

PYTHON PROGRAMMING
INTERNSHIP
PROJECT BY:
SAHIL PRASAD
TASK 3 : MEMORY PUZZLE GAME

CODE:

```
import tkinter as tk
from tkinter import messagebox
import random

class MemoryGame:
    def __init__(self, root):
        self.root = root
        self.root.title("Memory Puzzle Game")

        self.time_limit = 60 # Total time in seconds
        self.time_left = self.time_limit
        self.first_card = None
        self.second_card = None
```

```
self.lock = False  
self.buttons = []  
self.matches_found = 0  
  
# Symbols to be displayed on the cards  
self.symbols = ['🍏', '🍎', '🍇', '🍈', '🍉', '🍓', '🥝',  
'🍅', '🍅']  
self.cards = self.symbols * 2  
random.shuffle(self.cards)  
  
# Timer display  
self.timer_label = tk.Label(root, text=f"Time  
Left: {self.time_left}s", font=("Arial", 14))  
self.timer_label.grid(row=0, column=0,  
columnspan=4, pady=10)  
  
self.create_board()  
self.start_timer()  
  
# Restart button
```

```
restart_button = tk.Button(root,
text="Restart", command=self.restart_game,
font=("Arial", 12))

    restart_button.grid(row=5, column=1,
columnspan=2, pady=10)

def create_board(self):
    """Creates a 4x4 grid of buttons (cards)."""
    for i in range(4):
        for j in range(4):
            button = tk.Button(
                self.root,
                text="",
                font=("Arial", 24),
                width=4,
                height=2,
                command=lambda idx=(i * 4 + j):
self.flip_card(idx)
            )
            button.grid(row=i + 1, column=j, padx=5,
pady=5)
```

```
    self.buttons.append(button)

def flip_card(self, idx):
    """Handles card flip action."""
    if self.lock or self.buttons[idx]['state'] == 'disabled':
        return

    self.buttons[idx].config(text=self.cards[idx])

    if not self.first_card:
        self.first_card = idx
    else:
        self.second_card = idx
        self.check_match()

def check_match(self):
    """Checks if two flipped cards match."""
    self.lock = True
    if self.cards[self.first_card] ==
self.cards[self.second_card]:
```

```
        self.buttons[self.first_card]['state'] =  
    'disabled'  
        self.buttons[self.second_card]['state'] =  
    'disabled'  
        self.matches_found += 1  
  
    if self.matches_found == len(self.cards) //  
2:  
        messagebox.showinfo("Victory", "🎉  
You matched all cards in time!")  
        self.restart_game()  
        self.reset_selection()  
    else:  
        self.root.after(800, self.hide_cards)  
  
def hide_cards(self):  
    """Hides the symbols of unmatched cards."""  
    self.buttons[self.first_card].config(text="")  
    self.buttons[self.second_card].config(text="")  
    self.reset_selection()
```

```
def reset_selection(self):
    """Resets selected cards and unlocks the board."""
    self.first_card = None
    self.second_card = None
    self.lock = False

def start_timer(self):
    """Starts the countdown timer."""
    if self.time_left > 0:
        self.time_left -= 1
        self.timer_label.config(text=f"Time Left: {self.time_left}s")
        self.root.after(1000, self.start_timer)
    else:
        messagebox.showwarning("Time's Up!",
"⌚ You ran out of time!")
        self.disable_all()

def disable_all(self):
    """Disables all cards when time is up."""
```

```

    for btn in self.buttons:
        btn['state'] = 'disabled'

def restart_game(self):
    """Restarts the game."""
    random.shuffle(self.cards)
    self.time_left = self.time_limit
    self.timer_label.config(text=f"Time Left: {self.time_left}s")
    self.matches_found = 0
    self.reset_selection()

for btn in self.buttons:
    btn.config(text="", state='normal')

self.start_timer()

# Run the game
if __name__ == "__main__":
    root = tk.Tk()
    game = MemoryGame(root)

```

root.mainloop()

Output

The screenshot shows the Visual Studio Code interface with the following details:

- Explorer:** Shows "NO FOLDER OPENED".
- Terminal:** Displays Python code for a memory game and its execution in a terminal window.
- Editor:** Shows the "memory_game.py" file with code for a Memory Game.
- Output:** Shows the game's progress in a terminal window.
- Code Editor:** Shows the Python code for the Memory Game, including imports, class definition, and a 4x4 grid of symbols.
- Taskbar:** Shows various system icons and the date/time (28-09-2025).

