

MEMORY-PUZZLE GAME OF NUMBER PAIRS USING PYTHON

A project report submitted in partial fulfillment of the requirements

For the award of credits to

Data Preprocessing and Data Visualization

A skill-oriented course of

Bachelor of Technology

In

ELECTRONICS AND COMMUNICATION ENGINEERING

By

BUDATI LEKHA

21BQA0420



DEPARMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

(Approved by AICTE and permanently affiliated to JNTUK)

Accredited by NBA and NAAC with 'A' Grade

NAMBUR (V), PEDAKAKANI (M), GUNTUR- 522508

DECEMBER 2022

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

**VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY: NAMBUR
JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA**



CERTIFICATE

This is to certify that the project titled “ **MEMORY-PUZZLE GAME OF NUMBER PAIRS USING PYTHON** ” is a bonafide record of work done by **Ms. BUDATI LEKHA** under the guidance of **Mrs. T.Vineela** , **ASSISTANT PROFESSOR** in partial fulfillment of the requirement for the award of credits to **Data Preprocessing and Data Visualization** - a skill-oriented course of Bachelor of Technology in Electronics and Communication Engineering, JNTUK during the academic year 2022–23.

Mrs.T.Vineela , Assistant Professor

Course Instructor

Dr.M.Y.Bhanu Murthy

Head of the Depaement

DECLARATION

I **BUDATI LEKHA (21BQ1A0420)** hereby declare that the Project Report entitled “**MEEMORY-PUZZLE GAME OF NUMBER PAIRS**” done by me under the guidance of **T.VINEELA, Assistant Professor, Department of ECE** is submitted in partial fulfillment of the requirements for the award of degree of **BACHELOR OF TECHNOLOGY in ELECTRONICS AND COMMUNICATION ENGINEERING.**

DATE :

SIGNATURE OF THE CANDIDATE

PLACE : VVIT,NAMBUR.

(Budati Lekha)

ACKNOWLEDGEMENT

I express my sincere thanks wherever it is due I express my sincere thanks to the Chairman, Vasireddy Venkatadri Institute of Technology, Sri Vasireddy VidyaSagar for providing me well equipped infrastructure and environment.

I thank Dr. Y. Mallikarjuna Reddy, Principal, Vasireddy Venkatadri Institute of Technology, Nambur, for providing me the resourcesfor carrying out the project.

I express my sincere thanks to Dr. K. Giribabu, Dean of Studies for providing support and stimulating environment fordeveloping the project.

My sincere thanks to Dr. M. Y. Bhanumurthy, Head of the Department, Department of ECE, for his co-operation and guidance which help us to make our project successful and complete in all aspects.

I also express my sincere thanks and are grateful to our guide T.Vineela, Assistant Professor, Department of ECE, for motivating me to make our project successful and fully complete. I am grateful for her precious guidanceand suggestions.

I also place our floral gratitude to all other teaching staff and lab technicians for their constant support and advice throughout the project.

NAME OF THE CANDIDATE

BUDATI LEKHA (21BQ1A0420)

ACKNOWLEDGEMENT

I express my sincere thanks wherever it is due I express my sincere thanks to the Chairman, Vasireddy Venkatadri Institute of Technology, Sri Vasireddy VidyaSagar for providing me well equipped infrastructure and environment.

I thank Dr. Y. Mallikarjuna Reddy, Principal, Vasireddy Venkatadri Institute of Technology, Nambur, for providing me the resourcesfor carrying out the project.

I express my sincere thanks to Dr. K. Giribabu, Dean of Studies for providing support and stimulating environment for developing the project.

My sincere thanks to Dr. M. Y. Bhanumurthy, Head of the Department, Department of ECE, for his co-operation and guidance which help us to make our project successful and complete in all aspects.

I also express my sincere thanks and are grateful to our guide T.Vineela, Assistant Professor, Department of ECE, for motivating me to make our project successful and fully complete. I am grateful for her precious guidanceand suggestions.

I also place our floral gratitude to all other teaching staff and lab technicians for their constant support and advice throughout the project.

