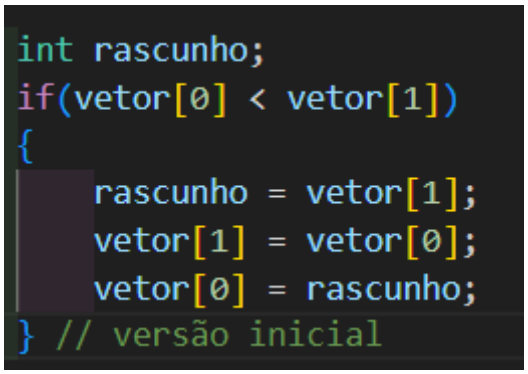
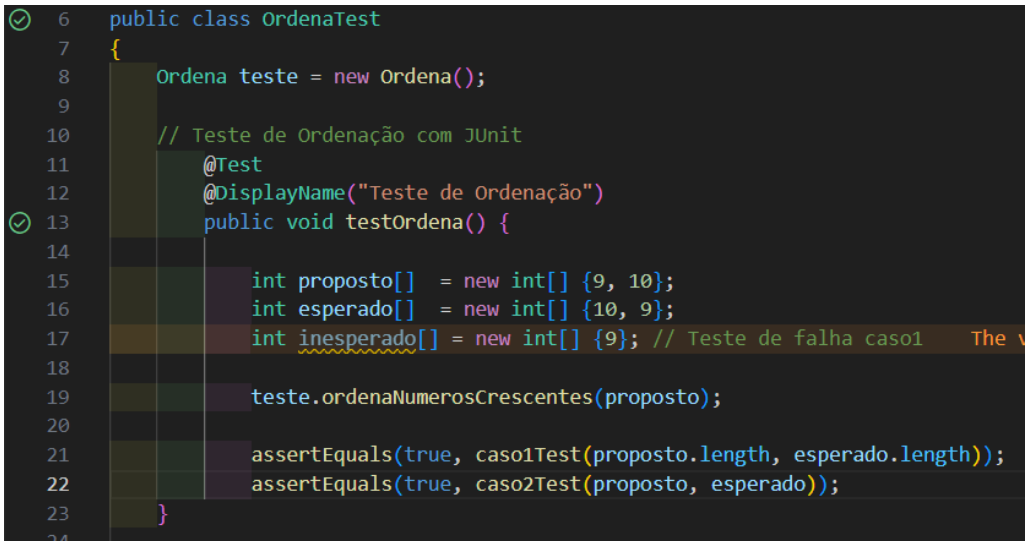
Ex1

Ordenação vazia gerou dois falsos

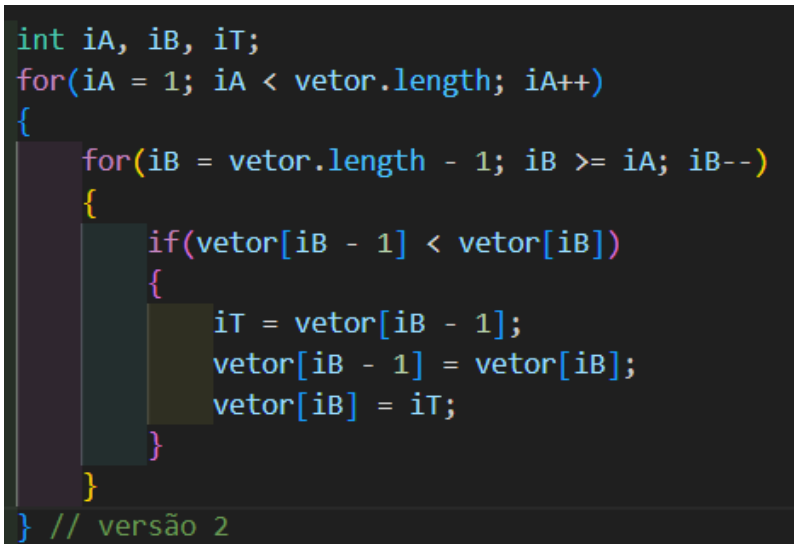
Ordenação específica para o exemplo gerou dois verdadeiros



