Task - 1

Team Members

- *Rawan Ibrahim – 20201701717*
- *Mariam Alaa - 20201701739*
- *Ali Shaker – 20201701725*
- *Diaa Amr – 20201701720*

Supervised by :

- *TA / Hazim Youssef*
- *TA / Mariam Hesham*

Table of Contents

1. Introduction:.....1
2. Functions.....1
   2.1. “display_cards” Function.....1
   2.2. “display_card_in_frame” Function.....1
   2.3. “compare_cards” Function.....2
  2
   2.4. "save_player_cards" Function.....2
   2.5. "display_card" Function.....3
   2.6. "create_center_frame" Function.....4
   2.7. "remove_cards" Function.....4
   2.8. "display_initial_setup" Function.....5
   2.9. "on_confirm_player1" and "on_confirm_player2" Functions.....5
   2.10. "play_again" Function.....6
   2.11. "quit_playing" Function.....6
   2.12. "start_game" Function.....7
   2.13. "display_winner_message" Function.....7
   2.14. "update_scoreboard" Function.....8
   2.15. "refresh_screen" Function.....8
   2.16. "confirm_and_display_player1" and "confirm_and_display_player2" Functions.....9
   2.17. "swap_cards" Function.....9
   2.18. "update_card_positions" Function.....9
   2.19. "display_ordered_cards" Function.....10
   2.20. "display_instructions" Function.....10
3. Final Product.....11
4. Payoff Matrix.....12

1. Introduction:

The given project represents a simulation of the "Simultaneous QUATRO-UNO" card game. It allows two players to compete against each other by strategically selecting piles of cards and aiming to be the last one with cards remaining. The game dynamics involve revealing cards simultaneously, removing lower-valued cards, and ultimately determining the winner based on the remaining cards. The code utilizes the Tkinter library for the graphical user interface (GUI) and implements game logic using Python programming.

2. Functions

2.1. "display_cards" Function

```
# Function to display cards for a player with animation
def display_cards(player, card_paths, start_x, delay_offset):
    for i, path in enumerate(card_paths):
        delay = i * delay_offset  # Adjust timing for each player
        display_card(player, path, start_x + i * 75, delay, i)
```

Description: Displays cards for a player with animation.

Usage: display_cards(player, card_paths, start_x, delay_offset)

Inputs:

- player: Player number (1 or 2).
- card_paths: List of file paths to card images.
- start_x: Starting x-coordinate for card display.
- delay_offset: Offset for timing of card animations.

Output: None

2.2. "display_card_in_frame" Function

```
def display_card_in_frame(frame, image_path, card_index):
    frame_width = 160  # Assuming the frame's width is 160 pixels
    card_width = 60
    spacing = 20  # Spacing between cards
    total width = 2 * card width + spacing
```

Description: Displays a card image within a frame at a specified position.

Usage: display_card_in_frame(frame, image_path, card_index)

Input:

- frame: Frame widget to display the card in.
- image_path: File path to the card image.
- card_index: Index of the card in the frame.

Output: None

2.3. “compare_cards” Function

```
def compare_cards():
    global player1_cards, player2_cards
```

Description: Compares the cards of both players and updates the game accordingly.

Usage: compare_cards()

Input: None

Output: None

2.4. "save_player_cards" Function

```
def save_player_cards(player_cards, player_name, button):
```

Description: Saves the positions of player's cards displayed on the GUI when the "Ready" button is clicked. Additionally, it toggles the color of the "Ready" button between grey and green.

Usage: save_player_cards(player_cards, player_name, button)

Input:

- player_cards: List containing tuples of card labels and their corresponding image paths.
- player_name: Name of the player (e.g., "Player 1", "Player 2").
- button: Tkinter Button widget representing the "Ready" button for the player.

Output: None

Internal Variables:

- card_data: Dictionary containing the positions of each card displayed on the GUI.
- index: Index counter for enumerating through the player's cards.
- card_name: Name of the card image file.
- position: Tuple containing the x and y coordinates of the card label.
- bg_color: Background color of the "Ready" button.

Additional Notes:

- The positions of the cards are stored as (x, y) coordinates relative to the GUI window.
- The button's background color toggles between grey and green to indicate whether the player has confirmed their card arrangement.

