

Cypress for QA Problem Set 1 Solutions

1 1. Basic UI Checks

Problem 1.1: Check page title

```
it('should have the correct title', () => {  
  cy.visit('https://example.com');  
  cy.title().should('include', 'Example Domain');  
});
```

Problem 1.2: Check element is visible

```
it('should show the main header', () => {  
  cy.get('h1').should('be.visible');  
});
```

Problem 1.3: Check if a button exists and is clickable

```
it('should find and click the button', () => {  
  cy.get('#submit-btn').should('exist').click();  
});
```

2 2. Forms and Inputs

Problem 2.1: Fill in a form and submit

```
it('should fill and submit login form', () => {  
  cy.get('#email').type('test@example.com');  
  cy.get('#password').type('password123');  
  cy.get('#login-btn').click();  
});
```

Problem 2.2: Check input validation error

```
it('should show validation for empty form', () => {  
  cy.get('#login-btn').click();  
  cy.get('.error').should('contain', 'Email is  
    required');  
});
```

Problem 2.3: Verify redirection after form submit

```
it('should redirect to dashboard', () => {  
  cy.get('#email').type('user@example.com');  
  cy.get('#password').type('pass');  
  cy.get('#login-btn').click();  
  cy.url().should('include', '/dashboard');  
});
```

3 3. Assertions and Conditions

Problem 3.1: Check if list contains 5 items

```
it('should render 5 list items', () => {  
  cy.get('.task-item').should('have.length', 5);  
});
```

Problem 3.2: Check for specific text in list item

```
it('should contain a task called "Buy milk"', () => {  
  cy.get('.task-item').contains('Buy milk');  
});
```

Problem 3.3: Conditional rendering based on checkbox

```
it('should show extra options when checkbox is checked',  
  () => {  
    cy.get('#extra-options').should('not.be.visible');  
    cy.get('#toggle-options').check();  
    cy.get('#extra-options').should('be.visible');  
  });
```

4 4. Network Interception and Waits

Problem 4.1: Wait for API response and verify data

```
it('should wait for the users API and check count', ()
=> {
  cy.intercept('GET', '/api/users').as('getUsers');
  cy.visit('/users');
  cy.wait('@getUsers').its('response.statusCode').
    should('eq', 200);
  cy.get('.user-card').should('have.length', 10);
});
```

Problem 4.2: Mock login API response

```
it('should mock login response', () => {
  cy.intercept('POST', '/api/login', {
    statusCode: 200,
    body: { token: 'fake-token' }
  }).as('login');

  cy.get('#email').type('test@example.com');
  cy.get('#password').type('1234');
  cy.get('#login-btn').click();

  cy.wait('@login');
  cy.get('.welcome-msg').should('contain', 'Welcome');
});
```

5 5. Intermediate Logic and Custom Commands

Problem 5.1: Custom command for login

```
// In commands.js
Cypress.Commands.add('login', (email, password) => {
  cy.get('#email').type(email);
  cy.get('#password').type(password);
  cy.get('#login-btn').click();
});

// Usage
it('should login using custom command', () => {
  cy.login('user@example.com', 'securepass');
  cy.url().should('include', '/dashboard');
});
```

Problem 5.2: Reusable command for adding a task

```
// In commands.js
Cypress.Commands.add('addTask', (task) => {
  cy.get('#new-task-input').type(task);
  cy.get('#add-task-btn').click();
});

// Usage
it('should add 3 tasks', () => {
  ['Task 1', 'Task 2', 'Task 3'].forEach(task => {
    cy.addTask(task);
  });
  cy.get('.task-item').should('have.length', 3);
});
```