Ordenação geral decrescente gerou dois verdadeiros



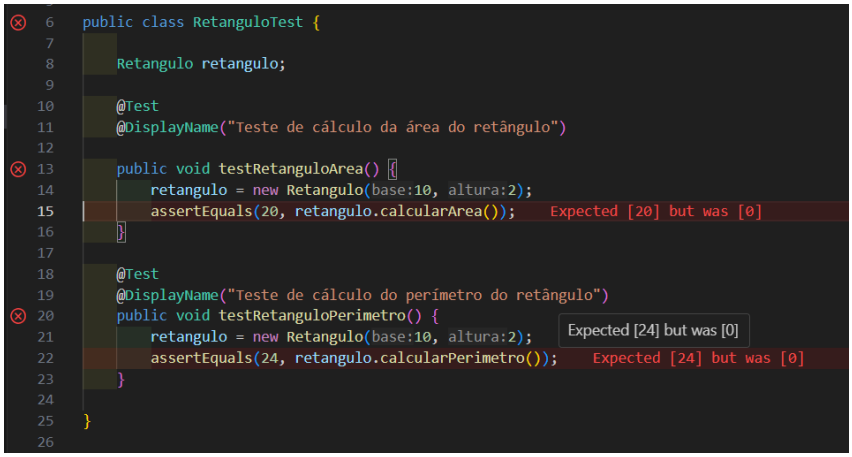
```java
public class OrdenaTest
{
    Ordena teste = new Ordena();

    // Teste de Ordenação com JUnit
        @Test
        @DisplayName("Teste de Ordenação")
        public void testOrdena() {

            int proposto[]  = new int[] {9, 10};
            int esperado[]  = new int[] {10, 9};
            int inesperado[] = new int[] {9}; // Teste de falha caso1    The v

            teste.ordenaNumerosCrescentes(proposto);

            assertEquals(true, caso1Test(proposto.length, esperado.length));
            assertEquals(true, caso2Test(proposto, esperado));
    }
```



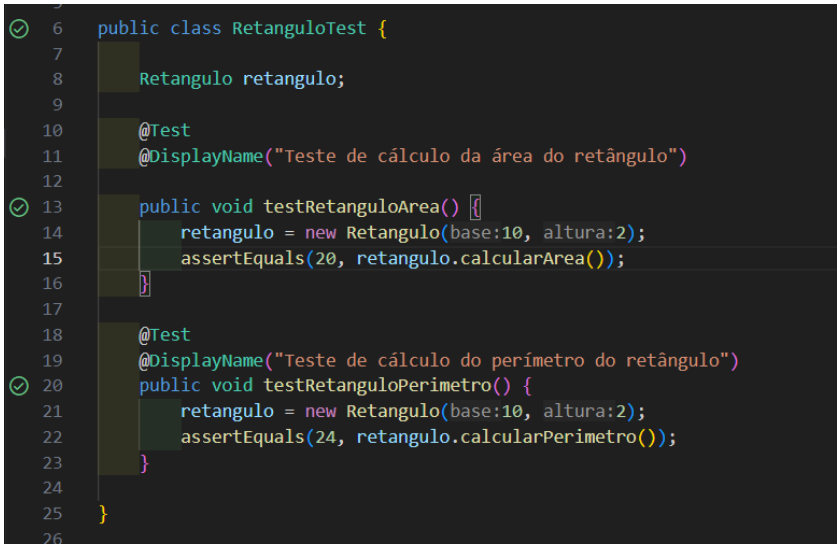
```java
int iA, iB, iT;
for(iA = 1; iA < vetor.length; iA++)
{
    for(iB = vetor.length - 1; iB >= iA; iB--)
    {
        if(vetor[iB - 1] < vetor[iB])
        {
            iT = vetor[iB - 1];
            vetor[iB - 1] = vetor[iB];
            vetor[iB] = iT;
        }
    }
} // versão 2
```