2.5. "display_card" Function

```
# Function to load, resize, and display a card image at a specified position
def display_card(player, image_path, x, delay, i):
    def load_and_place():
```

Description: Loads, resizes, and displays a card image at a specified position on the GUI. Additionally, it creates arrow buttons for swapping cards and places them below the displayed card.

Usage: display_card(player, image_path, x, delay, i)

Input:

- player: Player number (1 or 2).
- image_path: File path to the card image.
- x: x-coordinate for the position of the card.
- delay: Delay time (in milliseconds) before displaying the card.
- i: Index of the card in the player's hand.

Output: None

Internal Functions:

- load_and_place: Loads the card image, resizes it, creates arrow buttons, and places them below the card.

Internal Variables:

- image: Loaded card image from the specified file path.
- image_resized: Resized version of the card image to fit the GUI.
- photo: Tkinter PhotoImage object to display the resized card image.
- label: Tkinter Label widget to display the card image.
- left_arrow: Tkinter Button widget for the left arrow to swap cards.
- right_arrow: Tkinter Button widget for the right arrow to swap cards.
- button_width: Width of each arrow button.
- button_spacing: Spacing between the arrow buttons.
- total_button_width: Total width of both buttons and spacing.
- button_x: x-coordinate for the position of the left arrow button to center both buttons.
- card_info: Tuple containing the label widget and the image path of the displayed card, stored for reference.

Additional Notes:

- Arrow buttons allow players to swap the displayed card with adjacent cards in their hands.
- Card swapping functionality is implemented using the swap_cards function.

2.6. "create_center_frame" Function

```
def create_center_frame():
    center_frame = tk.Frame(master, bg="white", width=160, height=160)
    center_frame.place(x=420, y=250)
    return center_frame
```

Description: Creates a center frame on the GUI to display cards for comparison during gameplay.

Usage: create_center_frame()

Input: None

Output: Tkinter Frame widget representing the center frame.

Additional Notes:

- The center frame is white with dimensions 160x160 pixels.
- The frame is positioned at coordinates (420, 250) on the GUI window.

2.7. "remove_cards" Function

```
def remove_cards(center_frame, card_player1_value, card_player2_value):
    global player1_cards, player2_cards
```

Description: Removes cards from the players' hands based on the comparison of card values during gameplay. Additionally, it updates the GUI to reflect the changes after card removal.

Usage: remove_cards(center_frame, card_player1_value, card_player2_value)

Input:

- center_frame: Tkinter Frame widget representing the center frame where cards are displayed for comparison.
- card_player1_value: Numerical value of the card held by Player 1.
- card_player2_value: Numerical value of the card held by Player 2.

Output: None

Internal Variables:

- player1_cards, player2_cards: Global lists containing tuples of card labels and their corresponding image paths for Player 1 and Player 2, respectively.

Additional Notes:

- Cards are removed from the players' hands based on the comparison of their values.
- The GUI is updated to reflect the changes in the players' card arrangements after removal.

2.8. "display_initial_setup" Function

```
def display_initial_setup():
    # Display title and player labels
```

Description: Displays the initial setup of the game interface, including the title, player labels, instructions, and cards for both players. It also handles the functionality of the "Ready" buttons for each player and the "Start Game" button.

Usage: display_initial_setup()

Input: None

Output: None

Internal Functions:

- display_instructions(): Displays instructions for both players.
- display_cards(player, card_paths, start_x, delay_offset): Displays cards for the specified player with animation.
- on_confirm_player1(button): Callback function for the "Ready" button of Player 1.
- on_confirm_player2(button): Callback function for the "Ready" button of Player 2.

Additional Notes:

- The title and player labels are displayed using Tkinter Label widgets.
- "Ready" buttons are displayed for both players to confirm their card arrangements.
- The "Start Game" button is displayed after both players have confirmed their card arrangements.

2.9. "on_confirm_player1" and "on_confirm_player2" Functions

```
def on_confirm_player1(button):
```

```
def on_confirm_player2(button):
```

Description: These functions handle the actions triggered when the "Ready" buttons for Player 1 and Player 2 are clicked, respectively. They save the player's card arrangements and conditionally display the "Start Game" button based on both players' readiness.