NAME OF THE CANDIDATE

BUDATI LEKHA (21BQ1A0420)

CONTENTS

LIST OF FIGURES	
ABSTRACT	
1.INTRODUCTION	1
1.1 STEP BY STEP PROCESS TO INSTALL PYTHON	1
2.MODULES	4
2.1 TURTLE	4
2.2 RANDOM	6
2.3 FREEGAMES	7
3.IMPLEMENTATION	9
3.1 STEPS TO CREATE MEMORY PUZZLE GAME	9
3.2 IMPORTING MODULES	10
3.3 SETTING THE BACKGROUND SCREEN FOR THE GAME	10
3.4 DEFINE A FUNCTION FOR MAKING A SQUARE	11
3.5 DEFINE A FUNCTION TO CHECK THE INDEX NUMBER	11
3.6 FUNCTION TO MAKE THE GAME USER-FRIENDLY	12
3.7 FUNCTION TO DRAW TILES ON THE BASE	14
3.8 USING SHUFFLE() TO SHUFFLE THE NUMBERS	15
4.RESULTS AND CONCLUSION	16
5.REFERENCES	18
APPENDIX	20

LIST OF FIGURES

FIG NO.	TITLE	PAGE NO.
fig 1.1.1	Installation of python 3.10	3
fig 2.1.1	Example to import turtle and print letters	4
fig 2.1.2	Python turtle graphics	5
fig 2.2.1	Importing random in lists	7
fig 2.3.1	Installing freegames module	8
fig 3.6.1	Flowchart of if else condition	13
fig 4.1	Output 1	16
fig 4.2	Output 2	17
fig 4.3	Output 3	17

ABSTRACT

The memory game, or concentration, as it is sometimes called, is a popular card game played by children and adults around the world. Good memory is one of the qualities required in order to succeed in it. This, however, is not enough. When it is assumed that the players have perfect memory, the memory game can be seen as a game of strategy. The game is analysed under this assumption and the optimal strategy is found. It is simple and perhaps unexpected.

In contrast to the simplicity of the optimal strategy, the analysis leading to its optimality proof is rather involved. It supplies an interesting example of concrete mathematics of the sort used in the analysis of algorithms. It is doubtful whether this analysis could have been carried out without resort to experimentation and a substantial use of automated symbolic computations.

Memory games exercise the brain, making it sharp and alert. If you play memory games at least thirty minutes every day, your concentration and focusing ability will improve.

CHAPTER 1

INTRODUCTION

Memory Puzzle Game is the best brain exercise as it helps in enhancing the memory and concentration of the player. It is a popular game. Let's start developing a memory puzzle game in Python and learn some concepts.

In this simple form it's all about memorizing the cards which were flipped. The more cards, the more difficult the game is. It can be played by yourself, but better is to play it against each other. Click to open. Now we can build up and add more levels of difficulty. Instead of finding identical pairs, combinations need to be found.

To develop game in python, first we need to install python

1.1 Step by Step Process to install Python

Here are the steps that you need to follow carefully for installing the Python software on your Windows 10.

Step 1 Choose the Python Version you want to Install

- In the first installation process, download the official installer Python .exe and run it on your system.
- The Python version completely relies on the type of work you want to do on your OS. For example, if you want to complete an already started project build in the 3.10 python version, you need the same. Else, if you have the choice to opt for any version to start your project from scratch.
- If you want to install Python for learning purposes, then it is recommended to install both versions 2 and 3. This is because the 2 versions enable you to work on the old projects and be compatible with new ones.

Step 2 Download and Install Python Executable Installer

- To download the python on your system. You first need to open the web browser on your system. In the Windows section of the official python site, navigate to the downloads.

- Now search for versions of python you need to download. You have choices of python version 3.9.1 for robot framework it is the supported interpreter version
- After choosing the Python version, click on the link to download with executable Window x86-64 or only x86 installer.

Step 3 Complete Installation Process

- Run downloaded Python installer.
- Check you have marked the Install launcher for all your users. In the next step, add python selected version to PATH checkboxes. The last thing is that place the interpreter in the execution path.
- Click on Install and proceed with the installation process.

