ASSIGNMENT1

Question1:Execute the program ChessBoard using html and css.

PROGRAM

chess.html

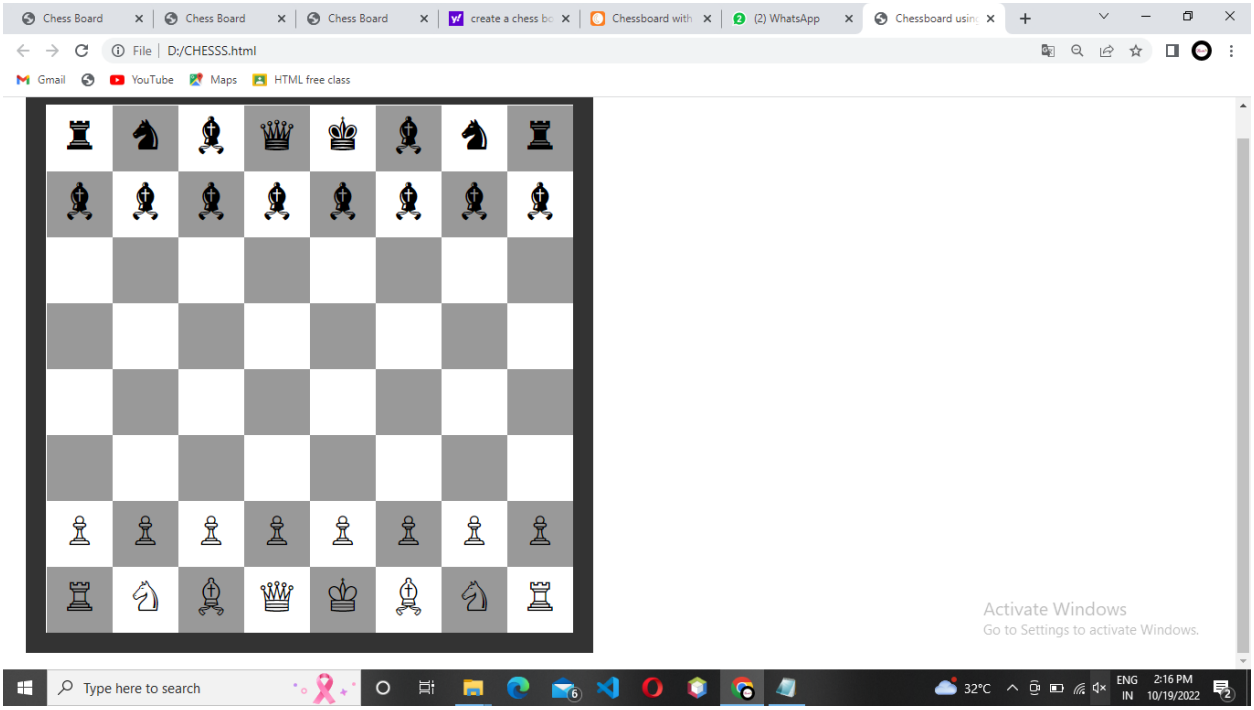
```
<html>

<head>
<meta charset="UTF-8">
<title>Chessboard using Pure CSS and HTML</title>

<style type="text/css">

.chessboard {
   width: 640px;
   height: 640px;
   margin: 20px;
   border: 25px solid #333;
}
.black {
   float: left;
   width: 80px;
   height: 80px;
   background-color: #999;
     font-size:50px;
   text-align:center;
   display: table-cell;
   vertical-align:middle;
}
.white {
   float: left;
   width: 80px;
   height: 80px;
   background-color: #fff;
   font-size:50px;
```

```
    text-align:center;
    display: table-cell;
    vertical-align:middle;
}

</style>

</head>

<body>

<div class="chessboard">
<!-- 1st -->
<div class="white">&#9820;</div>
<div class="black">&#9822;</div>
<div class="white">&#9821;</div>
<div class="black">&#9819;</div>
<div class="white">&#9818;</div>
<div class="black">&#9821;</div>
<div class="white">&#9822;</div>
<div class="black">&#9820;</div>
<!-- 2nd -->
<div class="black">&#9821;</div>
<div class="white">&#9821;</div>
<div class="black">&#9821;</div>
<div class="white">&#9821;</div>
<div class="black">&#9821;</div>
<div class="white">&#9821;</div>
<div class="black">&#9821;</div>
<div class="white">&#9821;</div>
<!-- 3th -->
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
```

```
<div class="white"></div>
<div class="black"></div>
<!-- 4st -->
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<!-- 5th -->
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<!-- 6th -->
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<div class="black"></div>
<div class="white"></div>
<!-- 7th -->
<div class="white">&#9817;</div>
<div class="black">&#9817;</div>
<div class="white">&#9817;</div>
<div class="black">&#9817;</div>
<div class="white">&#9817;</div>
<div class="black">&#9817;</div>
<div class="white">&#9817;</div>
```

```
<div class="black">&#9817;</div>
<!-- 8th -->
<div class="black">&#9814;</div>
<div class="white">&#9816;</div>
<div class="black">&#9815;</div>
<div class="white">&#9813;</div>
<div class="black">&#9812;</div>
<div class="white">&#9815;</div>
<div class="black">&#9816;</div>
<div class="white">&#9814;</div>
</div>
</body>
</html>
```

