Movie Rating System

`call()`, `apply()`, and `bind()`

1. Add a New Rating

- **Task:** Given this function that adds a new rating:

```
function addRating(rating) {
 this.ratings.push(rating);
}
```

Use the relevant method (`call()` , `apply()` , or `bind()`) to execute this function with a different context.

- **Inputs:** A Movie object and a rating object.
- **Example:**
  - Input: `Movie object {title: 'Inception', ratings: [8, 9, 10]}` , `Rating object {user: 'John Doe', rating: 8}`
  - Output: `Movie object {title: 'Inception', ratings: [8, 9, 10, 8]}`

2. Calculate Average Rating

- **Task:** Given this function that calculates the average rating:

```
function calculateAverageRating() {
 let sum = this.ratings.reduce((a, b) => a + b, 0);
 return sum / this.ratings.length;
}
```

Use the relevant method to execute this function with a different context.

- **Inputs:** A movie object.
- **Example:**
  - Input: `Movie object {title: 'Inception', ratings: [8, 9, 10, 8]}`
  - Output: `8.75`

3. Bind a User to a Rating Method

- **Task:** Given this function that adds a rating from a user:

```
function addRatingUser(rating) {
 this.ratings.push(rating);
}
```

Use the relevant method to permanently bind this function to a user context.

- **Inputs:** A user object.
- **Example:**
  - Input: `User object {name: 'John Doe'}`
  - Output: `Bound function: bound addRatingUser`

4. Display User Ratings

- **Task:** Given this function that displays all the ratings made by a user:

```
function displayRatings() {
 return this.ratings.map(rating => rating.rating);
}
```

Use the relevant method to invoke this function with different contexts.

- **Inputs:** A user object.
- **Example:**
  - Input:
    `User object {name: 'John Doe', ratings: [{movie: 'Inception', rating: 8}, {movie: 'Titanic', rating: 9}]}`
  - Output: `[8, 9]`

5. Calculate Average User Rating

- **Task:** Given this function that calculates the average rating given by a user:

```
function calculateAverageUserRating() {
 let sum = this.ratings.reduce((a, b) => a + b.rating, 0);
 return sum / this.ratings.length;
}
```

Use the relevant method to execute this function with a different context.

- **Inputs:** A user object.
- **Example:**
  - Input: `User object {name: 'John Doe', ratings: [{movie: 'Inception', rating: 8}, {movie: 'Titanic', rating: 9}]}`
  - Output: `8.5`

6. Bind a Movie to a Rating Display Method

- **Task:** Given this function that displays a movie's ratings:

```
function displayRatings() {
 return this.ratings;
}
```

Use the relevant method to permanently bind this function to a movie context.

- **Inputs:** A movie object.
- **Example:**
  - Input: `Movie object {title: 'Inception', ratings: [8, 9, 10]}`
  - Output: `Bound function: bound displayRatings`

7. Update Movie Rating

- **Task:** Given this function that updates a rating from a specific user:

```
function updateRating(user, newRating) {
 this.ratings = this.ratings.map(rating =>
   rating.user === user.name ? { user: user.name, rating: newRating } :
rating
 );
}
```

Use the relevant method to execute this function with a different context.

- **Inputs:** A movie object, a user object, and a new rating.
- **Example:**
  - Input: `Movie object {title: 'Inception', ratings: [{user: 'John Doe', rating: 8}]}`, `User object {name: 'John Doe'}`, `New rating: 10`

- Output:

```
Movie object {title: 'Inception', ratings: [{user: 'John Doe',
rating: 10}]}
```

8. Filter Movie Ratings

- **Task:** Given this function that filters the ratings of a movie based on a rating value:

```
function filterRatings(minRating) {
 return this.ratings.filter(rating => rating >= minRating);
}
```

Use the relevant method to execute this function with a different context.

- **Inputs:** A movie object and a minimum rating value.
- **Example:**
  - Input: `Movie object {title: 'Inception', ratings: [8, 9, 10, 8, 10]`
  - Output: `[9, 10, 10]`