CHAT GPT

 import **random**:

- The random module allows us to generate random values in Python. In this game, we use it to select a random move for the computer.

 import os:

- The os module provides a way to interact with the operating system. Here, we use it to clear the terminal screen after each round to make the game look cleaner.

 import re:

- The re module handles regular expressions, which are patterns used to match strings. In this game, we use it to validate user input (ensure that the user only types valid choices like fire, water, or grass).

valid_moves:

- This is a **list** that stores the three valid moves in the game: "fire", "water", and "grass". We use this list later to randomly select a move for the computer, and to validate the user's input.

 def clear_screen()::

- This is a function definition. clear_screen is a function that clears the terminal screen. The purpose of this is to refresh the screen between rounds for a cleaner user experience.

 os.system('cls' if os.name == 'nt' else 'clear'):

- os.system() allows us to run a command on the operating system from within Python.
- os.name checks the operating system (OS) you're running the script on.
    - If it's a Windows system (os.name == 'nt'), the command cls is used to clear the screen.
    - Otherwise, for Linux and macOS, the command clear is used.

 def get_user_choice()::

- This defines a function get_user_choice() that asks the user for input (to choose one of the moves: fire, water, or grass) and validates it.

 while True::

- This creates an infinite loop. We use this loop to repeatedly ask for user input until we get a valid response. This ensures the program won't proceed unless the user enters a valid move.

 user_input = input("Enter your move (fire, water, or grass): ").lower():

- input() displays a prompt in the console asking for the user's input.
- .lower() converts the user's input to lowercase, making it case-insensitive. So, no matter if the user types Fire, fIrE, or fire, it will always be converted to fire.

 if re.match(r'^(fire|water|grass)$', user_input)::

- This checks if the user_input matches one of the valid choices (i.e., "fire", "water", or "grass").
- re.match() is a function from the re module that checks if a string matches a given regular expression (regex).
    - r'^(fire|water|grass)$' is a regular expression:
        - ^ means "start of the string".
        - (fire|water|grass) is a group that matches one of the three options (fire, water, or grass).
        - $ means "end of the string".
- This ensures that the input is exactly fire, water, or grass and nothing else.

 return user_input:

- If the input matches one of the valid choices, the function returns the user_input, so it can be used later in the game.

 else::

- If the input does not match the valid choices, it prints an error message:
    - "Invalid choice. Please choose 'fire', 'water', or 'grass'."
- The loop then repeats, asking for valid input again.

 def get_computer_choice()::

- This defines a function to get the computer's move.

 return random.choice(valid_moves):

- random.choice(valid_moves) randomly selects an element from the valid_moves list. This simulates the computer's random choice between fire, water, or grass.

- **`def determine_winner(user_choice, computer_choice):`**
    - This function takes in two arguments: the user's choice and the computer's choice. It compares these choices to determine the winner.
- **`if user_choice == computer_choice:`**
    - If both the user and the computer choose the same thing, it's a tie.
    - It returns `"It's a tie!"`.
- **`if (user_choice == 'fire' and computer_choice == 'grass') or ...:`**
    - This block of conditions checks the possible outcomes of the game:
        - Fire beats Grass.
        - Grass beats Water.
        - Water beats Fire.
    - If any of these conditions are true, the function returns `"You win!"`.
- **`return "Computer wins!"`**:
    - If none of the above conditions are true, then the computer wins, and it returns `"Computer wins!"`.

- **`def play_game():`**
  - This is the main function that runs the game. It’s the entry point of the game, where everything happens.
- **`clear_screen()`:**
  - Clears the screen before each round for a clean slate.
- **`print("Welcome to Fire, Water, Grass!")`:**
  - This prints a welcome message to the player.
- **`user_choice = get_user_choice()`:**
  - Calls the function to get the user's choice and stores it in `user_choice`.
- **`computer_choice = get_computer_choice()`:**
  - Calls the function to get the computer's choice and stores it in `computer_choice`.
- **`print(f"\nYou chose: {user_choice}")`:**
  - Prints the user's move to the console.
- **`print(f"The computer chose: {computer_choice}")`:**
  - Prints the computer's move to the console.
- **`result = determine_winner(user_choice, computer_choice)`:**
  - Calls the `determine_winner()` function to decide the winner based on both the user’s and computer’s choices.
- **`print(result)`:**
  - Prints the result (whether the user won, the computer won, or it was a tie).
- **`play_again = input("\nDo you want to play again? (yes/no): ").lower()`:**
  - Asks the user if they want to play again. The `.lower()` method ensures the input is case-insensitive.
- **`if play_again == 'yes':`**
  - If the user enters `"yes"`, it recursively calls `play_game()` to start a new
  -