Pip and IDLE are recommended to install to execute and run your projects for all current or latest python versions. These features are unavailable in the old Python Versions. Next, a dialog box prompts on your screen to select whether you need to extend or disable the path length limit. Selecting this option will enable the programming language to bypass the maximum characters up to 260 in the Max_PATH limit. In simple words, you can use the long path name for your python.

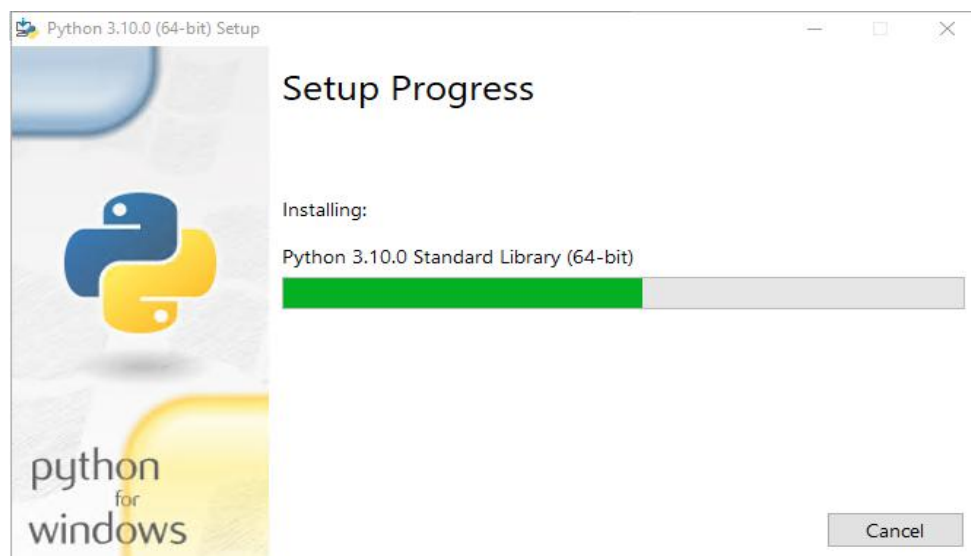


fig.1.1.1 : installing python 3.10

Step 4 Verify the installation of Python on Windows.

- Go to start menu In your laptop or pc
- Search for python 3.10
- Click the enter button and enter some small operations in the interpreter and try to run it .

CHAPTER 2

LIBRARIES AND MODULES

2.1 Turtle :

Turtle is a special features of Python. Using Turtle, we can easily draw in a drawing board. First we import the turtle module. Then create a window, next we create turtle object and using turtle method we can draw in the drawing board. forward() amount It moves the turtle forward by the specified

Turtle backward function and turtle up, down, left right and other functions are used to move the turtle along the canvas to draw patterns along its motion. Python turtle () function is used to create shapes and patterns like this. Use of Python turtle needs an import of Python turtle from Python library.

The turtle module is an extended reimplementation of the same-named module from the Python standard distribution up to version

```
1 import turtle// importing the turtle
2 x = turtle.Turtle()// creating the turtle
3 x.color('deep pink')// color
4 style = ('Courier', 30, 'italic')// font
5 x.write('Hello!', font=style, align='center')//what to write and align
6 x.hideturtle()// hides turtle
7 turtle.done()// ends script
```

Fig 2.1.1: Example to import turtle and print letters

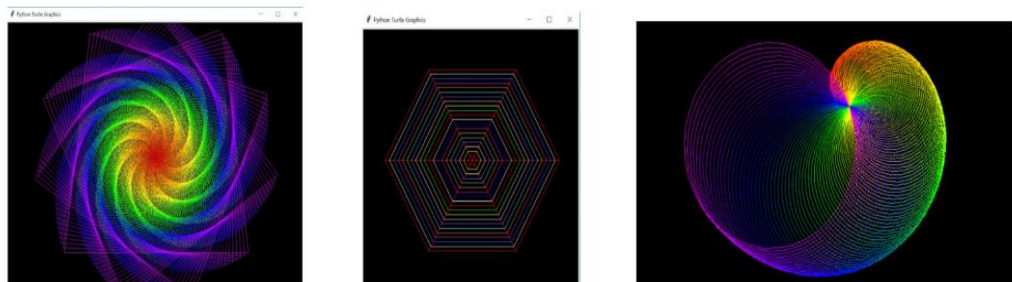
Turtle geometry is also sometimes used in graphics environments as an alternative to a strictly coordinate-addressed graphics system.

