PYTHON ESSENTIALS LABORATORY

MINI PROJECT

A REPORT

Submitted by

VARSHA.G.A(9517202306111)

SHREE HARINI.K (9517202306092)

MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI
(AUTONOMOUS)

JULY 2024

BONAFIDE CERTIFICATE

Certified that this project report is the bonafide work of

K. SHREE HARINI (9517202306092). (23BIT089),

G.A. VARSHA (9517202306111). (23BIT096)

who carried out the project work under my supervision.

SIGNATURE

Mrs. M. Blessa Binolin Pepsi, M.E.,

STAFF IN-CHARGE

Assistant Professor (Sl. G)

Information Technology

Mepco Schlenk Engineering College

Virudhunagar Dt. – 626 005

SIGNATURE

Dr. T. Revathi, M.E., Ph.D.,

HEAD OF THE DEPARTMENT

Senior Professor

Information Technology

Mepco Schlenk Engineering College

Virudhunagar Dt. – 626 005

Game name: **WORD HUNT**

Game description:

Dive into the timeless world of Hangman, where your vocabulary and guessing skills will be put to the ultimate test! You are the hero who must uncover the hidden word before the stick figure meets its doom. Each wrong guess brings the gallows closer to completion. Can you save the day by guessing the word correctly, or will the hangman meet his fate?

Gameplay Techniques:

- **Goal**: Guess the hidden word by selecting one letter at a time. Avoid making too many incorrect guesses, or the hangman will be completed, and the game will be lost.

- **Word List**: The game features a diverse set of words from various categories, challenging players to test their knowledge and guessing skills.

- **Guesses**: Players have a limited number of wrong guesses (typically 6-7). Each incorrect guess adds a part to the hangman drawing.

- **Hints and Feedback**: Correctly guessed letters are revealed in their respective positions in the word. Incorrect guesses are recorded and displayed to the player.

- **End of Game**: The game ends when the word is fully revealed (victory) or when the hangman drawing is complete (loss).

How to Play:

1. **Start the Game**: Launch the game from your Python environment

2. **Guess a Letter**: Enter a letter you think might be in the word. The game will reveal its position if you are correct or add a part to the hangman drawing if you are wrong.

3. **Continue Guessing**: Keep guessing letters to reveal more of the word. Be mindful of incorrect guesses as they bring you closer to completing the hangman.

4. **Win or Lose**: Win the game by revealing all the letters in the word or lose if the hangman drawing is completed before you can guess the word.

Modules and packages used:

PACKAGE USED: RANDOM ()

Python random module generates random numbers in python. These are pseudo-random numbers they are not truly random. This package can be used to random actions such as generating random numbers, printing random numbers, list, string, etc.
This package a variety of inbuilt functions and the one used in our game is used a random to generate a word from an existing list of words**.** This is a handy tool to choose a particular word in a set of words. The choices method returns multiple random elements from the list with replacement.
Since it is a built-in package, the functioning of the game is possible without pi game or other sources. This package was imported for our game to generate words to be guessed.
A function was created, the main purpose of this function is to check if the letters guessed by the user is present in the word generated or not. If the condition is true, then the letter is added to displayed letters else the word is returned and with every wrong word an image (stick man figure) slowly disappears.

Hangman:

If the condition is false then a set of stick figures imported into the folder is executed where for every wrong guess, a part of the stick figures slowly disappears. This follows the basic concept of a hangman game.
Now a function is created for hangman, where the maximum number of attempts corresponds to the length of the word, following three extra tries. Initially the number of attempts is zero, a message is printed on the screen to enter the letters guessed by the user, if the letter is already guessed a message appears to inform the user. Now with every attempt the number of guessed attempts is counted and if it matches with the maximum attempts then the game terminates with a