OUTPUT:



QUESTION2: Execute the program TIC TAC TOE

PROGRAM

tic tac toe.py:

```
# Tic Tac Toe game with GUI
# using tkinter

# importing all necessary libraries
```

```
import random
import tkinter
from tkinter import *
from functools import partial
from tkinter import messagebox
from copy import deepcopy

# sign variable to decide the turn of which player
sign = 0

# Creates an empty board
global board
board = [[" " for x in range(3)] for y in range(3)]

# Check l(O/X) won the match or not
# according to the rules of the game
def winner(b, l):
	return ((b[0][0] == l and b[0][1] == l and b[0][2] == l) or
				(b[1][0] == l and b[1][1] == l and b[1][2] == l) or
				(b[2][0] == l and b[2][1] == l and b[2][2] == l) or
				(b[0][0] == l and b[1][0] == l and b[2][0] == l) or
				(b[0][1] == l and b[1][1] == l and b[2][1] == l) or
				(b[0][2] == l and b[1][2] == l and b[2][2] == l) or
				(b[0][0] == l and b[1][1] == l and b[2][2] == l) or
				(b[0][2] == l and b[1][1] == l and b[2][0] == l))

# Configure text on button while playing with another player
def get_text(i, j, gb, l1, l2):
	global sign
	if board[i][j] == ' ':
		if sign % 2 == 0:
			l1.config(state=DISABLED)
			l2.config(state=ACTIVE)
			board[i][j] = "X"
		else:
			l2.config(state=DISABLED)
			l1.config(state=ACTIVE)
			board[i][j] = "O"
		sign += 1
		button[i][j].config(text=board[i][j])
	if winner(board, "X"):
		gb.destroy()
		box = messagebox.showinfo("Winner", "Player 1 won the match")
	elif winner(board, "O"):
		gb.destroy()
		box = messagebox.showinfo("Winner", "Player 2 won the match")
	elif(isfull()):
		gb.destroy()
```

