Inspiring Excellence

CSE423 Group Project

(Spring-2023)

Group-1

Project: Guessing Game

Section: 10

Submisison Date: 18/04/2023

Member's Name	Student ID	Email ID
Farhan Faruk	20301137	farhan.faruk@g.bracu.ac.bd
Mahdi Hasan Bhuiyan	20101541	mahdi.hasan.bhuiyan@g.bracu.ac.bd
Abu Bakar Hasnath	20301037	abu.bakar.hasnath@g.bracu.ac.bd
Utsha Sen Dhruba	20301338	utsha.sen.dhruba@g.bracu.ac.bd
Mushfera Fatema	20101272	mushfera.fatema@g.bracu.ac.bd

Group Member’s Information:

Code:

```
from OpenGL.GL import *
from OpenGL.GLUT import *
import OpenGL as gl
import random
import numpy as np
import math

height = 800                                                    #Necessary variables
width = 600
buttonX = []
buttonY = []
visibleButton = [1] * 26
keyboardKeys = [b'a', b'b', b'c', b'd', b'e', b'f', b'g', b'h', b'i', b'j', b'k', b'l', b'm',
                b'n', b'o', b'p', b'q', b'r', b's', b't', b'u', b'v', b'w', b'x', b'y', b'z']

won = True                                                      #Initially
pressedCount = 0
wasted_lives = 0
word = ""
displayWord = ""
hintWord = ""
guessed = []
wordList = []

with open('words.txt') as f:
    for line in f:
        wordList.append(line[:len(line) - 1])                   # Reading text file for our words and hints.
        if 'str' in line:
            break
wordList.pop()

def draw_points(x, y, size=2):
    glPointSize(size)
    glBegin(GL_POINTS)                                          #Draw Point Algorithm
    glVertex2f(x, y)
    glEnd()

def midpointLine(x1, y1, x2, y2, size=3):
    zone = eight_WayZone(x1, y1, x2, y2)
    mx1, my1, mx2, my2 = convertToZero(x1, y1, x2, y2, zone)
    dx = mx2 - mx1                                              #Mid Point Line Algorithm
    dy = my2 - my1
    d = (2 * dy) - dx
    delNE = 2 * (dy - dx)
    delE = 2 * dy
    while mx1 <= mx2 and my1 <= my2:
        convertBacktoOriginalZone(mx1, my1, zone, size)
        mx1 = mx1 + 1
        if d >= 0:
            my1 = my1 + 1
            d = d + delNE
```

```
        else:
            d = d + delE
def eight_WayZone(x1, y1, x2, y2):
    dx = x2 - x1
    dy = y2 - y1
    if abs(dx) >= abs(dy):
        if dx >= 0 and dy >= 0:
            return 0
        elif dx < 0 and dy >= 0:                    #Eight Way Zone Conversion Algorithm
            return 3
        elif dx < 0 and dy < 0:
            return 4
        elif dx >= 0 and dy < 0:
            return 7
    else:
        if dx >= 0 and dy >= 0:
            return 1
        elif dx < 0 and dy >= 0:
            return 2
        elif dx >= 0 and dy < 0:
            return 6
        elif dx < 0 and dy < 0:
            return 5

def convertToZero(x1, y1, x2, y2, zone):            #Convert to Zone 0
    if zone == 1:
        nx1 = y1
        ny1 = x1
        nx2 = y2
        ny2 = x2
        return nx1, ny1, nx2, ny2
    elif zone == 2:
        nx1 = y1
        ny1 = -x1
        nx2 = y2
        ny2 = -x2
        return nx1, ny1, nx2, ny2
    elif zone == 3:
        nx1 = -x1
        ny1 = y1
        nx2 = -x2
        ny2 = y2
        return nx1, ny1, nx2, ny2
    elif zone == 4:
        nx1 = -x1
        ny1 = -y1
        nx2 = -x2
        ny2 = -y2
        return nx1, ny1, nx2, ny2
    elif zone == 5:
        nx1 = -y1
        ny1 = -x1
        nx2 = -y2
        ny2 = -x2
```

```
        return nx1, ny1, nx2, ny2
    elif zone == 6:
        nx1 = -y1
        ny1 = x1
        nx2 = -y2
        ny2 = x2
        return nx1, ny1, nx2, ny2
    elif zone == 7:
        nx1 = x1
        ny1 = -y1
        nx2 = x2
        ny2 = -y2
        return nx1, ny1, nx2, ny2
    elif zone == 0:
        nx1 = x1
        ny1 = y1
        nx2 = x2
        ny2 = y2
        return nx1, ny1, nx2, ny2

def convertBacktoOriginalZone(x1, y1, zone, size):
    glPointSize(size)
    glBegin(GL_POINTS)
    if zone == 1:
        glVertex2f(y1, x1)
    elif zone == 2:
        glVertex2f(-y1, x1)
    elif zone == 3:
        glVertex2f(-x1, y1)