Usage:

- on_confirm_player1(button): Callback function for Player 1's "Ready" button.
- on_confirm_player2(button): Callback function for Player 2's "Ready" button.

Input:

- button: Tkinter Button widget representing the "Ready" button for the player.

Output: None

Internal Functions:

- save_player_cards(player_cards, player_name, button): Saves the positions of player's cards displayed on the GUI when the "Ready" button is clicked. Additionally, it toggles the color of the "Ready" button between grey and green.
- start_game(): Initiates the gameplay once both players have confirmed their card arrangements.

Additional Notes:

- These functions are assigned as callback functions to the respective "Ready" buttons for each player.
- They trigger the saving of player's card positions and conditionally display the "Start Game" button when both players are ready.

2.10. "play_again" Function

```
def play_again():
    global player1_cards, player2_cards, player1_score, player2_score, play_again_pressed
    play_again_pressed = True
```

Description: Resets the game state for a new round, including clearing player cards, displaying the initial setup, removing winner announcement labels, and restarting the game.

Usage: play_again()

Input: None

Output: None

Internal Functions:

- refresh_screen(): Refreshes the game screen.
- display_initial_setup(): Displays the initial setup of the game.
- update_scoreboard(): Updates the scoreboard with the current scores.
- start_game(): Initiates the game.

Additional Notes:

- Resets player cards, scoreboard, and sets the play_again_pressed flag to True.
- Destroys all existing widgets on the screen and redisplays the initial setup.
- Removes winner announcement labels from the previous round.
- Restarts the game by calling the start_game() function.

2.11. "quit_playing" Function

Description: Closes all windows and exits the game.

Usage: quit_playing()

Input: None

Output: None

Additional Notes:

- Destroys the master window, effectively closing the game.

2.12. "start_game" Function

```
def start_game():
    refresh_screen()
    confirm_and_display_player1()
    confirm_and_display_player2()
    compare_cards()
    update_scoreboard()
```

Description: Initiates the game by refreshing the screen, confirming and displaying cards for both players, comparing the cards, and updating the scoreboard.

Usage: start_game()

Input: None

Output: None

Internal Functions:

- refresh_screen(): Refreshes the game screen.
- confirm_and_display_player1(): Confirms and displays cards for Player 1.
- confirm_and_display_player2(): Confirms and displays cards for Player 2.
- compare_cards(): Compares the cards of both players to determine the winner.
- update_scoreboard(): Updates the scoreboard with the current scores.

Additional Notes:

- Initiates the game flow by executing various game-related functions.

2.13. "display_winner_message" Function

```
# Function to display winner message and update scoreboard
def display_winner_message(winner_message):
    global player1_score, player2_score, player1_score_label, player2_score_label
```

Description: Displays the winner message, updates the scoreboard, and provides options to play again or quit playing.

Usage: display_winner_message(winner_message)

Input:

- winner_message: String containing the message indicating the winner.

Output: None

Additional Notes:

- Displays the winner message and updates the scores based on the winner.
- Displays the scoreboard with the updated scores.
- Provides options to play again or quit playing with respective buttons.

2.14. "update_scoreboard" Function

```
# Function to update the scoreboard
def update_scoreboard():
    global player1_score, player2_score
    player1_score_label.config(text=f"Player 1: {player1_score}")
    player2_score_label.config(text=f"Player 2: {player2_score}")
```

Description: Updates the scoreboard with the current scores for both players.

Usage: update_scoreboard()

Input: None

Output: None

Internal Variables:

- player1_score_label: Label displaying Player 1's score.
- player2_score_label: Label displaying Player 2's score.

Additional Notes:

- Updates the text of player1_score_label and player2_score_label with the current scores.

2.15. "refresh_screen" Function

```
def refresh_screen():
    # Clear the screen
    for widget in master.winfo_children():
        widget.destroy()
```

Description: Clears the screen, displays the game board title, and creates frames for displaying ordered cards for both players.

Usage: refresh_screen()

Input: None

Output: None

Internal Functions:

- display_ordered_cards(frame, player): Displays ordered cards within a given frame for the specified player.