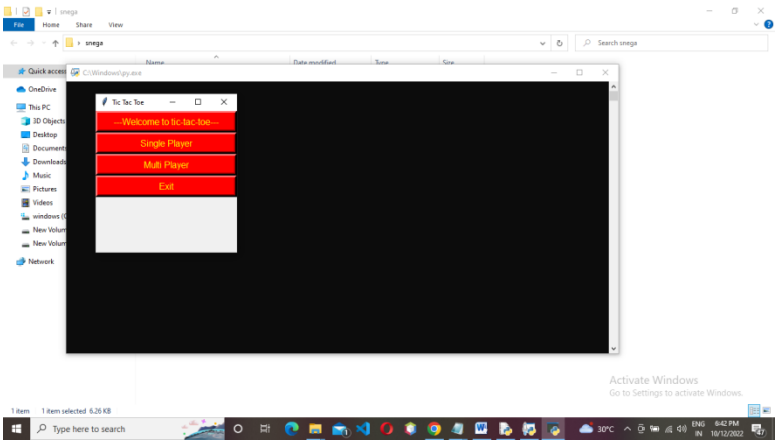
```
                box = messagebox.showinfo("Tie Game", "Tie Game")

# Check if the player can push the button or not
def isfree(i, j):
        return board[i][j] == " "

# Check the board is full or not
def isfull():
        flag = True
        for i in board:
                if(i.count(' ') > 0):
                        flag = False
        return flag

# Create the GUI of game board for play along with another player
def gameboard_pl(game_board, l1, l2):
        global button
        button = []
        for i in range(3):
                m = 3+i
                button.append(i)
                button[i] = []
                for j in range(3):
                        n = j
                        button[i].append(j)
                        get_t = partial(get_text, i, j, game_board, l1, l2)
                        button[i][j] = Button(
                                game_board, bd=5, command=get_t, height=4, width=8)
                        button[i][j].grid(row=m, column=n)
        game_board.mainloop()

# Decide the next move of system
def pc():
        possiblemove = []
        for i in range(len(board)):
                for j in range(len(board[i])):
                        if board[i][j] == ' ':
                                possiblemove.append([i, j])
        move = []
        if possiblemove == []:
                return
        else:
                for let in ['O', 'X']:
                        for i in possiblemove:
                                boardcopy = deepcopy(board)
                                boardcopy[i[0]][i[1]] = let
                                if winner(boardcopy, let):
                                        return i
```

