|  |  |  |
| --- | --- | --- |
|  |  | Memory Game  By Sun Woo Kim |

# Design Brief

## Primary Objective

Design and develop a memory game using python and its modules. I will be using Python 3 and the Pygame module since it is easy to make simple games like this memory game.

## Target Audience

The target audience is for everyone. The game can be played by people of any age.

## Project Specifics

The features required in this project include:

* A random 6 characters generator using any method
* At least one sub-program and at least one function
* The game should be visually appealing
* The game should be intuitive for the user
* The code should be commented
* The game must include a help/instruction screen explaining to the user how to play the game

# Detailed Design

## Sketches

## A picture containing bird Description automatically generated

A screenshot of a tree

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A picture containing bird, flower

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A picture containing bird, flower

Description automatically generated

A picture containing bird, flower

Description automatically generated

## Program Specifications

My memory game generates 6 random characters and the player is given a time frame in which they will have to remember the characters generated. The time frame will be different depending on which difficulty has been chosen. Some limitations of the program are that the timer functions in Pygame causes the game to freeze until the timer is finished.

## Diagrams

IPO

|  |  |  |
| --- | --- | --- |
| **Input** | **Process** | **Output** |
| Quit button | Calls quit function | Quits game |
| Start button | Calls select difficulty function | Shows difficulty selection screen |
| Instructions button | Calls instructions function | Shows instructions |
| Back button | Calls the previous screen’s function | Shows the previous screen |
| Easy button | Calls the timer function that lasts 10 seconds then calls the game screen | Shows game screen |
| Medium button | Calls the timer function that lasts 5 seconds then calls the game screen | Shows game screen |
| Hard button | Calls the timer function that lasts 5 seconds then calls the game screen | Shows game screen |
| Enter button: With correct answer | Calls a function that checks if the user’s input is equal to the answer then calls the screen for correct | Shows correct screen |
| Enter button: With incorrect answer | Calls a function that checks if the user’s input is equal to the answer then calls the screen for incorrect | Shows incorrect screen |
| Menu button | Calls menu function | Shows menu screen |

Dataflow

A close up of a logo

Description automatically generated

# Algorithm Design

## Pseudocode

IMPORT pygame

IMPORT random

# Initializing pygame

pygame.init()

# Window setup

SET window\_width TO 800

SET window\_height TO 600

SET window TO pygame.display.set\_mode((window\_width, window\_height))

SET background TO pygame.image.load("background.jpg")

SET background\_width TO 3840

SET background\_height TO 2160

SET background\_default\_x TO -1920

SET background\_default\_y TO -1080

# Window Caption

pygame.display.set\_caption("Memory Game")

# Variables

SET clock TO pygame.time.Clock()

SET fps TO 144

SET x\_center TO window\_width/2

SET y\_center TO window\_height/2

SET center TO x\_center, y\_center

SET button\_height TO 50

SET INPUT\_box\_width TO 400

SET INPUT\_box\_height TO 50

SET character\_set TO "QWERTYUIOPASDFGHJKLZXCVBNMqwertyuiopasdfghjklzxcvbnm1234567890"

SET word TO ""

SET INPUT\_text TO ""

SET mouse\_position TO pygame.mouse.get\_pos()

SET mouse\_handled TO False

SET sound\_handled TO False

SET sound\_handled\_position TO [0, 0, 0, 0]

# Fonts

SET largeText TO pygame.font.Font("roboto.ttf", 60)

SET mediumText TO pygame.font.Font("roboto.ttf", 40)

SET smallText TO pygame.font.Font("roboto.ttf", 20)

SET warningText TO pygame.font.Font("roboto.ttf", 10)

# Colours

SET white TO (255, 255, 255)

SET darker\_white TO (150, 150, 150)

SET black TO (0, 0, 0)

SET grey TO (75, 95, 110)

SET dark\_grey TO (35, 35, 35)

SET red TO (255, 0, 0)

SET blue TO (15, 30, 50)

SET lighter\_blue TO (50, 95, 130)

SET light\_blue TO (105, 215, 250)

# Sounds

SET button\_hover\_sound TO pygame.mixer.Sound("Button Hover.ogg")

SET button\_press\_sound TO pygame.mixer.Sound("Button press.ogg")

# Text renderer

DEFINE FUNCTION text\_objects(text, font, colour):

SET textSurface TO font.render(text, True, colour)

RETURN textSurface, textSurface.get\_rect()

# Button renderer

DEFINE FUNCTION button(button\_text, button\_x, button\_y, button\_width, button\_height, action=None):

SET mouse\_handled AS global

SET mouse\_position AS global

SET sound\_handled AS global

SET sound\_handled\_position AS global

# Mouse events

IF mouse\_position != pygame.mouse.get\_pos():

SET mouse\_handled TO True

SET mouse\_position TO pygame.mouse.get\_pos()