Turtle graphics are often associated with the Logo programming language.^[2] Seymour Papert added support for turtle graphics to Logo in the late 1960s to support his version of the turtle robot, a simple robot controlled from the user's workstation that is designed to carry out the drawing functions assigned to it using a small retractable pen set into or attached to the robot's body. Turtle geometry works somewhat differently from (x,y) addressed Cartesian geometry, being primarily vector-based (i.e. relative direction and distance from a starting point) in comparison to coordinate-addressed systems such as bitmaps or raster graphics. As a practical matter, the use of turtle geometry instead of a more traditional model mimics the actual movement logic of the turtle robot. The turtle is traditionally and most often represented pictorially either as a triangle or a turtle icon (though it can be represented by any icon).

Python Turtle Graphics

We will be commencing our Python Programming Journey by using the Python "Turtle" to draw colourful Shapes and Patterns.

We will work through a series of Lessons, getting our programmed drawing skills up to the level where we can create beautiful Rainbow Patterns like these:



HOWEVER..... We need to learn to Crawl, then Walk, then Run and so we will be starting by drawing Lines, Triangles, and Squares, then progressing to Spirals, then finally doing Rainbow Spirals.

Fig.2.1.2 python turtle graphics

2.2 random :

- In Python random is a module that is available in the NumPy library. This module returns an array of specified shapes and fills it with random floats and integers.
- It is based on pseudo-random number generation that means it is a mathematical way that generates a sequence of nearly random numbers
- Basically, it is a combination of a bit generator and a generator.

When we use random module:

We want the computer to pick a random number in a given range Pick a random element from a list, pick a random card from a deck, flip a coin etc. When making your password database more secure or powering a random page feature of your website.

The Random module contains some very useful functions.

Examples:

(i) from random import shuffle

```
x = [[i] for i in range(10)]
```

```
shuffle(x)
```

Output:

```
[[9], [2], [7], [0], [4], [5], [3], [1], [8], [6]]6
```

(ii) import random

```
print random.randint(0, 5)
```

output:

This will output either 1, 2, 3, 4 or 5.

```
[18]: import random
      list = [1,2,3,4,5]
      num = random.sample(list, len(list))
      list

[18]: [1, 2, 3, 4, 5]
```

Fig.2.2.1: importing random in lists

2.3 Freegames:

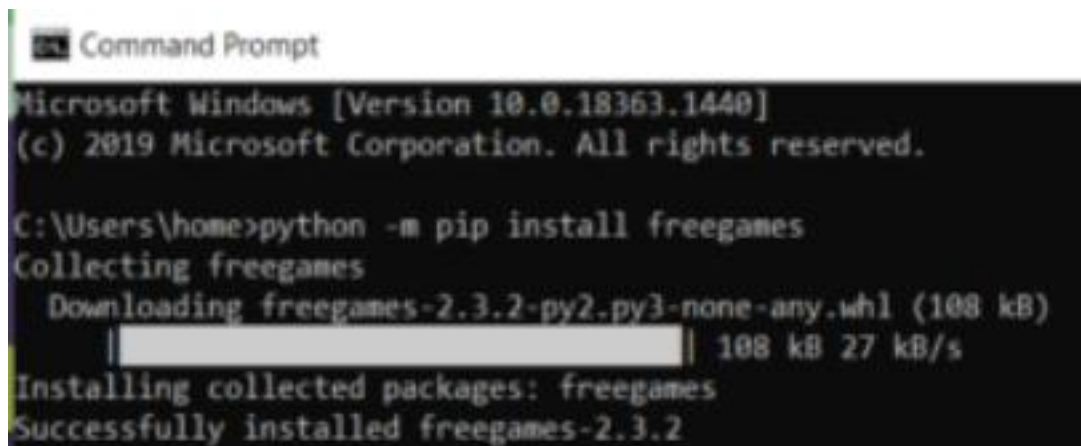
Free Python Games is an Apache2 licensed collection of free Python games intended for education and fun. The games are written in simple Python code and designed for experimentation and changes. Simplified versions of several classic arcade games are included.