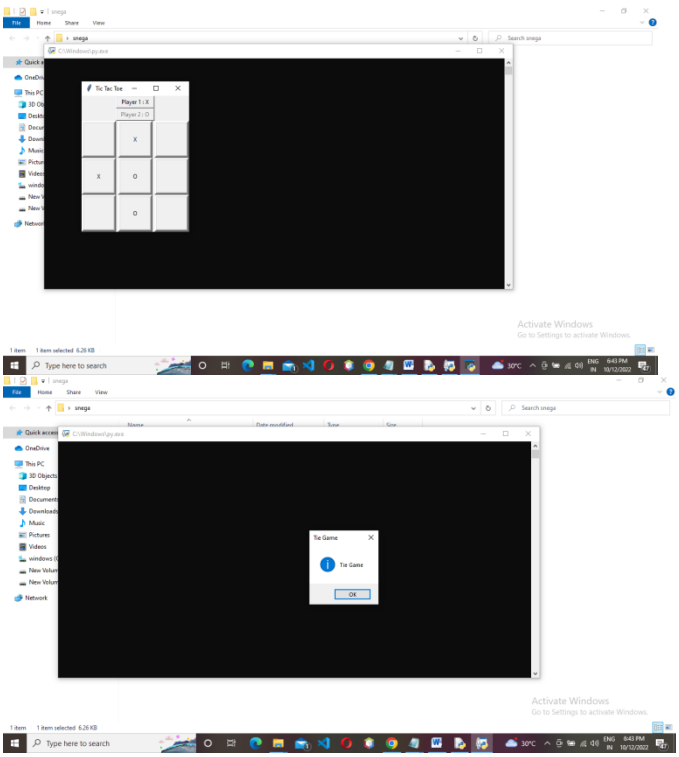
```
                corner = []
                for i in possiblemove:
                        if i in [[0, 0], [0, 2], [2, 0], [2, 2]]:
                                corner.append(i)
                if len(corner) > 0:
                        move = random.randint(0, len(corner)-1)
                        return corner[move]
                edge = []
                for i in possiblemove:
                        if i in [[0, 1], [1, 0], [1, 2], [2, 1]]:
                                edge.append(i)
                if len(edge) > 0:
                        move = random.randint(0, len(edge)-1)
                        return edge[move]

# Configure text on button while playing with system
def get_text_pc(i, j, gb, l1, l2):
        global sign
        if board[i][j] == ' ':
                if sign % 2 == 0:
                        l1.config(state=DISABLED)
                        l2.config(state=ACTIVE)
                        board[i][j] = "X"
                else:
                        button[i][j].config(state=ACTIVE)
                        l2.config(state=DISABLED)
                        l1.config(state=ACTIVE)
                        board[i][j] = "O"
                sign += 1
                button[i][j].config(text=board[i][j])
        x = True
        if winner(board, "X"):
                gb.destroy()
                x = False
                box = messagebox.showinfo("Winner", "Player won the match")
        elif winner(board, "O"):
                gb.destroy()
                x = False
                box = messagebox.showinfo("Winner", "Computer won the match")
        elif(isfull()):
                gb.destroy()
                x = False
                box = messagebox.showinfo("Tie Game", "Tie Game")
        if(x):
                if sign % 2 != 0:
                        move = pc()
                        button[move[0]][move[1]].config(state=DISABLED)
                        get_text_pc(move[0], move[1], gb, l1, l2)
```

```
# Create the GUI of game board for play along with system
def gameboard_pc(game_board, l1, l2):
	global button
	button = []
	for i in range(3):
		m = 3+i
		button.append(i)
		button[i] = []
		for j in range(3):
			n = j
			button[i].append(j)
			get_t = partial(get_text_pc, i, j, game_board, l1, l2)
			button[i][j] = Button(
				game_board, bd=5, command=get_t, height=4, width=8)
			button[i][j].grid(row=m, column=n)
	game_board.mainloop()

# Initialize the game board to play with system
def withpc(game_board):
	game_board.destroy()
	game_board = Tk()
	game_board.title("Tic Tac Toe")
	l1 = Button(game_board, text="Player : X", width=10)
	l1.grid(row=1, column=1)
	l2 = Button(game_board, text = "Computer : O",
				width = 10, state = DISABLED)

	l2.grid(row = 2, column = 1)
	gameboard_pc(game_board, l1, l2)

# Initialize the game board to play with another player
def withplayer(game_board):
	game_board.destroy()
	game_board = Tk()
	game_board.title("Tic Tac Toe")
	l1 = Button(game_board, text = "Player 1 : X", width = 10)

	l1.grid(row = 1, column = 1)
	l2 = Button(game_board, text = "Player 2 : O",
				width = 10, state = DISABLED)

	l2.grid(row = 2, column = 1)
	gameboard_pl(game_board, l1, l2)

# main function
def play():
	menu = Tk()
```

```
        menu.geometry("250x250")
        menu.title("Tic Tac Toe")
        wpc = partial(withpc, menu)
        wpl = partial(withplayer, menu)

        head = Button(menu, text = "---Welcome to tic-tac-toe---",
                                activeforeground = 'red',
                                activebackground = "yellow", bg = "red",
                                fg = "yellow", width = 500, font = 'summer', bd = 5)

        B1 = Button(menu, text = "Single Player", command = wpc,
                                activeforeground = 'red',
                                activebackground = "yellow", bg = "red",
                                fg = "yellow", width = 500, font = 'summer', bd = 5)

        B2 = Button(menu, text = "Multi Player", command = wpl, activeforeground = 'red',
                                activebackground = "yellow", bg = "red", fg = "yellow",
                                width = 500, font = 'summer', bd = 5)

        B3 = Button(menu, text = "Exit", command = menu.quit, activeforeground = 'red',
                                activebackground = "yellow", bg = "red", fg = "yellow",
                                width = 500, font = 'summer', bd = 5)
        head.pack(side = 'top')
        B1.pack(side = 'top')
        B2.pack(side = 'top')
        B3.pack(side = 'top')
        menu.mainloop()

# Call main function
if __name__ == '__main__':
        play()
```

OUTPUT:





QUESTION3: Registration form using html and css

PROGRAM:

Index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>Webpage Design</title>
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <div class="main">
      <div class="navbar">
         <div class="icon">
            <h2 class="logo">PraRoz</h2>
         </div>