SET mouse\_click TO pygame.mouse.get\_pressed()

# Sets mouse\_handled to false IF mouse isnt clicked

IF mouse\_click[0] EQUALS 0:

SET mouse\_handled TO False

# If mouse position is not position of button that handled sound

IF not (sound\_handled\_position[0] > mouse\_position[0] > sound\_handled\_position[1] and sound\_handled\_position[2] > mouse\_position[1] > sound\_handled\_position[3]):

SET sound\_handled TO False

# Button

IF button\_x + button\_width > mouse\_position[0] > button\_x and button\_y + button\_height > mouse\_position[1] > button\_y:

IF not sound\_handled:

SET sound\_handled TO True

# Get position of button that handled the sound

SET sound\_handled\_position[0] TO button\_x + button\_width

SET sound\_handled\_position[1] TO button\_x

SET sound\_handled\_position[2] TO button\_y + button\_height

SET sound\_handled\_position[3] TO button\_y

# Play hover sound

pygame.mixer.Sound.play(button\_hover\_sound)

# Button rectangle

pygame.draw.rect(window, light\_blue, (button\_x, button\_y, button\_width, button\_height))

pygame.draw.rect(window, lighter\_blue, (button\_x + 2, button\_y + 2, button\_width - 4, button\_height - 4))

# Button text

SET TextSurf, TextRect TO text\_objects(button\_text, mediumText, white)

SET TextRect.center TO (button\_x + button\_width/2, button\_y + button\_height/2)

window.blit(TextSurf, TextRect)

IF mouse\_click[0] EQUALS 1 and action is not None and not mouse\_handled:

# Play press sound

pygame.mixer.Sound.play(button\_press\_sound)

action()

ELSE:

# Button rectangle

pygame.draw.rect(window, lighter\_blue, (button\_x, button\_y, button\_width, button\_height))

# Button text

SET TextSurf, TextRect TO text\_objects(button\_text, mediumText, light\_blue)

SET TextRect.center TO (button\_x + button\_width/2, button\_y + button\_height/2)

window.blit(TextSurf, TextRect)

# Splashscreen

DEFINE FUNCTION splashscreen():

SET timer TO 2 \* fps

WHILE True:

# If exit button pressed

FOR event IN pygame.event.get():

IF event.type EQUALS pygame.QUIT:

quit()

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Splashscreen text

SET TextSurf, TextRect TO text\_objects("Sun Woo's Memory Game", largeText, light\_blue)

SET TextRect.center TO (center)

window.blit(TextSurf, TextRect)

# Display update

pygame.display.update()

clock.tick(fps)

# Timer

timer -= 1

IF timer EQUALS 0:

menu()

# Menu

DEFINE FUNCTION menu():

WHILE True:

# If exit button pressed

FOR event IN pygame.event.get():

IF event.type EQUALS pygame.QUIT:

quit()

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Menu text

SET TextSurf, TextRect TO text\_objects("Memory Game", largeText, light\_blue)

SET TextRect.center TO (x\_center, 50)

window.blit(TextSurf, TextRect)

# Start button

button("Start", 10, 215, 100, button\_height, difficulty\_selection)

# Instructions button

button("Instructions", 10, 275, 220, button\_height, instructions)

# Quit button

button("Quit", 10, 335, 80, button\_height, quit)

# Display update

pygame.display.update()

clock.tick(fps)

# Select diffuculty

DEFINE FUNCTION difficulty\_selection():

WHILE True:

# If exit button pressed

FOR event IN pygame.event.get():

IF event.type EQUALS pygame.QUIT:

quit()

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Select diffuculty text

SET TextSurf, TextRect TO text\_objects("Select difficulty", largeText, light\_blue)

SET TextRect.center TO (x\_center, 50)

window.blit(TextSurf, TextRect)

# Easy button

button("Easy", x\_center - 45, y\_center - 100, 90, button\_height, easy)

# Medium button

button("Medium", x\_center - 75, y\_center, 150, button\_height, medium)

# Hard button

button("Hard", x\_center - 45, y\_center + 100, 90, button\_height, hard)

# Back button

button("Back", x\_center - 350, y\_center + 200, 100, button\_height, menu)

# Display update

pygame.display.update()

clock.tick(fps)

# Instructions

DEFINE FUNCTION instructions():

WHILE True:

# If exit button pressed

FOR event IN pygame.event.get():

IF event.type EQUALS pygame.QUIT:

quit()

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Instructions text

SET TextSurf, TextRect TO text\_objects("How to play", largeText, light\_blue)

SET TextRect.center TO (x\_center, y\_center - 200)

window.blit(TextSurf, TextRect)

SET TextSurf, TextRect TO text\_objects("A random set of characters will be displayed on the screen FOR a set amount of time.", smallText, light\_blue)