In Python language, we get many such modules, with the help of which it is easy for us to do coding. One such module is the FreeGames module. This module is made for education and fun.

Starting in 2012, Free Python Games began as an after school program to teach programming to inner-city youth. Each game is entirely independent from the others and includes comments along with a list of exercises to work through with students. Creativity and flexibility is important. The games run anywhere Python can be installed which includes desktop computers running Windows, Mac OS, or Linux and older or low-power hardware such as the Raspberry Pi.

Installing freegames module :

To install freegames module, go to command prompt and type the following command **python -m pip install freegames** Wait until the installation process gets finished After the process gets finished, you are ready to go.



```
Command Prompt
Microsoft Windows [Version 10.0.18363.1440]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\home>python -m pip install freegames
Collecting freegames
  Downloading freegames-2.3.2-py2.py3-none-any.whl (108 kB)
    |#####| 108 kB 27 kB/s
Installing collected packages: freegames
Successfully installed freegames-2.3.2
```

2.2.2: installing freegames module

CHAPTER 3

IMPLEMENTATION

Memory Puzzle Game is the most popular best memory game or brain exercise. It helps in enhancing the concentration of the player and improves memory. Let's build the Memory Puzzle Game in python.

3.1 Steps to create Memory Puzzle Game

- 1.Import turtle and random module. ...
2. Set the screen and also choose the background color of your output screen window.
3. Define a function for making a square for the base of your game.
4. Define a function to keep a check of the index number.
5. Define a function to make your game user-friendly i.e user click.
6. Write a function to draw tiles on the square base
7. Finally use the shuffle() function to shuffle the numbers placed on the square tiles in the square box.
- .

3.2 Importing Modules

```
from random import *
```

```
from turtle import *
```

```
from freegames import path
```

Explanation:

Random module – Random module is an in-built module of Python which is to generate random words from list[].

Turtle - turtle is a pre-installed Python library that enables users to create pictures and shapes by providing them with a virtual canvas.

3.3 Setting the background screen for the game

```
car = path('car.gif')
tiles = list(range(32)) * 2
state = {'mark': None}
hide = [True] * 64
```

Explanation:

Here we set the background screen a car image using turtle and using lists we set the number of tiles in the game.

List - Lists are used to store multiple items in a single variable. Lists are one of 4 built-in data types in Python used to store collections of data.

Range - Using range The range () function returns a sequence of numbers, starting from 0 by default, and increments by 1 ending at a specified number.

3.4 Define a function for making a square for the base of game

```
def square(x, y):
    """Draw white square with black outline at (x, y)."""
    up()
    goto(x, y)
    down()
    color('black', 'white')
    begin_fill()
    for count in range(4):
        forward(50)
        left(90)
    end_fill()
```

Explanation:

In this we define a square function to draw the white square with black outline with x and y as coordinates, we change the colours of the outline and squares.

We also goto , A goto statement is a piece of code or syntax that can jump from the goto statement to a labeled statement which is marked as end within the same function.

For loop is also used, A **for** loop is used for iterating over a sequence. With the **for** loop we can execute a set of statements, once for each item in a list, tuple, set etc.

3.5 Define a function to keep a check of the index number.

```
def index(x, y):  
    """Convert (x, y) coordinates to tiles index."""  
    return int((x + 200) // 50 + ((y + 200) // 50) * 8)  
  
def xy(count):  
    """Convert tiles count to (x, y) coordinates."""  
    return (count % 8) * 50 - 200, (count // 8) * 50 - 200
```

Explanation:

In this step we define a function to convert x and y coordinates to tiles index and then we define function to count the number of tiles

Index () - The index () method finds the first occurrence of the specified value. The index () method raises an exception if the value is not found and it also helps to find the index position of an element or an item in a string of characters or a list of items.