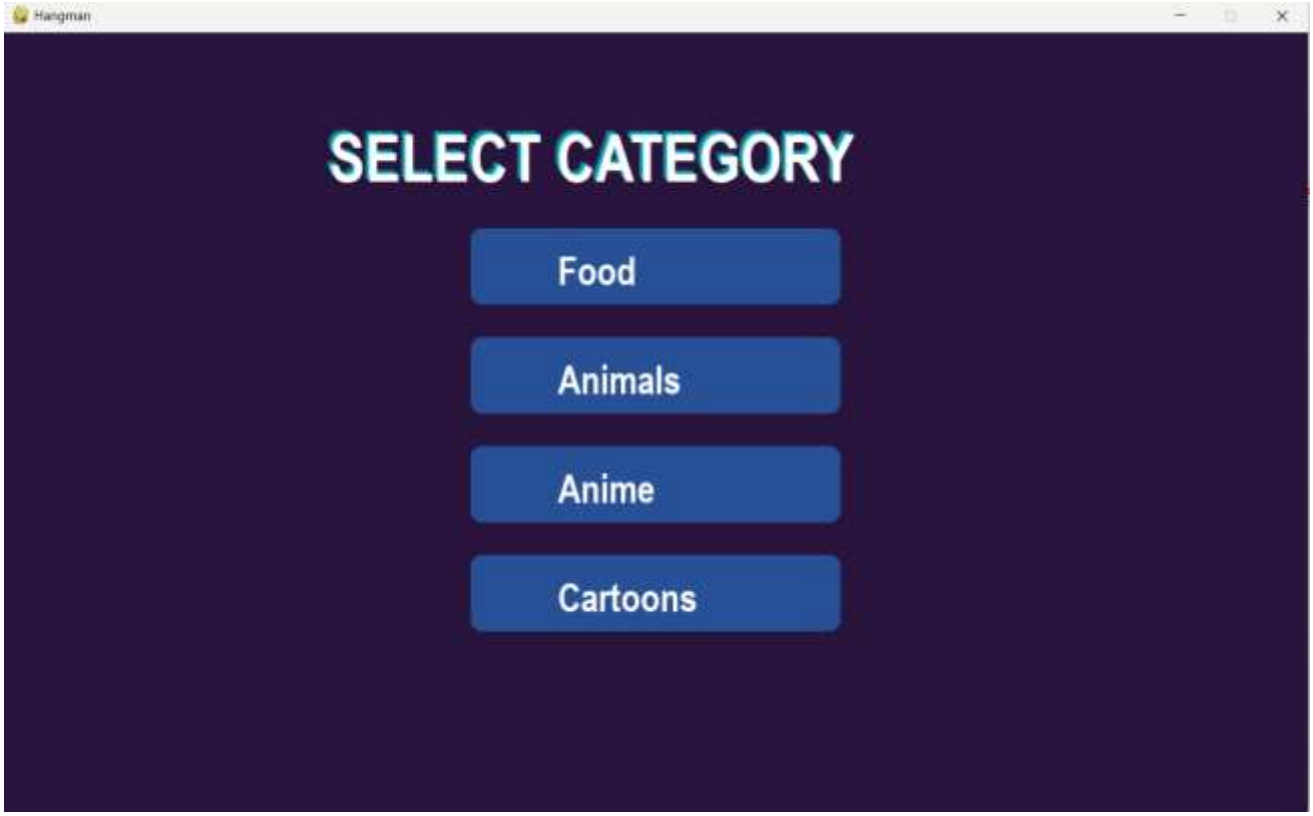
message informing that the player has lost, if the entire word is guessed before the said maximum attempts the player wins. This operation follows basic python looping concepts.