    elif zone == 4:
        glVertex2f(-x1, -y1)
    elif zone == 5:                                    #Convert Back to Original Zone
        glVertex2f(-y1, -x1)
    elif zone == 6:
        glVertex2f(y1, -x1)
    elif zone == 7:
        glVertex2f(x1, -y1)
    elif zone == 0:
        glVertex2f(x1, y1)
    glEnd()

def circle(x1, y1, r):
    x = 2 + r
    y = 0
    d = 0
    while (x >= y):
        draw_points(x1 + x, y1 + y)
        draw_points(x1 + y, y1 + x)
        draw_points(x1 - y, y1 + x)
        draw_points(x1 - x, y1 + y)
        draw_points(x1 - x, y1 - y)                    #Mid Point Circle Algorithm
        draw_points(x1 - y, y1 - x)
        draw_points(x1 + y, y1 - x)
        draw_points(x1 + x, y1 - y)
```

```
        if (d <= 0):
            y = y + 1
            d = d + 2 * y + 1
        if (d > 0):
            x = x - 1
            d = d - 2 * x + 1

def drawText(text, x, y):                                   #Draw text function font size-18
    glColor3fv((1, 0, 0))
    glWindowPos2f(x, y)
    for ch in text:
        glutBitmapCharacter(fonts.GLUT_BITMAP_HELVETICA_18, ord(ch))

def drawTextL(text, x, y):
    glColor3fv((1, 0, 0))
    glWindowPos2f(x, y)                                     #Draw text function font size-24
    for ch in text:
        glutBitmapCharacter(fonts.GLUT_BITMAP_TIMES_ROMAN_24, ord(ch))

def drawCircleButtons():                                    #Circles
    global buttonX
    global buttonY
    buttonX = []                                            #Letter button on the screen
    buttonY = []
    buttonStartX = 70                                       #alphabets & button on x axis
    buttonStartY = 180                                      #alphabets & button on y axis
    for i in range(2):
        for j in range(13):
            buttonX.append(buttonStartX + j * 50 + 28)
            buttonY.append(buttonStartY - i * 80)

            if visibleButton[j + i * 13] == 1:
                circle(buttonStartX + j * 50 + 30, buttonStartY - i * 80, 20)

def drawPole():
    glColor3fv((0, 0, 1))
    midpointLine(130, 570, 130, 350, 7)                     #vertical line
    glColor3fv((0, 0, 1))
    midpointLine(115, 560, 220, 560, 7)                     #horizontal line
    glColor3fv((0, 0, 1))
    midpointLine(115, 530, 160, 570, 7)                     #horizontal line
    glColor3fv((0, 0, 1))
    midpointLine(200, 527, 200, 560, 5)                     #vertical line

def drawTextOnButton():                                     #alphabets inside circle
    global buttonX
    global buttonY                                          #use of text render
    for i in range(0, len(buttonX)):
        if visibleButton[i] == 1:
            drawText(chr(65 + i), buttonX[i] - 5, buttonY[i] - 5)   #character alignment

def drawHangModel():
    if wasted_lives==0:
```

```
        circle(700, 300, 25)
        midpointLine(700, 475, 700, 420)
        midpointLine(710, 475, 710, 420)
        midpointLine(720, 475, 720, 420)
        midpointLine(730, 475, 730, 420)
        midpointLine(690, 475, 690, 420)

    if wasted_lives > 0:                                    #Head
        circle(200, 500, 25)
        midpointLine(193, 486, 207, 486)
        draw_points(210, 505, 6)                            #right eye
        draw_points(190, 505, 6)                            #left eye

    if wasted_lives > 1:                                    # Body
        midpointLine(200, 475, 200, 420)
    if wasted_lives > 2:                                    # Left leg
        midpointLine(170, 380, 200, 420)
    if wasted_lives > 3:                                    # Right leg
        midpointLine(230, 380, 200, 420)
    if wasted_lives > 4:                                    # Left hand
        midpointLine(170, 440, 200, 460)
    if wasted_lives > 5:                                    # Right hand
        midpointLine(230, 440, 200, 460)

def drawBox():