         <div class="menu">
            <ul>
               <li><a href="#">HOME</a></li>
               <li><a href="#">ABOUT</a></li>
               <li><a href="#">SERVICE</a></li>
               <li><a href="#">DESIGN</a></li>
               <li><a href="#">CONTACT</a></li>
            </ul>
         </div>
```

```
        <div class="search">
          <input class="srch" type="search" name="" placeholder="Type to text">
          <a href="#"> <button class="btn">Search</button></a>
        </div>
      </div>
      <div class="content">
        <h1>Web Design & <br><span>Development</span><br>Course</h1>
        <p class="par">Lorem ipsum dolor sit amet consectetur adipisicing elit.Sunt neque
          expedita atque eveniet <br> quis nesciunt.Quos nulla vera consequnntur,fugit
          <br>a quae totam ipsa illum minus laudantium?</p>

          <button class="cn"><a href="#">JOIN US</a></button>

          <div class="form">
            <h2>Login Here</h2>
            <input type="email" name="email" placeholder="Enter Email Here">
            <input type="password" name="" placeholder="Enter Password Here">
            <button class="btnn"><a href="#">Login</a></button>

            <p class="link">Don't have an account<br>
            <a href="#">Sign up</a>here</a></p>
            <p class="liw">Login with</p>

            <div class="icon">
              <a href="#"><ion-icon name="logo-facebook"></ion-icon></a>
              <a href="#"><ion-icon name="logo-instagram"></ion-icon></a>
              <a href="#"><ion-icon name="logo-twitter"></ion-icon></a>
              <a href="#"><ion-icon name="logo-skype"></ion-icon></a>
              <a href="#"><ion-icon name="logo-google"></ion-icon></a>
            </div>

          </div>
            </div>
          </div>
      </div>
    </div>
    <script src="https://unpkg.com/ionicons@5.4.0/dist/ionicons.js"></script>
</body>
</html>
```

Style.css

```
*{
    margin: 0;
    padding: 0;

}

.main{
    width: 100;
```

```
    background: linear-gradient(to top, rgba(0,0,0,0.5)50%,rgba(0,0,0,0.5)50%), url(1.jpeg);
    background-position: center;
    background-size: cover;
    height: 109vh;
}

.navbar{
    width: 1200px;
    height: 75px;
    margin: auto;
}

.icon{
    width: 200px;
    float: left;
    height: 70px;
}

.logo{
    color:darkgrey;
    font-size: 35px;
    font-family: Arial;
    padding-left: 20px;
    float: left;
    padding-top: 10px;
}

.menu{
    width: 400px;
    float: left;
    height: 70px;
}

ul{
    float: left;
    display: flex;
    justify-content: center;
    align-items: center;
}

ul li{
    list-style: none;
    margin-left: 62px;
    margin-top: 27px;
    font-size: 14px;
}

ul li a{
```

```
    text-decoration: none;
    color:darkgoldenrod;
    font-family: Arial;
    font-weight: bold;
    transition: 0.4s ease-in-out;
}

ul li a:hover{
    color:floralwhite;
}

.search{
    width: 330px;
    float: left;
    margin-left: 270px;
}

.srch{
    font-family: 'Times New Roman';
    width: 200px;
    height: 40px;
    background: transparent;
    border: 1px solid#ff7200;
    margin-top: 13px;
    color: white;
    border-right: none;
    font-size: 16px;
    float: left;
    padding: 10px;
    border-bottom-left-radius: 5px;
    border-top-left-radius: 5px;
}

.btn{
    width: 100px;
    height: 40px;
    background:dimgray;
    border: 2px solid dimgray ;
    margin-top: 13px;
    color: aliceblue;
    font-size: 15px;
    border-bottom-right-radius: 5px;
    border-top-right-radius: 5px;
}

.btn:focus{
    outline: none;
}
```

```
.srch:focus{
    outline: none;
}

.content{
    width: 1200px;
    height: auto;
    margin: auto;
    color: seashell;
    position: relative;
}

