UNIT TEST

**Function 1: multiplication**

**Test Specifications**:

1. Expect multiplication (2, 3) to be a number.
2. Expect multiplication (2, 3) to be equal to 6.
3. Expect multiplication (0, 5) to be equal to 0.
4. Expect multiplication (-2, 3) to be equal to -6.
5. Expect multiplication ("a", 3) to be an error.
6. Expect multiplication (2) to be an error.

**Function 2: concatOdds**

**Test Specifications**:

1. Expect concatOdds ([3, 2, 1], [9, 1, 1, 1, 4, 15, -1]) to be an array.
2. Expect concatOdds ([3, 2, 1], [9, 1, 1, 1, 4, 15, -1]) to be equal to [-1, 1, 3, 9, 15].
3. Expect concatOdds ([], []) to be an empty array [].
4. Expect concatOdds ("not an array", [1, 2, 3]) to be an error or an empty array [].
5. Expect concatOdds ([3.5, "a", 1], [9, 1, 4.2]) to be equal to [1, 9].
6. Expect concatOdds ([1, 1, 3], [3, 5, 5]) to be equal to [1, 3, 5].
7. Expect concatOdds ([-3, -2, 1], [5, -1, 7]) to be equal to [-3, -1, 1, 5, 7].

Functional Test

**Function 1: multiplication**

**Test Specifications**:

1. When a user calls multiplication (2, 3), it should return the number 6.
2. When a user calls multiplication (0, 5), it should return the number 0.
3. When a user calls multiplication (-2, 3), it should return the number -6.
4. When a user calls multiplication ("a", 3), it should throw an error.
5. When a user calls multiplication (2) with only one argument, it should throw an error.

Functional Test

**Test Specifications**

**Format 1: "Expect [action] to be [some result]"**

1. **Empty Cart**:
   * Expect attempts to check out with an empty cart to show an error message: "Your cart is empty. Please add items to proceed."
2. **Guest Checkout**:
   * Expect a guest user to be able to check out without creating an account.
   * Expect the guest user to be prompted with a message: "Would you like to create an account or log in to save your details for next time?"
3. **Logged-In User Checkout**:
   * Expect a logged-in user to be able to check out without additional prompts.
   * Expect the logged-in user’s saved payment and shipping information to be pre-filled (if available).
4. **Create Account After Guest Checkout**:
   * Expect the guest user to be given an option to create an account after completing the checkout.
   * Expect the guest user’s order details to be transferred to their new account if they choose to create one.
5. **Payment and Shipping Validation**:
   * Expect the user to be shown an error if payment information is invalid or missing.
   * Expect the user to be shown an error if shipping information is invalid or missing.
6. **Order Confirmation**:
   * Expect the user to be shown an order confirmation page with a summary of their purchase after successful checkout.

**Assumptions and Behaviors to Consider**

1. **Empty Cart**:
   * If the cart is empty, the user should not be allowed to proceed to checkout. Instead, they should be prompted to add items.
2. **Steps During Checkout**:
   * The user should be shown a clear sequence of steps: Cart Review → Shipping Information → Payment Information → Order Confirmation.
   * At each step, the user should be able to go back and make changes.
3. **Guest vs. Logged-In User**:
   * Guest users should not be forced to create an account but should be encouraged to do so for future convenience.
   * Logged-in users should have a streamlined experience with pre-filled details.
4. **Prompts for Account Creation**:
   * After guest checkout, the user should be given a clear and non-intrusive option to create an account or log in.
5. **Error Handling**:
   * The system should validate all inputs (e.g., payment details, shipping address) and provide clear error messages if something is missing or invalid.
6. **Order Confirmation**:
   * The confirmation page should include all relevant details: items purchased, total cost, shipping address, and estimated delivery date.