    glColor3fv((0, 1, 0))
    midpointLine(40, 580, 40, 20, 8)
    glColor3fv((0, 1, 0))
    midpointLine(40, 580, 760, 580, 8)
    glColor3fv((0,1, 0))
    midpointLine(760, 580, 760, 20, 8)
    glColor3fv((0, 1, 0))
    midpointLine(40, 20, 760, 20, 8)

                                                            #Text rendering
def drawInfo():
    data = ["CSE-423 Sec:10 Group:01:", "Mahdi Hasan Bhuiyan - 20101541", "Farhan Faruk - 20301137",
            "Mushfera Fatema - 20101272", "Abu Bakar Hasnath - 20301037" "Utsha Sen Dhruba - 20301338"]

def draw():
    drawBox()                                               #Mistake count
    drawInfo()
    if pressedCount == 0:
        drawTextL("Press 'SPACE' to Start!!!", 280, 320)
    if not won and wasted_lives != 6:
        drawPole()
        drawCircleButtons()
        drawTextOnButton()
        drawTextL("HINT: " + hintWord, 370, 480)
        drawTextL(displayWord, 380, 440)
        drawHangModel()
    if won and wasted_lives < 6 and len(guessed) > 0:
        drawTextL("You Won!", 330, 350)
```

```
        drawTextL("Good Job!!", 325, 300)
        drawTextL("Press 'Space' key for restart..", 275, 250)

    if wasted_lives == 6:
        drawTextL("Oucchhh... You DEAD!", 270, 350)
        drawTextL("Study harder!!", 325, 300)
        drawTextL("Press 'Space' key for restart..", 265, 250)

def iterate():
    glViewport(0, 0, height, width)
    glMatrixMode(GL_PROJECTION)
    try:
        glPushMatrix()
    except:
        glPopMatrix()
        glPopMatrix()
    glLoadIdentity()
    glOrtho(0.0, height, 0.0, width, 0.0, 1.0)
    glMatrixMode(GL_MODELVIEW)
    glLoadIdentity()

def showScreen():
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glLoadIdentity()
    iterate()
    try:
        glPopMatrix()
    except:
        pass
    glClear(GL_COLOR_BUFFER_BIT)
    glColor3f(0, 0, 0)
    gameLogic()
    draw()
    glutSwapBuffers()

def onButtonPress(key, x, y):
    global wasted_lives, won, pressedCount
    if key == b' ' and (wasted_lives == 6 or (pressedCount > 0 and won == True)):          #input
        resetGame()
    elif wasted_lives == 6:                                                                #Push keyboard
        return
    if key == b' ' or pressedCount > 0:
        pressedCount += 1                                                                  #count increment
    else:
        return

    for i in range(len(keyboardKeys)):
        if key == keyboardKeys[i]:
            if chr(65 + i) not in word and chr(65 + i) not in guessed:
                wasted_lives += 1                                                          #wested life increment
            guessed.append(chr(65 + i))
            visibleButton[i] = 0
    won = True
    for letter in word:
```

```
        if letter not in guessed:
            won = False
            break
    glutPostRedisplay()

def Trans():
    a = math.cos(math.radians(45))
    b = math.sin(math.radians(45))

    r = np.array([[-0.7, -0.7, 0],
                  [b, a, 0],
                  [0, 0, 1]])

    sc = 0.5
    s = np.array([[0.5, 0, 0],
                  [0, sc, 0],
                  [0, 0, 1]])

    rs = np.matmul(r, s)

    v1 = np.array([[.2],
                   [.2],
                   [1]])
    v2 = np.array([[-.2],
                   [.2],
                   [1]])
    v3 = np.array([[-.2],
                   [-.2],
                   [1]])
v4 = np.array([[.2],
               [-.2],
               [1]])

v11 = np.matmul(r,v1)
v22 = np.matmul(r,v2)
v33 = np.matmul(r,v3)
v44 = np.matmul(r,v4)

v11 = np.matmul(s,v1)
v22 = np.matmul(s,v2)
v33 = np.matmul(s,v3)
v44 = np.matmul(s,v4)

v11 = np.matmul(rs, v1)
v22 = np.matmul(rs, v2)
v33 = np.matmul(rs, v3)
v44 = np.matmul(rs, v4)

gl.glColor3f(1, 0, 0)
gl.glBegin(gl.GL_QUADS)
gl.glVertex2f(v11[0][0], v11[1][0])
gl.glVertex2f(v22[0][0], v22[1][0])
gl.glVertex2f(v33[0][0], v33[1][0])
gl.glVertex2f(v44[0][0], v44[1][0])
```