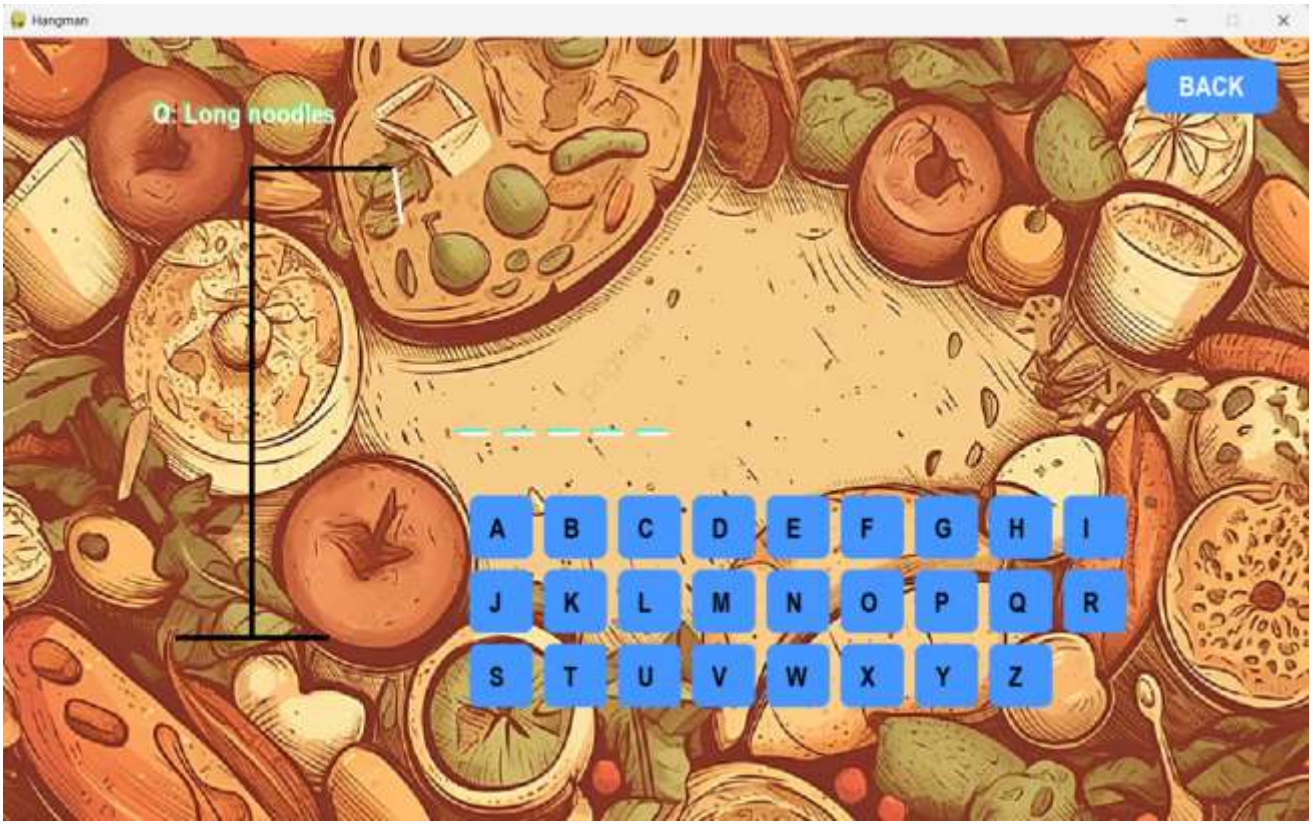
OTHER KEYWORDS USED:

Continue statement
The continue statement is used to skip the remaining code inside a loop for the current iteration only. For instance, we used the keyword in the hangman function created.

Break statement
'Break' in Python is a loop control statement. It is used to control the sequence of the loop. Suppose you want to terminate a loop and skip to the next code after the loop, break will help you do that. Break statement is used to end the game after the maximum number of tries is over.

Screenshots:


Hangman
SELECT CATEGORY
Food
Animals
Anime
Cartoons


Hangman
BACK
Q: Long noodles
A B C D E F G H I
J K L M N O P Q R
S T U V W X Y Z

Code:

```
import pygame
import random
import math
import sys

pygame.init()
pygame.mixer.init()

WIDTH, HEIGHT = 1200, 720
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Hangman Ultimate")

clock = pygame.time.Clock()

WHITE=(255,255,255)
BLACK=(0,0,0)
BLUE=(70,150,255)
GREEN=(80,255,120)
GRAY=(120,120,120)
NEON=(0,255,255)

font_big=pygame.font.SysFont("arial",60,bold=True)
font_mid=pygame.font.SysFont("arial",36,bold=True)
font_small=pygame.font.SysFont("arial",25,bold=True)


def load_bg(path,color):
    try:
        img=pygame.image.load(path).convert()
        return pygame.transform.scale(img,(WIDTH,HEIGHT))
    except:
        surf=pygame.Surface((WIDTH,HEIGHT))
        surf.fill(color)
        return surf

loading_bg=load_bg("assets/loading.jpg",(20,20,50))
menu_bg=load_bg("assets/menu.jpg",(40,20,60))

category_bg={
 "Food":load_bg("assets/food.jpg",(40,20,20)),
 "Animals":load_bg("assets/animal.jpg",(20,40,20)),
 "Anime":load_bg("assets/anime.jpg",(20,20,40)),
 "Cartoons":load_bg("assets/catroon.jpg",(40,40,20))
```

```
}


def safe_sound(path):
    try:
        return pygame.mixer.Sound(path)
    except:
        return None

try:
    pygame.mixer.music.load("assets/music.mp3")
    pygame.mixer.music.set_volume(0.5)
    pygame.mixer.music.play(-1)
except:
    pass

correct_sound=safe_sound("assets/correct.mp3")
wrong_sound=safe_sound("assets/wrong.mp3")
click_sound=safe_sound("assets/click.mp3")


particles=[]

def create_particles():
    particles.clear()
    for i in range(120):
        particles.append([
            WIDTH//2,
            HEIGHT//2,
            random.randint(-6,6),
            random.randint(-6,6),
            random.randint(3,6)
        ])

def draw_particles():
    for p in particles:
        pygame.draw.circle(screen,
            (random.randint(100,255),
             random.randint(100,255),
             random.randint(100,255)),
            (p[0],p[1]),p[4])

        p[0]+=p[2]
        p[1]+=p[3]
        p[4]=max(1,p[4]-0.05)

categories={
```

```
 "Food":[("Italian cheesy dish","PIZZA"),
         ("Fast food sandwich","BURGER"),
         ("Long noodles","PASTA")],

 "Animals":[("King of jungle","TIGER"),
           ("Animal with trunk","ELEPHANT"),
           ("Climbs trees","MONKEY")],

 "Anime":[("Ninja anime hero","NARUTO"),
          ("Soul reaper anime","BLEACH"),
          ("Pirate anime","ONEPIECE")],

 "Cartoons":[("Blue robot cat","DORAEMON"),
            ("Alien watch boy","BEN10"),
            ("Electric mouse","POKEMON")]
}


def glow_text(text,font,color,x,y):
    for i in range(4,0,-1):
        s=font.render(text,True,color)
        s.set_alpha(70)
        screen.blit(s,(x-i,y-i))
    screen.blit(font.render(text,True,WHITE),(x,y))
def fade():
    f=pygame.Surface((WIDTH,HEIGHT))
    f.fill(BLACK)
    for a in range(0,255,10):
        f.set_alpha(a)
        screen.blit(f,(0,0))
        pygame.display.update()
        pygame.time.delay(6)


def start_screen():
    btn=pygame.Rect(480,400,240,80)
    while True:
        screen.blit(loading_bg,(0,0))

        hover=btn.collidepoint(pygame.mouse.get_pos())
        color=(70,150,255) if hover else (40,80,150)

        pygame.draw.rect(screen,color,btn,border_radius=15)
        screen.blit(font_mid.render("START",True,WHITE),
                    (btn.x+60,btn.y+20))

        pygame.display.update()
```

```
        for e in pygame.event.get():
            if e.type==pygame.QUIT:
                pygame.quit();sys.exit()

            if e.type==pygame.MOUSEBUTTONDOWN:
                if btn.collidepoint(e.pos):
                    if click_sound: click_sound.play()
                    fade()
                    return


    def category_menu():

        names=["Food","Animals","Anime","Cartoons"]
        buttons=[]

        y=180
        for n in names:
            buttons.append((n,pygame.Rect(430,y,340,70)))
            y+=100