3.6 Define a function to make your game user-friendly

```
def tap(x, y):
    """Update mark and hidden tiles based on tap."""
    spot = index(x, y)
    mark = state['mark']

    if mark is None or mark == spot or tiles[mark] != tiles[spot]:
        state['mark'] = spot
    else:
        hide[spot] = False
        hide[mark] = False
        state['mark'] = None
```

Explanation:

In this step , we define a function tap() which is user friendly i.e.,user click. This function helps to mark the hidden tiles based on the tap by the user.

def **tap** (x, f): f (x) return x Usage: >>> **tap** ([], lambda x: x.append (1)) [1] However it won't be so much use in **Python** 2.x as it is in Ruby because **lambda functions in Python** are quite restrictive. For example you can't inline a call to print because it is a keyword, so you can't use it for inline debugging code.

Example:

```
>>> tap(2, lambda x: print(x)) + 3
```

```
2
5
```

If – else : Decision making is the most important aspect of almost all the programming languages. As the name implies, decision making allows us to run a particular block of code for a particular decision. Here, the decisions are made on the validity of the particular conditions. Condition checking is the backbone of decision making.

The if-else statement is similar to if statement except the fact that, it also provides the block of the code for the false case of the condition to be checked. If the

condition provided in the if statement is false, then the else statement will be executed.

