```
# Créez le plateau pour commencer le jeu de Tic Tac Toe
def create_board():
    return [" " for _ in range(9)]

# Affiche le plateau de jeu
def display_board(board):
    for row in [board[i:i+3] for i in range(0, 9, 3)]:
        print(" | ".join(row))
        print("-" * 9)

# Vérifie si un joueur a gagné
def check_winner(board, player):
    winning_combinations = [(0, 1, 2), (3, 4, 5), (6, 7, 8),
                            (0, 3, 6), (1, 4, 7), (2, 5, 8),
                            (0, 4, 8), (2, 4, 6)]
    for combo in winning_combinations:
        if all(board[i] == player for i in combo):
            return True
    return False

# Joue le jeu
def play_game():
    board = create_board()
    current_player = "X"
    while True:
        display_board(board)
```

```
        position = int(input(f"Joueur {current_player}, choisissez une position (1-9) :
")) - 1
        if board[position] == " ":
            board[position] = current_player
            if check_winner(board, current_player):
                display_board(board)
                print(f"Joueur {current_player} a gagné !")
                break
            elif " " not in board:
                display_board(board)
                print("Match nul !")
                break
            current_player = "O" if current_player == "X" else "X"
        else:
            print("Position déjà occupée. Essayez à nouveau.")

if __name__ == "__main__":
    play_game()
```

Les deux code avec l´interface

```
import tkinter as tk
from tkinter import messagebox

class TicTacToe:
    def __init__(self):
        self.window = tk.Tk()
        self.window.title("Tic Tac Toe")
        self.current_player = "X"
        self.board = [" " for _ in range(9)]

        for i in range(9):
            button = tk.Button(self.window, text=" ", font=("Helvetica", 24), height=2,
width=5,
```

```
                              command=lambda i=i: self.make_move(i))
            button.grid(row=i // 3, column=i % 3)

    def make_move(self, position):
        if self.board[position] == " ":
            self.board[position] = self.current_player
            self.update_board()
            if self.check_winner():
                messagebox.showinfo("Winner", f"Player {self.current_player} wins!")
                self.reset_board()
            elif " " not in self.board:
                messagebox.showinfo("Tie", "It's a tie!")
                self.reset_board()
            else:
                self.current_player = "O" if self.current_player == "X" else "X"

    def update_board(self):
        for i, cell in enumerate(self.board):
            self.window.grid_slaves(row=i // 3, column=i % 3)[0].config(text=cell)

    def check_winner(self):
        winning_combinations = [(0, 1, 2), (3, 4, 5), (6, 7, 8),
                                (0, 3, 6), (1, 4, 7), (2, 5, 8),
                                (0, 4, 8), (2, 4, 6)]
        for combo in winning_combinations:
            if all(self.board[i] == self.current_player for i in combo):
                return True
        return False

    def reset_board(self):
        self.current_player = "X"
        self.board = [" " for _ in range(9)]
        self.update_board()

    def run(self):
        self.window.mainloop()

if __name__ == "__main__":
    game = TicTacToe()
    game.run()
```

Les troisièmes code

```
# Créez un plateau vide de Tic Tac Toe
def initialiser_plateau():
    return [" "] * 9

# Affiche le plateau de jeu
def afficher_plateau(plateau):
    print(f"{plateau[0]} | {plateau[1]} | {plateau[2]}")
    print("---------")
    print(f"{plateau[3]} | {plateau[4]} | {plateau[5]}")
    print("---------")
    print(f"{plateau[6]} | {plateau[7]} | {plateau[8]}")

# Vérifie si un joueur a gagné
def verifier_victoire(plateau, joueur):
    combinaisons_gagnantes = [
        [0, 1, 2], [3, 4, 5], [6, 7, 8],  # Lignes
        [0, 3, 6], [1, 4, 7], [2, 5, 8],  # Colonnes
        [0, 4, 8], [2, 4, 6]              # Diagonales
    ]
    for combinaison in combinaisons_gagnantes:
        if all(plateau[i] == joueur for i in combinaison):
            return True
    return False

# Joue une partie de Tic Tac Toe
def jouer_tic_tac_toe():
    plateau = initialiser_plateau()
    joueur_actuel = "X"
    while True:
        afficher_plateau(plateau)
        position = int(input(f"Joueur {joueur_actuel}, choisissez une position (1-9) :
")) - 1
        if plateau[position] == " ":
            plateau[position] = joueur_actuel
            if verifier_victoire(plateau, joueur_actuel):
                afficher_plateau(plateau)
```

```
            print(f"Joueur {joueur_actuel} a gagné !")
            break
        joueur_actuel = "O" if joueur_actuel == "X" else "X"
    else:
        print("Position déjà occupée. Choisissez une autre position.")
        continue

if __name__ == "__main__":
    jouer_tic_tac_toe()
```