        while True:

            screen.blit(menu_bg,(0,0))
            glow_text("SELECT CATEGORY",font_big,NEON,300,80)

            mouse=pygame.mouse.get_pos()

            for n,r in buttons:
                col=(70,150,255) if r.collidepoint(mouse) else (40,80,150)
                pygame.draw.rect(screen,col,r,border_radius=10)
                screen.blit(font_mid.render(n,True,WHITE),(r.x+80,r.y+18))

            pygame.display.update()

            for e in pygame.event.get():
                if e.type==pygame.QUIT:
                    pygame.quit();sys.exit()

                if e.type==pygame.MOUSEBUTTONDOWN:
                    for n,r in buttons:
                        if r.collidepoint(e.pos):
                            if click_sound: click_sound.play()
                            fade()
                            return n
    def keyboard():
```

```
    keys=[]
    sx,sy=430,420
    for i in range(26):
        x=sx+(i%9)*68
        y=sy+(i//9)*68
        keys.append([chr(65+i),pygame.Rect(x,y,58,58)])
    return keys
def draw_hangman(stage,tick,offset):

    swing=int(math.sin(tick*0.05)*8)
    ox,oy=offset

    pygame.draw.line(screen,BLACK,(160+ox,550+oy),(300+ox,550+oy),5)
    pygame.draw.line(screen,BLACK,(230+ox,550+oy),(230+ox,120+oy),5)
    pygame.draw.line(screen,BLACK,(230+ox,120+oy),(360+ox,120+oy),5)

    pygame.draw.line(screen,WHITE,(360+ox,120+oy),(360+swing+ox,170+oy),4)

    x=360+swing+ox

    if stage>=1: pygame.draw.circle(screen,BLACK,(x,200+oy),30,4)
    if stage>=2: pygame.draw.line(screen,BLACK,(x,230+oy),(x,320+oy),4)
    if stage>=3: pygame.draw.line(screen,BLACK,(x,260+oy),(x-45,300+oy),4)
    if stage>=4: pygame.draw.line(screen,BLACK,(x,260+oy),(x+45,300+oy),4)
    if stage>=5: pygame.draw.line(screen,BLACK,(x,320+oy),(x-45,370+oy),4)
    if stage>=6: pygame.draw.line(screen,BLACK,(x,320+oy),(x+45,370+oy),4)

def game(category):

    bg=category_bg[category]
    question,word=random.choice(categories[category])

    guessed=[]
    wrong=0
    keys=keyboard()
    tick=0
    shake=0

    back_btn=pygame.Rect(1050,20,120,50)

    while True:

        tick+=1

        if shake>0:
            offset=(random.randint(-6,6),random.randint(-6,6))
            shake-=1
```

```
else:
    offset=(0,0)

screen.blit(bg,offset)

pygame.draw.rect(screen,BLUE,
    (back_btn.x+offset[0],back_btn.y+offset[1],
     back_btn.w,back_btn.h),border_radius=10)

screen.blit(font_small.render("BACK",True,WHITE),
            (1080+offset[0],30+offset[1]))

glow_text("Q: "+question,font_small,GREEN,140+offset[0],55+offset[1])

display=""
for l in word:
    display += l+" " if l in guessed else "_ "

glow_text(display,font_big,NEON,420+offset[0],300+offset[1])

draw_hangman(wrong,tick,offset)

for l,r in keys:
    c=BLUE if l not in guessed else GRAY
    pygame.draw.rect(screen,c,
        (r.x+offset[0],r.y+offset[1],r.w,r.h),border_radius=8)
    screen.blit(font_small.render(l,True,BLACK),
                (r.x+18+offset[0],r.y+15+offset[1]))

pygame.display.update()
if "_" not in display:
    create_particles()
    for i in range(80):
        screen.blit(bg,(0,0))
        draw_particles()
        pygame.display.update()
        clock.tick(60)
    fade()
    return

if wrong>=6:
    fade()
    return

for e in pygame.event.get():

    if e.type==pygame.QUIT:
```

```
            pygame.quit();sys.exit()

        if e.type==pygame.MOUSEBUTTONDOWN:

            if back_btn.collidepoint(e.pos):
                if click_sound: click_sound.play()
                fade()
                return

            for l,r in keys:
                if r.collidepoint(e.pos) and l not in guessed:

                    guessed.append(l)

                    if l in word:
                        if correct_sound: correct_sound.play()
                    else:
                        wrong+=1
                        shake=10
                        if wrong_sound: wrong_sound.play()

    clock.tick(60)
start_screen()

while True:
    cat=category_menu()
    game(cat)
```