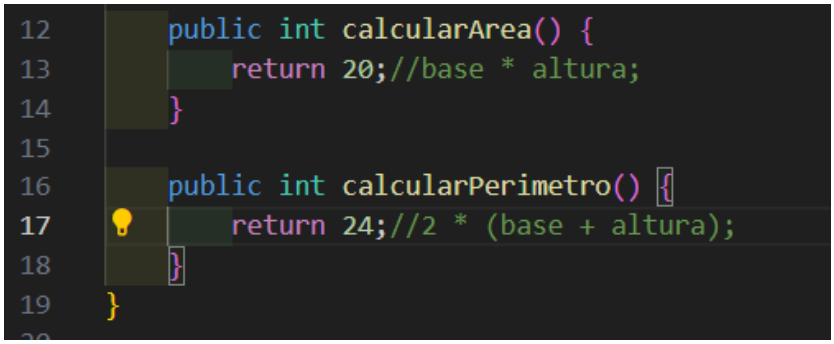
Ex 2

Cálculos vazios geraram dois falses



Respostas diretas gerara dois verdadeiros





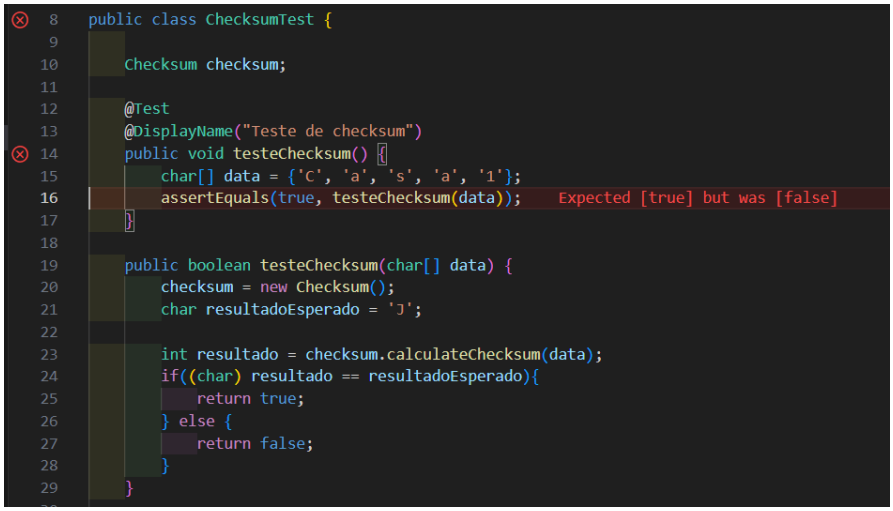
Cálculos certos geraram dois verdadeiros

```java
public class RetanguloTest {

    Retangulo retangulo;

    @Test
    @DisplayName("Teste de cálculo da área do retângulo")

    public void testRetanguloArea() {
        retangulo = new Retangulo(base:10, altura:2);
        assertEquals(20, retangulo.calcularArea());
    }

    @Test
    @DisplayName("Teste de cálculo do perímetro do retângulo")
    public void testRetanguloPerimetro() {
        retangulo = new Retangulo(base:10, altura:2);
        assertEquals(24, retangulo.calcularPerimetro());
    }

}
```

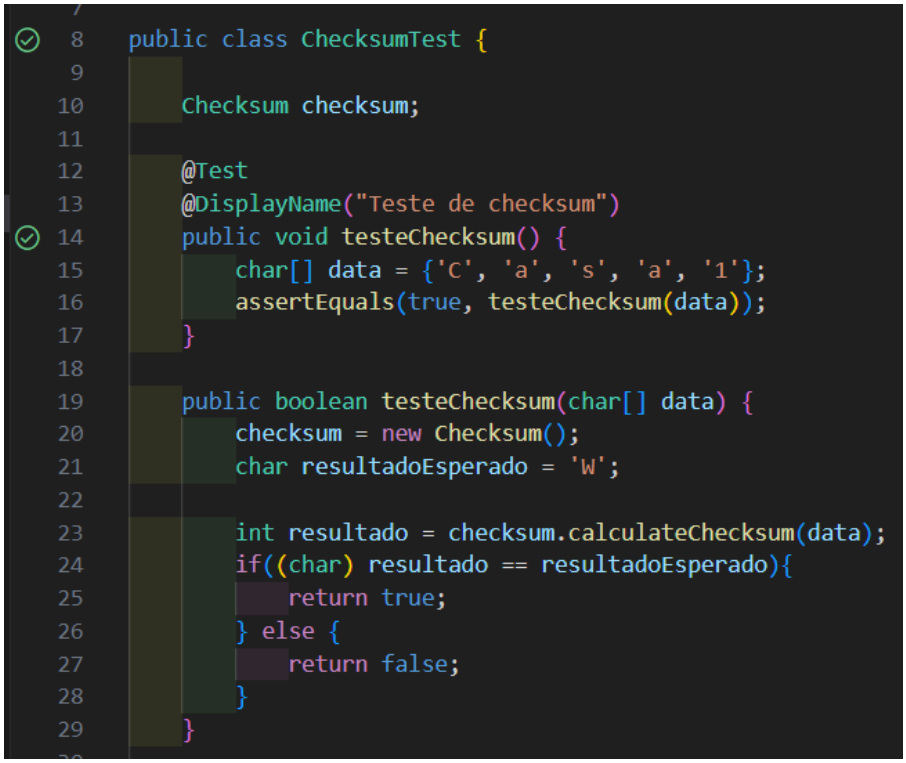
```java
    public int calcularArea() {
        return base * altura;
    }

    public int calcularPerimetro() {
        return 2 * (base + altura);
    }
}
```

Ex 3

Colocando o resultado do checksum esperado errado gera falso



Colocando resultado do checksum direto gera verdadeiro



```java
public int calculateChecksum(char[] data) {
    /*int checksum = 0;
    int sum = 0;
    for (int i = 0; i < data.length; i++) {
        sum += data[i];
    }
    checksum = (~sum + 1) & 0xFF;*/
    int checksum = 87;
    return checksum;
}
```