#Fill in the blanks

#rotation

#scaling

#rotation scaling

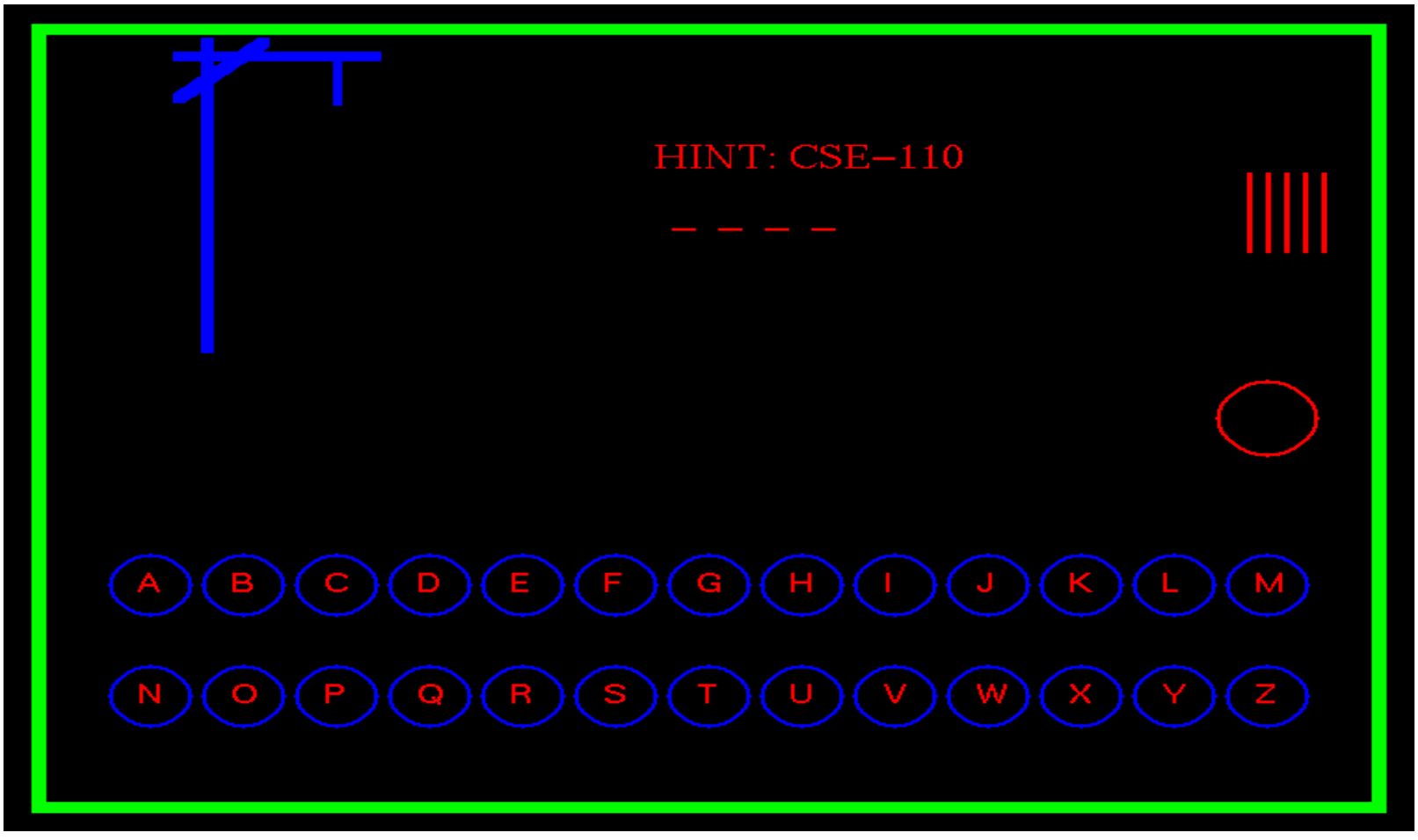
```
    gl.glEnd()

def gameLogic():
    global displayWord
    displayWord = ""
    for letter in word:                                          #Reset function for the game
        if letter in guessed:
            displayWord += letter + " "
        else:
            displayWord += "_ "

def resetGame():
    global word, displayWord, guessed, visibleButton
    global pressedCount, wasted_lives, hintWord, wordList
    randomChoice = random.choice(wordList).split('_')            #random word choose from list
    word = randomChoice[0]
    hintWord = randomChoice[1]
    displayWord = ""
    guessed = []
    visibleButton = [1] * 26
    pressedCount = 0
    wasted_lives = 0

resetGame()
glutInit()
glutInitDisplayMode(GLUT_RGBA)
glutInitWindowSize(height, width)
glutInitWindowPosition(0, 0)
wind = glutCreateWindow(b"GuessingGame")
glutDisplayFunc(showScreen)
glutKeyboardFunc(onButtonPress)
glutMainLoop()
```