SET TextRect.center TO (x\_center, y\_center - 100)

window.blit(TextSurf, TextRect)

SET TextSurf, TextRect TO text\_objects("Easy: 5 seconds", smallText, light\_blue)

SET TextRect.center TO (x\_center, y\_center - 50)

window.blit(TextSurf, TextRect)

SET TextSurf, TextRect TO text\_objects("Medium: 3 seconds", smallText, light\_blue)

SET TextRect.center TO (x\_center, y\_center - 25)

window.blit(TextSurf, TextRect)

SET TextSurf, TextRect TO text\_objects("Hard: 1 seconds", smallText, light\_blue)

SET TextRect.center TO (x\_center, y\_center)

window.blit(TextSurf, TextRect)

SET TextSurf, TextRect TO text\_objects("After the timer is up type the characters IN to the INPUT box.", smallText, light\_blue)

SET TextRect.center TO (x\_center, y\_center + 50)

window.blit(TextSurf, TextRect)

# Back button

button("Back", x\_center - 350, y\_center + 200, 100, button\_height, menu)

# Display update

pygame.display.update()

clock.tick(fps)

# Easy

DEFINE FUNCTION easy():

random\_word()

global INPUT\_text

SET timer TO 10 \* fps

WHILE True:

# If exit button pressed

FOR event IN pygame.event.get():

IF event.type EQUALS pygame.QUIT:

quit()

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Display word

SET TextSurf, TextRect TO text\_objects(f"The word is: {word}", largeText, light\_blue)

SET TextRect.center TO (center)

window.blit(TextSurf, TextRect)

# Display timer

SET TextSurf, TextRect TO text\_objects(f"Starting IN {round(timer/fps, 1)}...", mediumText, light\_blue)

SET TextRect.center TO (x\_center, y\_center + 50)

window.blit(TextSurf, TextRect)

# Display update

pygame.display.update()

clock.tick(fps)

# Start game when timer reaches 0

timer -= 1

IF timer EQUALS 0:

SET INPUT\_text TO ""

game()

# Medium

DEFINE FUNCTION medium():

random\_word()

global INPUT\_text

SET timer TO 5 \* fps

WHILE True:

# If exit button pressed

FOR event IN pygame.event.get():

IF event.type EQUALS pygame.QUIT:

quit()

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Display word

SET TextSurf, TextRect TO text\_objects(f"The word is: {word}", largeText, light\_blue)

SET TextRect.center TO (center)

window.blit(TextSurf, TextRect)

# Display timer

SET TextSurf, TextRect TO text\_objects(f"Starting IN {round(timer/fps, 1)}...", mediumText, light\_blue)

SET TextRect.center TO (x\_center, y\_center + 50)

window.blit(TextSurf, TextRect)

# Display update

pygame.display.update()

clock.tick(fps)

# Start game when timer reaches 0

timer -= 1

IF timer EQUALS 0:

SET INPUT\_text TO ""

game()

# Hard

DEFINE FUNCTION hard():

random\_word()

global INPUT\_text

SET timer TO 3 \* fps

WHILE True:

# If exit button pressed

FOR event IN pygame.event.get():

IF event.type EQUALS pygame.QUIT:

quit()

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Display word

SET TextSurf, TextRect TO text\_objects(f"The word is: {word}", largeText, light\_blue)

SET TextRect.center TO (center)

window.blit(TextSurf, TextRect)

# Display timer

SET TextSurf, TextRect TO text\_objects(f"Starting IN {round(timer/fps, 1)}...", mediumText, light\_blue)

SET TextRect.center TO (x\_center, y\_center + 50)

window.blit(TextSurf, TextRect)

# Display update

pygame.display.update()

clock.tick(fps)

# Start game when timer reaches 0

timer -= 1

IF timer EQUALS 0:

SET INPUT\_text TO ""

game()

# Word generation

DEFINE FUNCTION random\_word():

global word

SET word TO ""

SET i TO 0

WHILE i < 6:

word += random.choice(character\_set)

i += 1

# Game

DEFINE FUNCTION game():

global INPUT\_text

global clicked

SET INPUT\_box\_selected TO False

WHILE True:

# Mouse events

SET mouse\_position TO pygame.mouse.get\_pos()

# Events

FOR event IN pygame.event.get():

# If exit button pressed

IF event.type EQUALS pygame.QUIT:

quit()

# Selecting the INPUT box

IF event.type EQUALS pygame.MOUSEBUTTONDOWN:

IF x\_center + INPUT\_box\_width/2 > mouse\_position[0] > x\_center - INPUT\_box\_width/2 and y\_center + INPUT\_box\_height/2 > mouse\_position[1] > y\_center - INPUT\_box\_height/2:

SET INPUT\_box\_selected TO True

ELSE:

SET INPUT\_box\_selected TO False

# Detect keys

IF event.type EQUALS pygame.KEYDOWN:

IF INPUT\_box\_selected:

IF event.key EQUALS pygame.K\_BACKSPACE:

SET INPUT\_text TO INPUT\_text[:-1]

ELSEIF event.key EQUALS pygame.K\_RETURN:

confirm()

ELSE:

IF len(INPUT\_text) < 6:

INPUT\_text += event.unicode

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Title text

SET TextSurf, TextRect TO text\_objects("Type the word below", mediumText, light\_blue)

SET TextRect.center TO (x\_center, y\_center - 200)

window.blit(TextSurf, TextRect)

# Enter button

button("Enter", x\_center - 50, y\_center + 200, 100, button\_height, confirm)

# Input box

IF INPUT\_box\_selected:

# Box

pygame.draw.rect(window, light\_blue, (x\_center - INPUT\_box\_width/2, y\_center - INPUT\_box\_height/2, INPUT\_box\_width, INPUT\_box\_height))

pygame.draw.rect(window, grey, (x\_center - (INPUT\_box\_width - 4)/2, y\_center - (INPUT\_box\_height - 4)/2, INPUT\_box\_width - 4, INPUT\_box\_height - 4))

# Text

SET TextSurf, TextRect TO text\_objects(INPUT\_text, smallText, white)

SET TextRect.center TO (x\_center, y\_center)

window.blit(TextSurf, TextRect)

ELSE:

# Box

pygame.draw.rect(window, dark\_grey, (x\_center - INPUT\_box\_width/2, y\_center - INPUT\_box\_height/2, INPUT\_box\_width, INPUT\_box\_height))

# Text

SET TextSurf, TextRect TO text\_objects(INPUT\_text, smallText, white)

SET TextRect.center TO (x\_center, y\_center)

window.blit(TextSurf, TextRect)

# Warning

SET TextSurf, TextRect TO text\_objects("Warning: Text box is not selected", warningText, red)

SET TextRect.center TO (x\_center, y\_center + 40)

window.blit(TextSurf, TextRect)

# Display update

pygame.display.update()

clock.tick(fps)

# Confirm

DEFINE FUNCTION confirm():

global word

global INPUT\_text

IF INPUT\_text EQUALS word:

correct()

ELSE:

incorrect()

# Correct

DEFINE FUNCTION correct():

WHILE True:

# If exit button pressed

FOR event IN pygame.event.get():

IF event.type EQUALS pygame.QUIT:

quit()

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Display correct

SET TextSurf, TextRect TO text\_objects("Correct", largeText, light\_blue)

SET TextRect.center TO (center)

window.blit(TextSurf, TextRect)

# Menu button

button("Menu", 50, 500, 110, button\_height, menu)

# Quit button

button("Quit", 660, 500, 90, button\_height, quit)

# Display update

pygame.display.update()

clock.tick(fps)

# Incorrect

DEFINE FUNCTION incorrect():

global word

WHILE True:

# If exit button pressed

FOR event IN pygame.event.get():

IF event.type EQUALS pygame.QUIT:

quit()

# Mouse position

SET mouse\_position TO pygame.mouse.get\_pos()

SET background\_x TO mouse\_position[0] + background\_default\_x

SET background\_y TO mouse\_position[1] + background\_default\_y

# Background

window.blit(background, (background\_x, background\_y))

# Display incorrect

SET TextSurf, TextRect TO text\_objects("Incorrect", largeText, light\_blue)

SET TextRect.center TO (center)

window.blit(TextSurf, TextRect)

# Display answer

SET TextSurf, TextRect TO text\_objects(f"The correct answer was {word}", smallText, light\_blue)

SET TextRect.center TO (x\_center, y\_center + 50)

window.blit(TextSurf, TextRect)

# Menu button

button("Menu", 50, 500, 110, button\_height, menu)

# Quit button

button("Quit", 660, 500, 90, button\_height, quit)

# Display update

pygame.display.update()

clock.tick(fps)

# Start splashscreen

splashscreen()

# Quit

DEFINE FUNCTION quit():

pygame.quit()

## Program Testing

|  |  |  |
| --- | --- | --- |
| **Input** | **Expected Output** | **Actual Output** |
| Quit button | Quits game | Quits game |
| Start button | Shows difficulty selection screen | Shows difficulty selection screen |
| Instructions button | Shows instructions | Shows instructions |
| Back button | Shows the previous screen | Shows the previous screen |
| Easy button | Timer then shows game screen | Timer then shows game screen |
| Medium button | Timer then shows game screen | Timer then shows game screen |
| Hard button | Timer then shows game screen | Timer then shows game screen |
| Enter button: With correct answer | Shows correct screen | Shows correct screen |
| Enter button: With incorrect answer | Shows incorrect screen | Shows incorrect screen |
| Menu button | Shows menu screen | Shows menu screen |