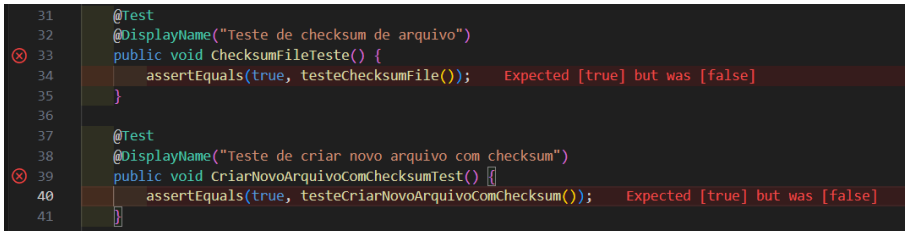
Colocando checksum para calcular e comparar com checksum esperado gera verdadeiro

```java
public class ChecksumTest {

    Checksum checksum;

    @Test
    @DisplayName("Teste de checksum")
    public void testeChecksum() {
        char[] data = {'C', 'a', 's', 'a', '1'};
        assertEquals(true, testeChecksum(data));
    }

    public boolean testeChecksum(char[] data) {
        checksum = new Checksum();
        char resultadoEsperado = 'W';

        int resultado = checksum.calculateChecksum(data);
        if((char) resultado == resultadoEsperado){
            return true;
        } else {
            return false;
        }
    }
```

```java
public int calculateChecksum(char[] data) {
    int checksum = 0;
    int sum = 0;
    for (int i = 0; i < data.length; i++) {
        sum += data[i];
    }
    checksum = (~sum + 1) & 0xFF;
    return checksum;
}
```

Ex 4

sem encontrar o arquivo para leitura gera dois falsos



Encontra o arquivo e gera novo, checksum resposta direta gera dois verdadeiros

```
31      @Test
32      @DisplayName("Teste de checksum de arquivo")
33      public void ChecksumFileTeste() {
34          assertEquals(true, testeChecksumFile());
35      }
36
37      @Test
38      @DisplayName("Teste de criar novo arquivo com checksum")
39      public void CriarNovoArquivoComChecksumTest() {
40          assertEquals(true, testeCriarNovoArquivoComChecksum());
41      }
42
```

```
public int calculateChecksumFile(String filename) throws Exception {
    int checksum = 87;
    return checksum;  // hardcode
}
```

Encontra o arquivo e gera novo, checksum calculada gera dois verdadeiros

```
31      @Test
32      @DisplayName("Teste de checksum de arquivo")
33      public void ChecksumFileTeste() {
34          assertEquals(true, testeChecksumFile());
35      }
36
37      @Test
38      @DisplayName("Teste de criar novo arquivo com checksum")
39      public void CriarNovoArquivoComChecksumTest() {
40          assertEquals(true, testeCriarNovoArquivoComChecksum());
41      }
```

```java
public int calculateChecksumFile(String filename) throws Exception {
    FileInputStream fis = null;
    int soma = 0;

    try {
        fis = new FileInputStream(filename);
        int byteLido;

        while ((byteLido = fis.read()) != -1) {
            soma += byteLido;
        }
    } finally {
        if (fis != null) {
            fis.close();
        }
    }

    int checksum = (~soma + 1) & 0xFF;
    return checksum;
}
```

Ex5

Valor esperado calculado com CRC-16, porém CRC usado para teste foi CRC-8 gerando falso

```java
80  @Test
81  @DisplayName("Teste de CRC")
82  public void CRCTeste() {
83      char[] data = {'C', 'a', 's', 'a', '1'};
84      assertEquals(true, testeCRC(data));    Expected [true] but was [false]
85  }
```

Forçando resultado gera verdadeiro

```java
80      @Test
81      @DisplayName("Teste de CRC")
82      public void CRCTeste() {
83          char[] data = {'C', 'a', 's', 'a', '1'};
84          assertEquals(true, testeCRC(data));
85      }
86
```

```java
public int calulateCRC(char[] data) {
    int resto = 2145812606;
    return resto;
}
```

CRC-16 funcionando gera verdadeiro

```java
    @Test
    @DisplayName("Teste de CRC")
    public void CRCTeste() {
        char[] data = {'C', 'a', 's', 'a', '1'};
        assertEquals(true, testeCRC(data));
    }
```

```java
public int calulateCRC(char[] data) {
    int resto = 0;
    int pol = 0x8005;
    int ordem = 0x8000;

    for (int i = 0; i < data.length; i++) {
        resto ^= (data[i] << 15);
        for (int j = 0; j < 15; j++) {
            if ((resto & ordem) != 0) {
                resto = (resto << 1) ^ pol;
            } else {
                resto = (resto << 1);
            }
        }
    }
    return resto;
}
```