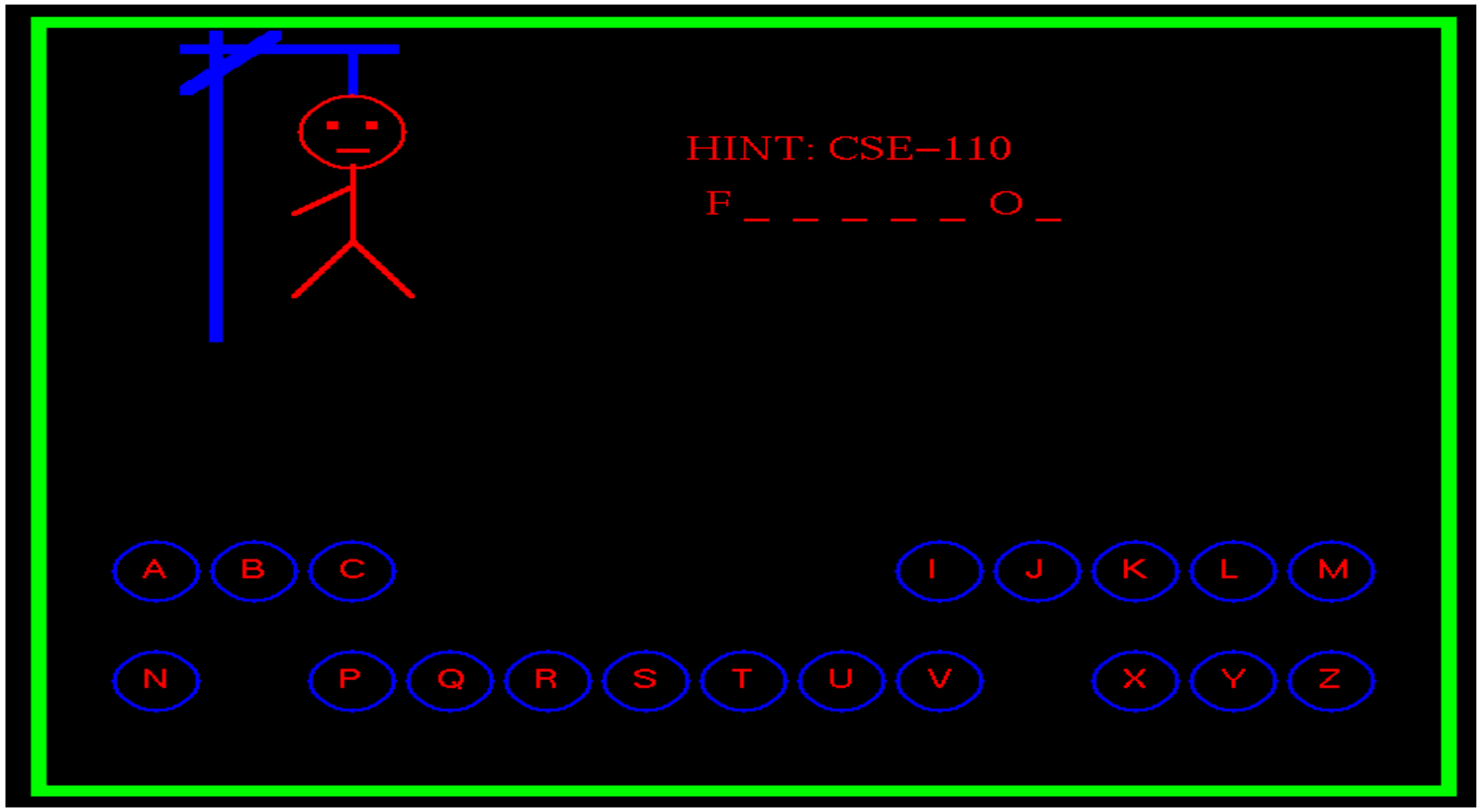
####################Inside the Test file####################

