Lista de Cenários de Teste

Cenários	**Descrição**
Exibição inicial	**Descrição**: Verificar se o display mostra `0` ao iniciar a calculadora **Passos**: Abrir a calculadora **Resultado Esperado**: Display deve mostrar `0`
Digitação de múltiplos dígitos	**Descrição**: Verificar se a calculadora concatena dígitos (ex: `1` → `2` → `3` deve resultar em `123`) **Passos**: Clicar em `1`, `2`, `3` **Resultado Esperado**: Display mostra `123`
Ignorar zeros à esquerda	**Descrição**: Verificar se `005` é exibido como `5` **Passos**: Clicar em `0`, `0`, `5` **Resultado Esperado**: Display mostra `5`
Inserção de ponto decimal	**Descrição**: Verificar se `0.56` é exibido corretamente **Passos**: Clicar em `.`, `5`, `6` **Resultado Esperado**: Display mostra `0.56`
Bloqueio de múltiplos pontos decimais	**Descrição**: Verificar se a calculadora impede `1.2.3` **Passos**: Clicar em `1`, `.`, `2`, `.`, `3` **Resultado Esperado**: Display mostra `1.23` (ignora o segundo `.`)

Adição simples (5 + 3 = 8)	**Passos:** 5, +, 3, = **Resultado Esperado:** 8.
Subtração simples (9 – 4 = 5)	**Passos:** 9, –, 4, = **Resultado Esperado:** 5
Multiplicação simples (6 × 7 = 42)	**Passos:** 6, ×, 7, = **Resultado Esperado:** 42
Divisão simples (8 ÷ 2 = 4)	**Passos:** 8, ÷, 2, = **Resultado Esperado:** 4
Divisão por zero (5 ÷ 0 = ?)	**Descrição:** Verificar como a calculadora lida com divisão por zero. **Passos:** 5, ÷, 0, = **Resultado Esperado:** Exibir "Error" ou "Infinity"
Operação sequencial (2 + 3 × 4 = 14)	**Passos:** 2, +, 3, ×, 4, = **Resultado Esperado:** 14 (prioridade de multiplicação)
Limpar display (AC)	**Passos:** 1, 2, 3, AC **Resultado Esperado:** Display volta para 0
Operação após resultado (5 + 3 = 8 → + 2 = 10)	**Passos:** 5, +, 3, =, +, 2, = **Resultado Esperado:** 10
Troca de operador (5 + × 3 = 15)	**Passos:** 5, +, ×, 3, = **Resultado Esperado:** 15 (substitui + por ×)
Números negativos (5 – 7 = -2)	**Passos:** 5, –, 7, = **Resultado Esperado:** -2
Máximo de dígitos no display	**Descrição:** Verificar se a calculadora limita dígitos (ex: 15 caracteres) **Passos:** Clicar em 1 16 vezes **Resultado Esperado:** Apenas 15 dígitos são exibidos

Operação sem segundo número (`5 + =`)	**Passos:** `5`, `+`, `=` **Resultado Esperado:** `5` (ignora `=` sem segundo número)
Pressionar `=` sem operação (`5 =`)	**Passos:** `5`, `=` **Resultado Esperado:** `5` (não altera o valor)
Operação com decimal (`0.5 × 2 = 1`)	**Passos:** `.`, `5`, `×`, `2`, `=` **Resultado Esperado:** `1`
Reset após erro (`5 ÷ 0 = Error → AC`)	**Passos:** `5`, `÷`, `0`, `=`, `AC` **Resultado Esperado:** Display volta para `0`

Dicas:

```
it('5 + 3 = 8', () => {
  cy.contains('button', '5').click();
  cy.contains('button', '+').click();
  cy.contains('button', '3').click();
  cy.contains('button', '=').click();
  cy.get('.display').should('have.text', '8');
});
```

1. Usar `cy.get('.display')` para verificar o resultado.
2. Usar `cy.contains('button', '5').click()` para simular cliques.
3. Implementar comandos personalizados (ex: `cy.clickButtons(['1', '+', '2'])`)

DOCUMENTAÇÃO DAS FALHAS

1. Operação sequencial (2 + 3 × 4 = 14)

```
it('Operação sequencial (2 + 3 × 4 = 14)', () => {
  clickButtons(['2', '+', '3', '×', '4', '=']);
  cy.get('.display').should('have.text', '14');
});
```

- **Falha potencial:**
  A expressão `2 + 3 × 4` está sendo interpretada com **precedência matemática correta**, resultando em `2 + (3 × 4) = 14`.
  No entanto, muitas calculadoras simples executam operações **da esquerda para a direita**, sem considerar precedência, o que daria `((2 + 3) × 4) = 20`.

- **Impacto:** Comportamento pode confundir o usuário se não for o padrão esperado para o tipo da calculadora (simples vs científica).

- **Solução possível:** Definir e documentar claramente a lógica da calculadora (com ou sem precedência).

---

2. Máximo de dígitos no display (15 dígitos)

```
it('Máximo de dígitos no display (15 dígitos)', () => {
  clickButtons(Array(16).fill('1'));
  cy.get('.display').invoke('text').should('have.length.lte', 15);
});
```

- **Falha potencial:**
  O teste verifica o `length` do texto no display, mas isso não garante que:
  - A entrada seja realmente **limitada**.
  - O sistema **bloqueie** o usuário após o 15º dígito.

- **Problema comum:** A digitação pode continuar, mas o texto é apenas cortado visualmente ou ocorre estouro de valor.

- **Solução possível:** Validar se o botão de número **deixa de funcionar** após o limite.

---

3. Pressionar "=" sem operação (5 =)

```
it('Pressionar = sem operação (5 =)', () => {
  clickButtons(['5', '=']);
  cy.get('.display').should('have.text', '5');
});
```

- **Falha potencial:**
  O teste espera que o display mantenha o `5`, o que está correto.
  Porém, **não valida** se pressionar `=` várias vezes:

  - Executa uma operação anterior armazenada (como `5 + 0`, por exemplo).

  - Gera erro ou comportamento inesperado.

- **Solução possível:** Testar múltiplos `=` após um número sem operador.

---

4. Inserção de ponto decimal

```
it('Inserção de ponto decimal', () => {
  clickButtons(['.', '5', '6']);
  cy.get('.display').should('have.text', '0.56');
});
```

- **Falha potencial:**
  O teste cobre `['.', '5', '6']` esperando `0.56`, mas **não cobre casos como:**

    - Pressionar `.` após uma operação (`5 + .`)
    - Pressionar `.` após um resultado (`5 + 3 = .`)
    - Pressionar `.` sem número anterior (`.` → deve mostrar `0.`)
- **Solução possível:** Adicionar testes para **ponto isolado** e **ponto após resultado/operação**.

SUGESTÕES DE NOVOS TESTES

Bloqueio após 15 dígitos

Justificativa:
Garante que a entrada de números **não ultrapasse o limite máximo** visual e de cálculo, prevenindo erros de layout, overflow ou problemas de precisão.

Inserção de ponto decimal isolado

Justificativa:
Verifica se o ponto decimal **sozinho** é tratado corretamente e não gera erro visual ou de lógica. Usuários costumam começar com `.` ao digitar valores menores que 1.

Continuação da operação após resultado com operador

Justificativa:

Já existe esse teste, mas é importante destacar que ele cobre uma lógica crucial: **usar o resultado como primeiro número da próxima operação**, comportamento típico de calculadoras comuns.