Additional Notes:

- Clears all existing widgets from the screen.
- Displays the "Game Board" title.
- Creates frames for displaying ordered cards for both Player 1 and Player 2.
- Calls display_ordered_cards to populate the frames with ordered cards.

2.16. "confirm_and_display_player1" and "confirm_and_display_player2" Functions

Description: Confirms and displays ordered cards for Player 1 and Player 2, respectively.

Usage:

- confirm_and_display_player1()
- confirm_and_display_player2()

Input: None

Output: None

Additional Notes:

- Extracts the paths of ordered cards for the respective players.
- Updates the display to show the confirmed and ordered cards for the respective players.

2.17. "swap_cards" Function

Description: Swaps the position of a card with the adjacent card in the player's hand.

Usage:

- swap_cards(player, card_index, direction)

Input:

- player: Integer representing the player (1 or 2).
- card_index: Index of the card to be swapped.
- direction: Integer representing the direction of swapping (-1 for left, 1 for right).

Output: None

Additional Notes:

- Updates the positions of cards in the player's hand after swapping.
- Only swaps cards within the bounds of the player's hand.

2.18. "update_card_positions" Function

Description: Updates the positions of cards in the player's hand after swapping.

Usage: update_card_positions(player)

Input: player: Integer representing the player (1 or 2).

Output: None

Additional Notes:

- Calculates the new x-coordinate for each card label based on the player's hand.
- Repositions each card label accordingly.

2.19. "display_ordered_cards" Function

Description: Displays ordered cards for each player within a given frame.

Usage: display_ordered_cards(frame, player)

Input:

- frame: Frame object where the cards will be displayed.
- player: Integer representing the player (1 or 2).

Output: None

Additional Notes:

- Displays ordered cards for the specified player within the provided frame.
- Extracts the paths of ordered cards for the specified player and displays them in the frame.

2.20. "display_instructions" Function

Description: Displays instructions for ordering cards after all card animations.

Usage: display_instructions()