Avec de l´interface

Sans interface

```
# Créez un plateau vide de Tic Tac Toe
def initialiser_plateau():
# Créez un plateau vide de Tic Tac Toe
def initialiser_plateau():
    return [" "] * 9

# Affiche le plateau de jeu
def afficher_plateau(plateau):
    print(f"{plateau[0]} | {plateau[1]} | {plateau[2]}")
    print("---------")
    print(f"{plateau[3]} | {plateau[4]} | {plateau[5]}")
    print("---------")
    print(f"{plateau[6]} | {plateau[7]} | {plateau[8]}")

# Vérifie si un joueur a gagné
def verifier_victoire(plateau, joueur):
    combinaisons_gagnantes = [
        [0, 1, 2], [3, 4, 5], [6, 7, 8],  # Lignes
        [0, 3, 6], [1, 4, 7], [2, 5, 8],  # Colonnes
        [0, 4, 8], [2, 4, 6]              # Diagonales
    ]
    for combinaison in combinaisons_gagnantes:
```

```
        if all(plateau[i] == joueur for i in combinaison):
            return True
    return False

# Joue une partie de Tic Tac Toe
def jouer_tic_tac_toe():
    plateau = initialiser_plateau()
    joueur_actuel = "X"
    while True:
        afficher_plateau(plateau)
        position = int(input(f"Joueur {joueur_actuel}, choisissez une position (1-9) :
")) - 1
        if plateau[position] == " ":

          plateau[position] = joueur_actuel
            if verifier_victoire(plateau, joueur_actuel):
                afficher_plateau(plateau)
                print(f"Joueur {joueur_actuel} a gagné !")
                break
            joueur_actuel = "O" if joueur_actuel == "X" else "X"
        else:
            print("Position déjà occupée. Choisissez une autre position.")
            continue

if __name__ == "__main__":
    jouer_tic_tac_toe()

    return [" "] * 9

# Affiche le plateau de jeu
def afficher_plateau(plateau):
    print(f"{plateau[0]} | {plateau[1]} | {plateau[2]}")
    print("---------")
    print(f"{plateau[3]} | {plateau[4]} | {plateau[5]}")
    print("---------")
    print(f"{plateau[6]} | {plateau[7]} | {plateau[8]}")

# Vérifie si un joueur a gagné
def verifier_victoire(plateau, joueur):
    combinaisons_gagnantes = [
        [0, 1, 2], [3, 4, 5], [6, 7, 8],  # Lignes
        [0, 3, 6], [1, 4, 7], [2, 5, 8],  # Colonnes
```

```
        [0, 4, 8], [2, 4, 6]           # Diagonales
    ]
    for combinaison in combinaisons_gagnantes:
        if all(plateau[i] == joueur for i in combinaison):
            return True
    return False

# Joue une partie de Tic Tac Toe
def jouer_tic_tac_toe():
    plateau = initialiser_plateau()
    joueur_actuel = "X"
    while True:
        afficher_plateau(plateau)
        position = int(input(f"Joueur {joueur_actuel}, choisissez une position (1-9) :
")) - 1
        if plateau[position] == " ":
            plateau[position] = joueur_actuel
            if verifier_victoire(plateau, joueur_actuel):
                afficher_plateau(plateau)
                print(f"Joueur {joueur_actuel} a gagné !")
                break
            joueur_actuel = "O" if joueur_actuel == "X" else "X"
        else:
            print("Position déjà occupée. Choisissez une autre position.")
            continue

if __name__ == "__main__":
    jouer_tic_tac_toe()
```