

## Busca de elementos

Aline está criando testes para a sua plataforma de e-commerce, e precisa testar a funcionalidade de busca. Para isso, ela criou o seguinte código:

**buscaProduto.po.ts**

```
import {browser, element, by} from 'protractor';
export class BuscaProdutoPage {

  navegarParaBuscaProdutos(){
    return browser.get('/buscaProduto');
  }

  retornarInputBuscaProduto(){
    return element(by.css('.inputBusca')).sendKeys('Computador')
  }

  retornarBotaoDeBusca(){
    return element(by.id('botaoBusca'))
  }
}
```

E o arquivo **produto.e2e-spec.ts**:

```
import { BuscaProdutoPage } from 'buscaProduto.po';

describe('Testando tela de Produto', ()=>{
  let buscaProdutoPage: BuscaProdutoPage;
  beforeEach(()=>{
    buscaProdutoPage = new BuscaProdutoPage();
  })

  it('Deve testar funcionalidade de busca', ()=>{
    buscaProdutoPage.navegarParaBuscaProdutos();
    buscaProdutoPage.retonarInputBuscaProduto();
    expect(buscaProdutoPage.retonarBotaoDeBusca().getText()).toEqual('Buscar');
    buscaProdutoPage.retonarBotaoDeBusca().click();

  })
})
```

Com base nos códigos apresentados acima, é correto afirmar que:

*Selecione uma alternativa*

**A**

Não funcionará, pois o `sendKeys` não é um método que pertence ao `element()`, e sim, ao `by()`.

**B**

Funcionará até o momento em que fizer a comparação do `toEqual()` no arquivo `produto.e2e-spec.ts`, pois a busca de um elemento não retorna ele inteiro (com todos os atributos `html`) mas só o que foi selecionado no `pageObject`, no caso apenas o `id`.

**C**

Funcionará, pois foi criado em uma estrutura de `PageObject` onde ficam encapsulados os elementos de uma página, e chamados para serem comparados e testados no arquivo `e2e` de acordo com um fluxo de busca.

**D**

Não funcionará, pois o correto deveria ser utilizar todas as buscas de maneira apartada. Ou seja, cada `it` deveria conter apenas um método chamado no `page Objects`, pois do contrário, atrapalhará a sincronicidade da execução do código.