if (!this.empty()) {
        return;
    }
    // Put the player's symbol on the tile
    this.label = SYMBOLS[playerTurn];
    // Change the turn
    playerTurn++;
    if (playerTurn >= SYMBOLS.length) {
        playerTurn = 0;
    }
};
/* handleMouseVlicked method of tile object class */
Tile.prototype.handleMouseClick = function(x, y) {
    // Check for mouse clicks inside the tile
    if (x >= this.x && x<= this.x + this.size && y>= this.y && y<= this.y + this.size)
    {
        this.onClick();
    }
};
/*Creates an array of tile objects */
for (var i = 0; i < NUM_COLS; i++) {
    for (var j = 0; j < NUM_ROWS; j++) {
        tiles.push(new Tile(i * (width/NUM_COLS-1), j * (height/NUM_ROWS-1)));
    }
}

var drawTiles = function() {
    for (var i in tiles) {
        tiles[i].draw();
    }
};
/* check each tile to see if the user clicked in that tile */
mouseReleased = function() {
    if (currentScene === 0)
    {
        btn1.handleMouseClick();
    }
    else
    {
    for (var i in tiles) {
        tiles[i].handleMouseClick(mouseX, mouseY);
    }
    }
    if (checkWin())
    {
        if (playerTurn === 0)
        { println ("Red wins!"); }
        else
        { println("Yellow wins!");}
    }
};

draw = function() {
    if (currentScene ===0)
    {
        splash();
    }
    else
    {
        background(143, 143, 143);
        drawTiles();
    }

};
```