.content .par{
    padding-left: 20px;
    padding-bottom: 25px;
    font-family: Arial;
    letter-spacing: 1.2px;
    line-height: 30px;
}

.content h1{
    font-family: 'Times New Roman';
    font-size: 50px;
    padding-left: 20px;
    margin-top: 9%;
    letter-spacing: 2px;
}

.content .cn{
    width: 160px;
    height: 40px;
    background:mediumseagreen;
    border: none;
    margin-bottom: 10px;
    margin-left: 20px;
    font-size: 18px;
    border-radius: 10px;
    cursor: pointer;
    transition: .4s ease;
}

.content .cn a{
    text-decoration: none;
    color: black;
    transition: .3s ease;
}
```

```
.cn:hover{
   background-color: white;
}

.content span{
   color: seagreen;
   font-size: 60px;
}

.form{
   width: 250px;
   height: 380px;
   background: linear-gradient(to top, rgba(0,0,0,0.8)50%,rgba(0,0,0,0.8)50%);
   position: absolute;
   top: -20px;
   left: 870px;
   border-radius: 10px;
   padding: 25px;
}

.form h2{
   width: 220px;
   font-family: sans-serif;
   text-align: center;
   color: chocolate;
   font-size: 22px;
   background-color: cornsilk;
   border-radius: 10px;
   margin: 2px;
   padding: 8px;
}

.form input{
   width: 240px;
   height: 35px;
   background: transparent;
   border-bottom: 1px solid chocolate;
   border-top: none;
   border-right: none;
   border-left: none;
   color: white;
   font-size: 15px;
   letter-spacing: 1px;
   margin-top: 30px;
   font-family: sans-serif;
}

.form input:focus{
```

```
    outline: none;
}

::placeholder{
    color: white;
    font-family: Arial;
}

.btnn{
    width: 240px;
    height: 40px;
    background: seagreen;
    border: none;
    margin-top: 30px;
    font-size: 18px;
    border-radius: 10px;
    cursor: pointer;
    color: #fff;
    transition: 0.4s ease;
}

.btnn:hover{
    background: #fff;
    color: tan;
}

.btnn a{
    text-decoration: none;
    color: black;
    font-weight: bold;
}

.form .link{
    font-family: Arial;
    font-size: 17px;
    padding-top: 20px;
    text-align: center;
}

.form .link a{
    text-decoration: none;
    color: goldenrod;
}

.liw{
    padding-top: 15px;
    padding-bottom: 10px;
    text-align: center;
```

```
}

.icon a{
   text-decoration: none;
   color: #fff;
}

.icon ion-icon{
   color: #fff;
   font-size: 20px;
   padding-left: 14px;
   padding-top: 5px;
   transition: 0.3s ease;
}

.icon ion-icon:hover{
   color:#ff7200;

}
```

Link.javascript:

```
(function(doc){
  var scriptElm = doc.scripts[doc.scripts.length - 1];
  var warn = ['[ionicons] Deprecated script, please remove: ' + scriptElm.outerHTML];

  warn.push('To improve performance it is recommended to set the differential scripts in the head as
follows:')

  var parts = scriptElm.src.split('/');
  parts.pop();
  parts.push('ionicons');
  var url = parts.join('/');

  var scriptElm = doc.createElement('script');
  scriptElm.setAttribute('type', 'module');
  scriptElm.src = url + '/ionicons.esm.js';
  warn.push(scriptElm.outerHTML);
  scriptElm.setAttribute('data-stencil-namespace', 'ionicons');
  doc.head.appendChild(scriptElm);


  scriptElm = doc.createElement('script');
  scriptElm.setAttribute('nomodule', '');
  scriptElm.src = url + '/ionicons.js';
  warn.push(scriptElm.outerHTML);
  scriptElm.setAttribute('data-stencil-namespace', 'ionicons');
  doc.head.appendChild(scriptElm)

  console.warn(warn.join('\n'));
```

})(document);

OUTPUT: