Raffle App

This Raffle App is designed to manage and conduct a raffle, where users can purchase tickets to participate in draws for various prizes. The application supports different prize groups and displays winners for each prize group based on the number of matching numbers in the purchased tickets.

Installation and Running the Application

Prerequisites

- Python 3.10 and Windows is required.
- Ensure all necessary libraries (standard libraries) are installed.

Steps to Run

1. Clone or download the application files to your local machine.
2. Navigate to the directory containing the files.

   ```
   cd path-to-folder/raffle-app
   ```

3. Run the application with the following command:

   ```
   python src/main.py
   ```

File Structure

```
raffle-app
├── .coverage
└── htmlcov
└── src
│   ├── main.py
│   ├── prize_group.py
│   ├── raffle.py
│   ├── ticket.py
│   ├── user.py
└── tests
    ├── __pycache__
    ├── test_main.py
    ├── test_prize_group.py
    ├── test_raffle.py
    ├── test_ticket.py
    └── test_user.py
```

1. `main.py`
    - Serves as the main entry point for the raffle application.
    - Presents a user-friendly menu allowing participants to:
        - Start a new raffle draw.
        - Purchase tickets by entering their name and the number of tickets they wish to buy.
        - Run the raffle to randomly generate winning numbers, calculate results, and display winners.
2. `raffle.py`
    - Contains the `Raffle` class, which manages the entire raffle process, including ticket purchases, winning number generation, prize distribution, and result display.
3. `ticket.py`
    - Contains the `Ticket` class, which represents a raffle ticket.
    - Each ticket has a unique set of 5 numbers between 1 to 15 and is associated with a user.
4. `user.py`
    - Contains the `User` class, which represents a participant in the raffle.
    - Manages user information, including purchased tickets.
5. `prize_group.py`
    - Contains the `PrizeGroup` class, which categorises prizes based on criteria.
    - Calculates the rewards for each group.

Running Tests

Run All Tests

- To run all tests with detailed test output, navigate to the root directory of the application and execute the following command (including the `-v` verbose flag):

```
pytest -v
```

Run a Specific Test File

- To run a specific test file with detailed test output, use:

```
pytest -v tests/test_xxx.py # where test_xxx is the test file name
```

Note: All pytest files should start with `test_` or end with `_test`.

Run Tests with Code Coverage

- To measure code coverage, we use the pytest-cov plugin. If you haven't installed it yet, run:

  ```
  pip install pytest-cov
  ```

- To run tests with code coverage and display a coverage report in the terminal:

  ```
  pytest --cov=src --cov-report=term
  ```

- To create a more detailed HTML report, use:

  ```
  pytest --cov=src --cov-report=html
  ```

  Note: After running this command, open htmlcov/index.html in a browser to view the coverage report. As of the last report, the existing code coverage is approximately **97%**. Testing `main()` is omitted to avoid redundancy, as its functionality is already covered through unit tests on `display_menu()` and `handle_menu_choice()`.

Assumptions

- **Ticket Purchase Limitations**: Each user can buy multiple tickets in a single purchase but is limited to a total maximum of 5 tickets.
  - If the user has already reached the maximum ticket limit, they cannot buy more tickets.
  - If the requested ticket count exceeds the remaining allowance, only the allowed amount is purchased.
- **Rewards Distribution**: Rewards are structured into different prize groups, each with a percentage of the total pot and a specific number of winning tickets required:

Prize Group	Winning Numbers Required	Percentage of Total Pot
Group 2	2 winning numbers	10%
Group 3	3 winning numbers	15%
Group 4	4 winning numbers	25%
Group 5 (Jackpot)	5 winning numbers	50%

  - If there is more than one winning ticket in any prize group, the reward for that group will be **shared evenly** among the ticket holders. Rewards are rounded off to 2 decimal places.
  - Any **remaining** amount in the pot after rewards distribution will **roll over** to the next raffle draw.

- **Initial Pot**: The raffle starts with a seeded amount of **100** in the pot.
- **Ticket Sales**: All proceeds from ticket sales are added to the pot, increasing the total amount available for rewards.
- **Ticket Structure**: Each ticket consists of **5 randomly generated and unique numbers** between **1 and 15**.
- **Randomisation**: The selection of winning tickets is **random**, ensuring fair distribution among participants.
- **Data Management**: The application assumes in-memory data management without persistence (i.e., no database). All data is **reset** upon each run.