Input: None

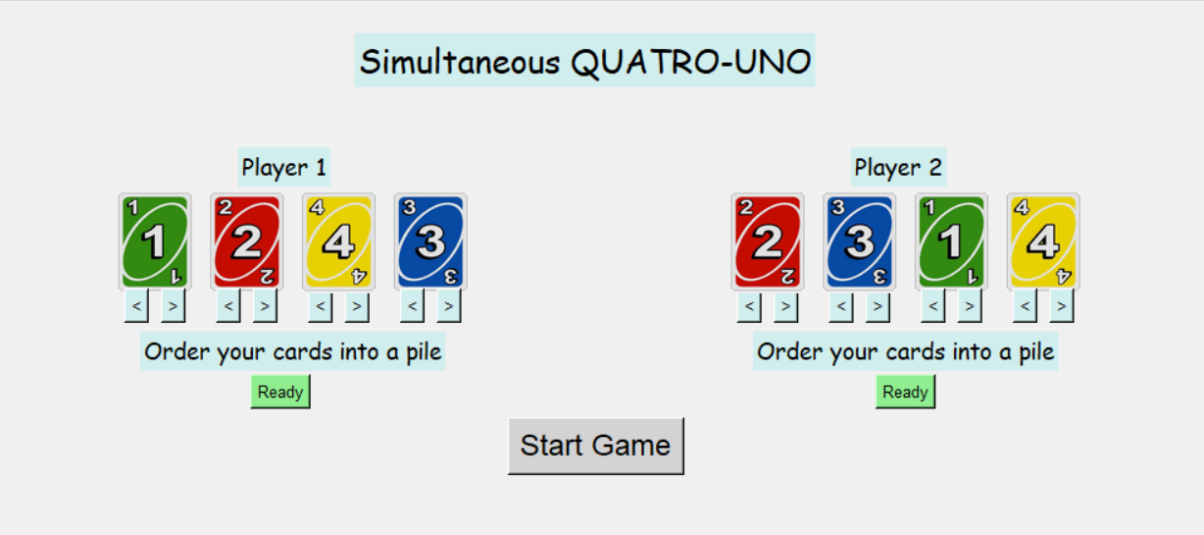
Output: None

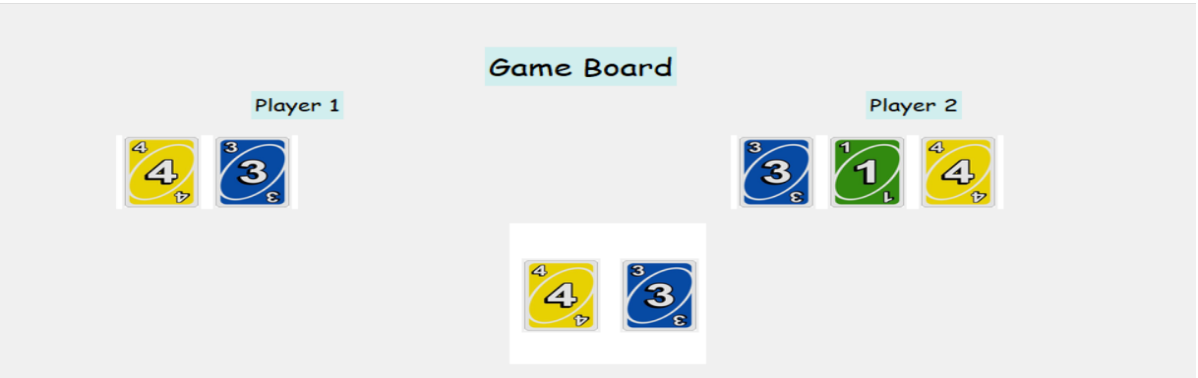
Additional Notes:

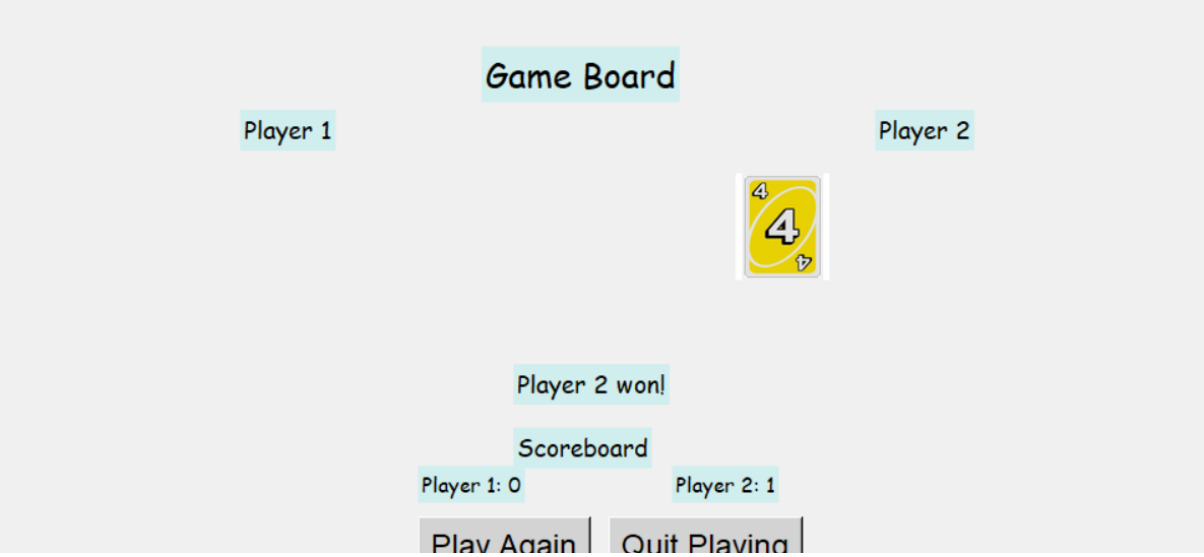
- Displays instructions for both players to order their cards into a pile.
- Instructions are displayed on the GUI after all card animations have finished.

3. Final Product

Sample Run:







4. Payoff Matrix

	A	B	C	D	E
1		1	2	3	4
2	1	0	1	-1	0
3	2	-1	0	1	-1
4	3	1	-1	0	1
5	4	0	1	-1	0