The screenshot shows the Visual Studio Code interface with the following details:

- Explorer:** Shows "NO FOLDER OPENED".
- Terminal:** Displays Python code for a memory game and its execution in a terminal window.
- Editor:** Shows the "memory_game.py" file with code for a Memory Game.
- Output:** Shows the game's progress in a terminal window.
- Code Editor:** Shows the Python code for the Memory Game, including imports, class definition, and a 4x4 grid of symbols.
- Taskbar:** Shows various system icons and the date/time (28-09-2025).

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows "NO FOLDER OPENED".
- Search Bar:** Shows "Search".
- Editor:** Displays the file "memory_game.py" with the following code:

```
C:\> Users > sahil > OneDrive > Desktop > 🎁 memory_game.py > ...
5  class MemoryGame:
33
34      """Creates a 4x4 grid of buttons (cards)."""
35      for i in range(4):
36          for j in range(4):
37              button = tk.Button(
38                  self.root,
39                  text='',
40                  font=("Arial", 24),
41                  width=4,
42                  height=2,
43                  command=lambda idx=(i * 4 + j): self.flip_card(idx)
44              )
45              button.grid(row=i + 1, column=j, padx=5, pady=5)
46              self.buttons.append(button)
47
48      def flip_card(self, idx):
49          """Handles card flip action."""
50          if self.lock or self.buttons[idx]['state'] == 'disabled':
51              return
52
53          self.buttons[idx].config(text=self.cards[idx])
54
55          if not self.first_card:
56              self.first_card = idx
57          else:
58              self.second_card = idx
59              self.check_match()
60
61      def check_match(self):
62          """Checks if two flipped cards match."""
63
```

The status bar at the bottom indicates: Ln 122, Col 1, Spaces: 4, UTF-8, CRLF, Python, 3.13.1, Go Live, 19:57, ENG IN, 28-09-2025.

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows "NO FOLDER OPENED".
- Search Bar:** Shows "Search".
- Editor:** Displays the file "memory_game.py" with the following code:

```
C:\> Users > sahil > OneDrive > Desktop > 🎁 memory_game.py > ...
5  class MemoryGame:
62      def check_match(self):
63          self.lock = True
64          if self.cards[self.first_card] == self.cards[self.second_card]:
65              self.buttons[self.first_card]['state'] = 'disabled'
66              self.buttons[self.second_card]['state'] = 'disabled'
67              self.matches_found += 1
68
69          if self.matches_found == len(self.cards) // 2:
70              messagebox.showinfo("Victory", "🎉 You matched all cards in time!")
71              self.restart_game()
72              self.reset_selection()
73          else:
74              self.root.after(800, self.hide_cards)
75
76      def hide_cards(self):
77          """Hides the symbols of unmatched cards."""
78          self.buttons[self.first_card].config(text='')
79          self.buttons[self.second_card].config(text='')
80          self.reset_selection()
81
82      def reset_selection(self):
83          """Resets selected cards and unlocks the board."""
84          self.first_card = None
85          self.second_card = None
86          self.lock = False
87
88      def start_timer(self):
89          """Starts the countdown timer."""
90          if self.time_left > 0:
91              self.time_left -= 1
92              self.timer_label.config(text=f"Time left: {self.time_left}s")
93
```

The status bar at the bottom indicates: Ln 122, Col 1, Spaces: 4, UTF-8, CRLF, Python, 3.13.1, Go Live, 19:57, ENG IN, 28-09-2025.

The screenshot shows a code editor interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Run, ...
- Search Bar:** Search
- Explorer:** NO FOLDER OPENED. Sub-sections: Open Folder, Create Java Project.
- Editor Area:** Two tabs: Welcome and memory_game.py. The code is as follows:

```
C:\Users\sahil>OneDrive>Desktop>memory_game>...
5  class MemoryGame:
89     def start_timer(self):
94         self.root.after(1000, self.start_timer)
95     else:
96         messagebox.showwarning("Time's Up!", "You ran out of time!")
97         self.disable_all()
98
99     def disable_all(self):
100        """Disables all cards when time is up."""
101        for btn in self.buttons:
102            btn['state'] = 'disabled'
103
104    def restart_game(self):
105        """Restarts the game."""
106        random.shuffle(self.cards)
107        self.time_left = self.time_limit
108        self.timer_label.config(text=f"Time Left: {self.time_left}s")
109        self.matches_found = 0
110        self.reset_selection()
111
112        for btn in self.buttons:
113            btn.config(text=' ', state='normal')
114
115        self.start_timer()
116
117    # Run the game
118    if __name__ == "__main__":
119        root = tk.Tk()
120        game = MemoryGame(root)
121        root.mainloop()
122
```

Bottom status bar: Ln 122, Col 1 | Spaces: 4 | UTF-8 | CRLF | Python | 3.13.1 | Go Live | 19:57 | 28-09-2025