12

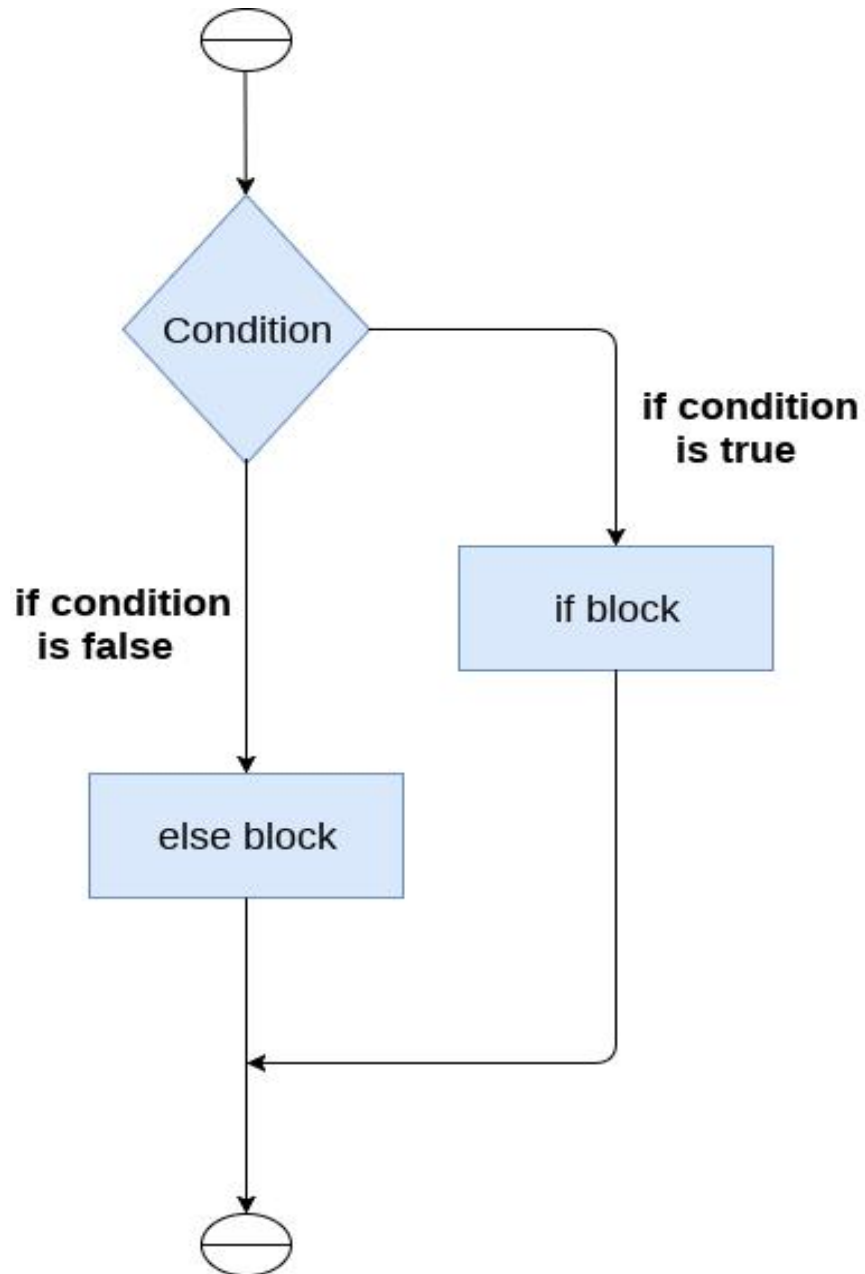


Fig.3.6.1: Flowchart for if else statement

3.7 Write a function to draw tiles on the square base

```
def draw():
    """Draw image and tiles."""
    clear()
    goto(0, 0)
    shape(car)
    stamp()

    for count in range(64):
        if hide[count]:
            x, y = xy(count)
            square(x, y)

    mark = state['mark']

    if mark is not None and hide[mark]:
        x, y = xy(mark)
        up()
        goto(x + 2, y)
        color('black')
        write(tiles[mark], font=('Arial', 30, 'normal'))

    update()
    ontimer(draw, 100)
```

Explanation:

In this step, we define a function to draw() to draw the image and tiles for the game and here we give the count of the tiles in the range 64.

Clear() - The clear () method removes all items from the dictionary. The clear () method doesn't take any parameters. The clear () method doesn't return any value.

As we are not passing any parameters there is no chance for any error.

Update() - The update () method inserts the specified items to the dictionary. The specified items can be a dictionary, or iterable object with key value pairs.

If statement - The if statement is used to test a particular condition and if the condition is true, it executes a block of code known as if-block. The condition of if statement can be any valid logical expression which can be either evaluated to true or false.

14

3.8 Using the shuffle() function to shuffle the numbers

```
shuffle(tiles)
setup(420, 420, 370, 0)
addshape(car)
hideturtle()
tracer(False)
onscreenclick(tap)
draw()
done()
```

Explanation:

Finally, we use the shuffle statement to shuffle the numbers placed on the square tiles in the square box.

shuffle() - The **shuffle()** is an inbuilt method of the random module. It is used to shuffle a sequence (list). Shuffling a list of objects means changing the position of the elements of the sequence using Python.

CHAPTER 4

RESULTS AND CONCLUSIONS

Memory – puzzle game of number pairs is a memory game where you need to match pairs of identical numbers, by matching all the number pairs the image behind the tiles will be revealed.

The output is as follows:

1. All the tiles are closed.

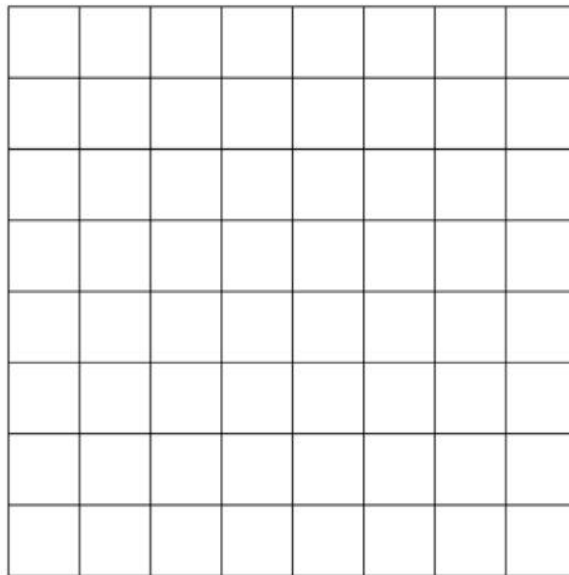


Fig.4.1: output 1

2. By matching the numbers the tiles are opened.

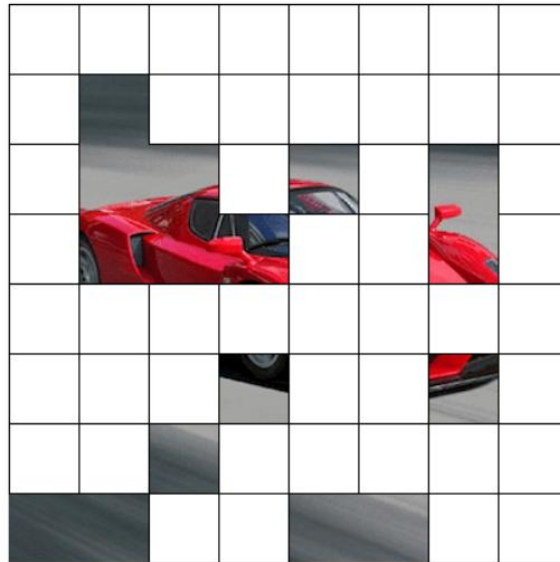


Fig.4.2: output 2

3. The image is revealed by opening the all tiles.

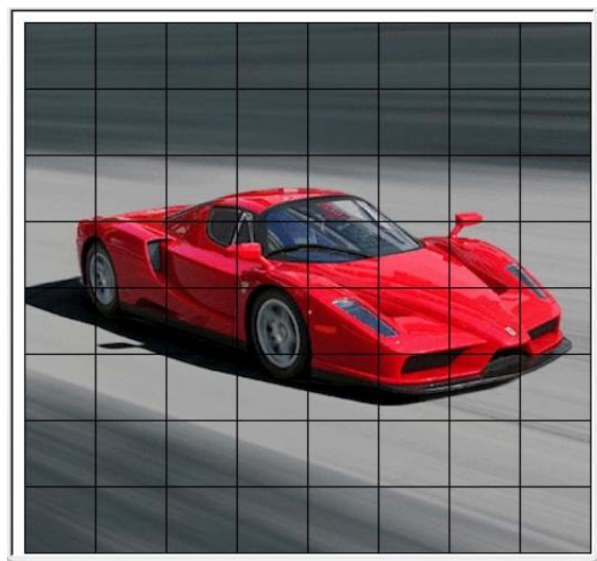


Fig.4.3: output 3

REFERENCES

1. <https://www.igi-global.com/journal/international-journal-gaming-computer-mediated/1125>
2. <https://www.researchgate.net/journal/The-Computer-Games-Journal-2052-773X>
3. <https://www.javatpoint.com/top-python-frameworks-for-gaming>
4. <https://www.upgrad.com/blog/python-game-projects-topics>
5. <https://www.gamedesigning.org/learn/python>
6. <https://theconversation.com/us/topics/gaming>
7. <https://github.com/anthonymonori/python-minigames>
8. <https://www.hindawi.com/journals/ijcg>

APPENDIX

```
from random import *
from turtle import *
from freegames import path

car = path('car.gif')
tiles = list(range(32)) * 2
state = {'mark': None}
hide = [True] * 64

def square(x, y):
    """Draw white square with black outline at (x, y)."""
    up()
    goto(x, y)
    down()
    color('black', 'white')
    begin_fill()
    for count in range(4):
        forward(50)
        left(90)
    end_fill()

def index(x, y):
    """Convert (x, y) coordinates to tiles index."""
    return int((x + 200) // 50 + ((y + 200) // 50) * 8)
```

```

def xy(count):
    """Convert tiles count to (x, y) coordinates."""
    return (count % 8) * 50 - 200, (count // 8) * 50 - 200

def tap(x, y):
    """Update mark and hidden tiles based on tap."""
    spot = index(x, y)
    mark = state['mark']

    if mark is None or mark == spot or tiles[mark] != tiles[spot]:
        state['mark'] = spot
    else:
        hide[spot] = False
        hide[mark] = False
        state['mark'] = None

def draw():
    """Draw image and tiles."""
    clear()
    goto(0, 0)
    shape(car)
    stamp()
    for count in range(64):
        if hide[count]:
            x, y = xy(count)
            square(x, y)
    mark = state['mark']

```

```
if mark is not None and hide[mark]:  
    x, y = xy(mark)  
    up()  
    goto(x + 2, y)  
    color('black')  
    write(tiles[mark], font=('Arial', 30, 'normal'))
```

```
update()  
ontimer(draw, 100)
```

```
shuffle(tiles)  
setup(420, 420, 370, 0)  
addshape(car)  
hideturtle()  
tracer(False)  
onscreenclick(tap)  
draw()  
done()
```