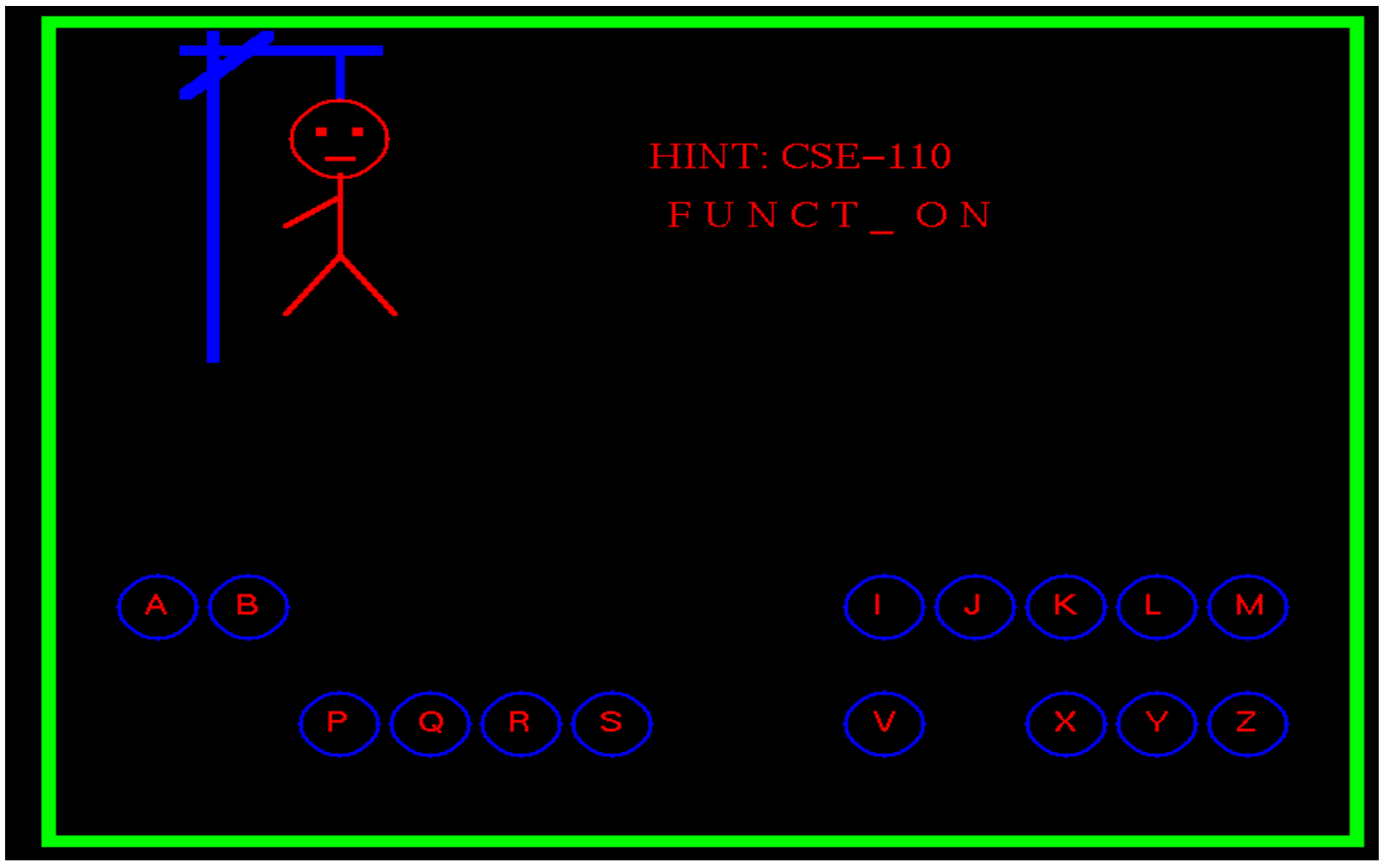
```
FUNCTION_CSE-110
LOOP_CSE-110
OOP_CSE-111
INHERITANCE_CSE-111
HASHTABLE_CSE-220
RECURSION_CSE-220
BFS_CSE-221
DFS_CSE-221
PERMUTATION_CSE-230
COMBINATION_CSE-230
KCL_CSE-250
KVL_CSE-250
BJT_CSE-251
MOSFET_CSE-251
FLIPFLOP_CSE-260
MULTIPLEXER_CSE-260
MIPS_CSE-340
STALL_CSE-340
EER_CSE-370
DATABASE_CSE-370
SCRUM_CSE-470
```

MVC_CSE-470
SINGLETON_CSE-470
####################Inside the Test file####################

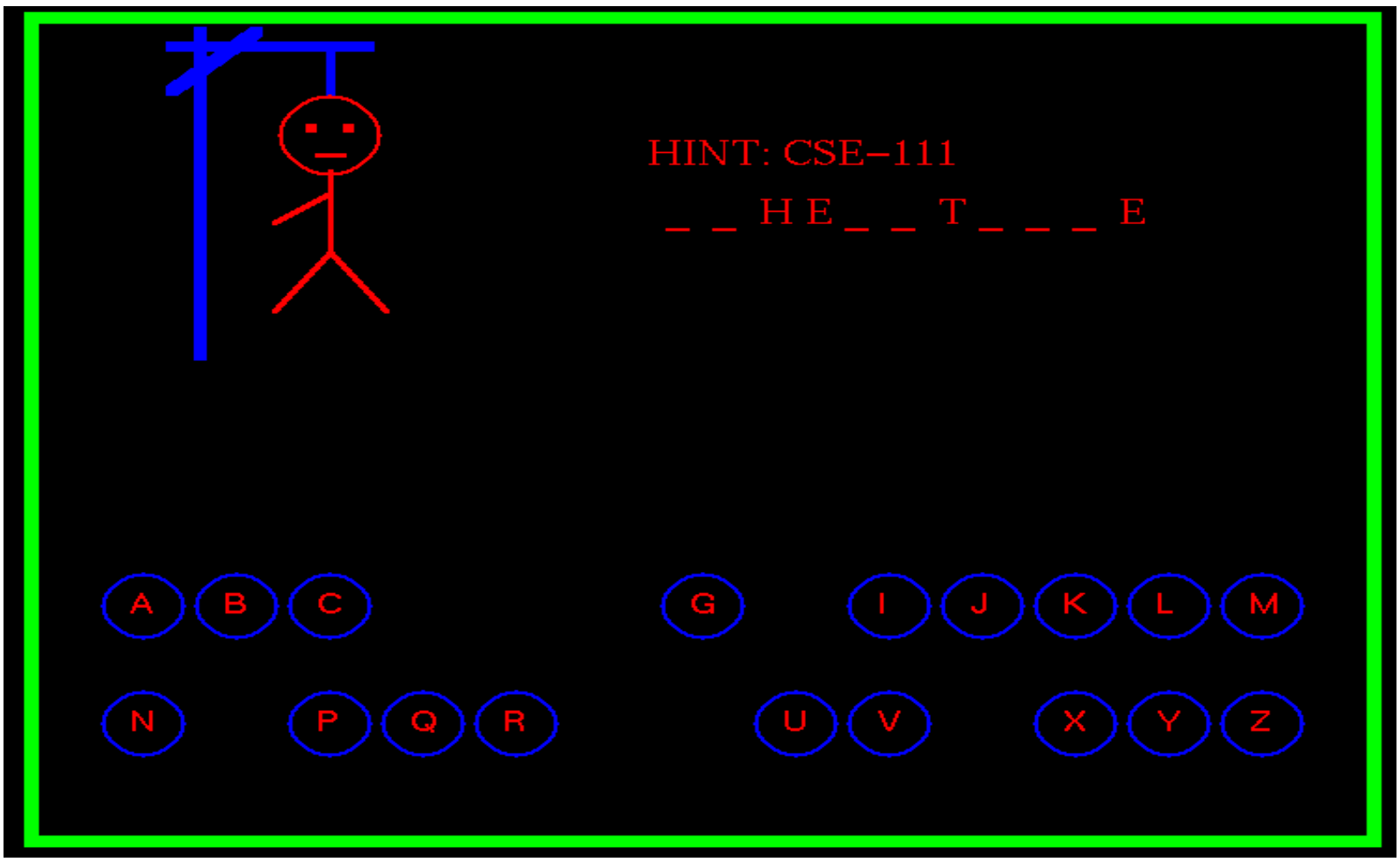
Output:

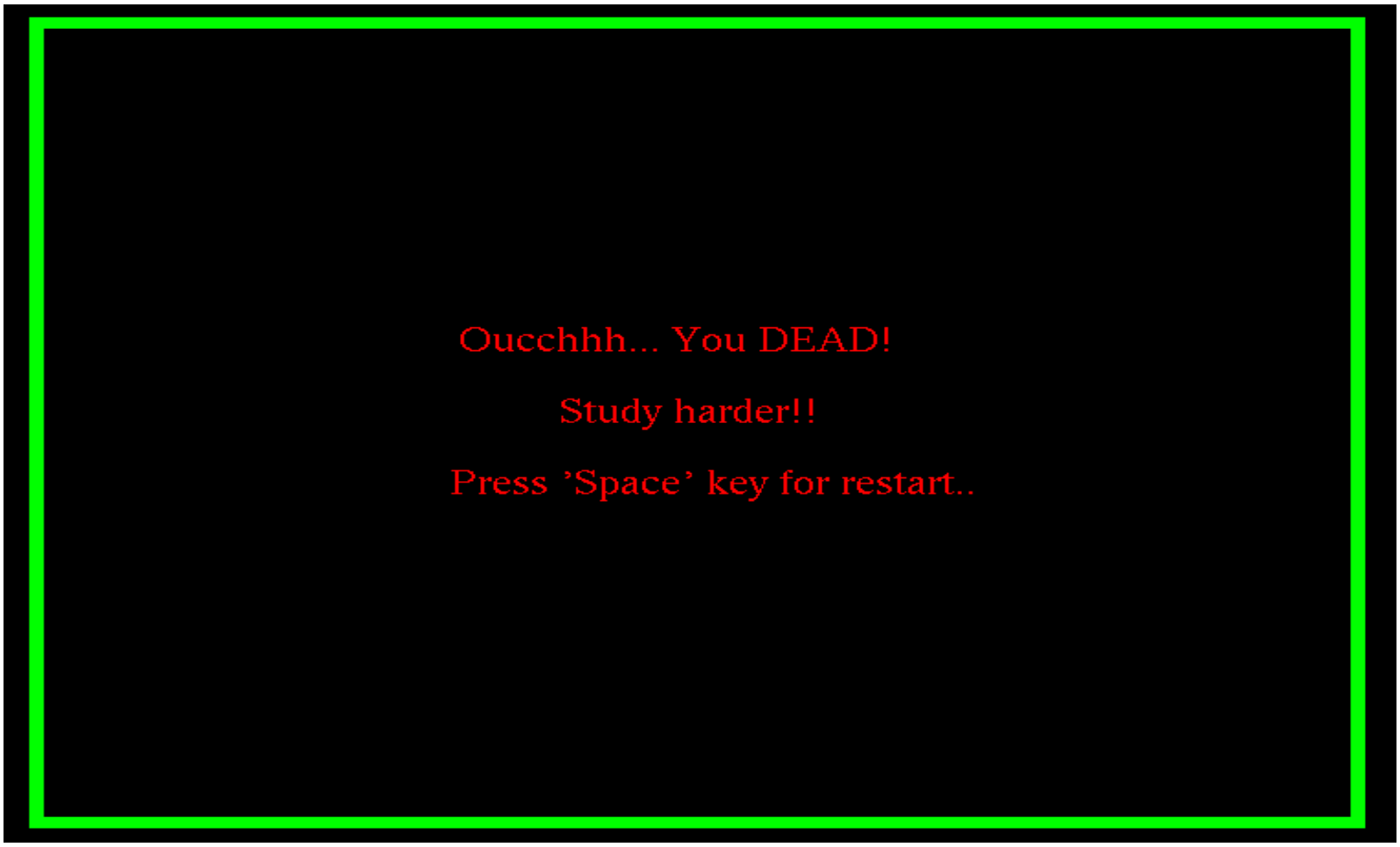



HINT: CSE–110
F _ _ _ _ _ O _
A B C I J K L M
N P Q R S T U V X Y Z


HINT: CSE–110
FUNCT_ON
A B I J K L M
P Q R S V X Y Z


You Won!
Good Job!!
Press 'Space' key for restart..


HINT: CSE-111
_ _ H E _ _ T _ _ _ E
A B C
G
I J K L M
N
P Q R
U V
X Y Z



Thank you…