Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

1. **Ter um método chamado converterCamelCase**
   **1.1. Teste**

Foi criado um teste para usar o método converterCamelCase conforme requisitado no enunciado.

Código do teste:

```
@Test
public void testeAssinatura() {
	List<String> listRetorno = new ArrayList<String>();
	listRetorno.add("nome");

	assertEquals(listRetorno, Texto.converterCamelCase("nome"));
}
```

**1.2. Desenvolvimento**

- Foi criada a **classe Texto** vazia.
- Foi criado o **método converterCamelCase**, retornando uma lista com um único item de valor fixo (pois ainda não tem outro teste para processar).
- O teste foi finalizado com sucesso.

Código desenvolvido

```
import java.util.ArrayList;
import java.util.List;

public class Texto {

	public static List<String> converterCamelCase(String
original) {

		List<String> listRetorno = new ArrayList<String>();
		listRetorno.add("nome");

		return listRetorno;

	}
}
```

**1.3. Refatoração**

Foi comentado o que cada função irá realizar.

Código refatorado

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
// Metodo base para interacao, unico metodo obrigatorio segundo o
exercicio
public static List<String> converterCamelCase(String original) {

	List<String> listRetorno = new ArrayList<String>();
	listRetorno.add("nome");

	return listRetorno;

}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

2. **Converter uma palavra com a primeira letra maiuscula**
   **2.1. Teste**

Foi criado um teste para usar o método converterCamelCase com o valor “Nome” e que deve retornar a lista [“nome”]

Código do teste:

```
@Test
public void testeUmaPalavraEmCamelCase() {
	List<String> listRetorno = new ArrayList<String>();
	listRetorno.add("nome");

	assertEquals(listRetorno, Texto.converterCamelCase("Nome"));
}
```

**2.2. Desenvolvimento**

O código do método converterCamelCase foi editado para retornar o item que for passado no parâmetro, porém todos os caracteres em minusculo.

Código desenvolvido

```
// Metodo base para interacao, unico metodo obrigatorio segundo
o exercicio
public static List<String> converterCamelCase(String original) {

	List<String> listRetorno = new ArrayList<String>();
	listRetorno.add(original.toLowerCase());

	return listRetorno;

}
```

**2.3. Refatoração**

Foi trocado o nome do parâmetro do método converterCamelCase e colocados comentários no método de forma a facilitar a leitura do código

Código refatorado

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
// Metodo base para interacao, unico metodo obrigatorio segundo o
exercicio
public static List<String> converterCamelCase(String valor) {

	List<String> listRetorno = new ArrayList<String>(); //
Instancia uma lista do tipo Array para o objeto String.
	listRetorno.add(valor.toLowerCase()); // Adiciona o valor
passado no parametro do metodo em caixa baixa.

	return listRetorno; // Retorna a lista editada acima

}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

3. **Converter duas palavras no Camel Case**
   **3.1. Teste**

Foi criado um teste para usar o método converterCamelCase com o valor “nomeComposto” e que deve retornar a lista [“nome”, “composto”]

Código do teste:

```
@Test
public void testeDuasPalavrasEmCamelCase() {
      List<String> listRetorno = new ArrayList<String>();
      listRetorno.add("nome");
      listRetorno.add("composto");

      assertEquals(listRetorno,
Texto.converterCamelCase("nomeComposto"));
}
```

**3.2. Desenvolvimento**

O código do método converterCamelCase foi editado para retornar o item que for passado no parâmetro, porém todos os caracteres em minusculo.

Código desenvolvido

```
// Metodo base para interacao, unico metodo obrigatorio segundo o exercicio
public static List<String> converterCamelCase(String valor) {

      List<String> listRetorno = new ArrayList<String>(); // Instancia uma lista do tipo
Array para o objeto String.
      int ultimoCharMaiusculo = 0;

      for(int i = 0; i < valor.length(); i++) {
            char caractere = valor.charAt(i);
            if (Character.isUpperCase(caractere)) {
                  listRetorno.add(valor.substring(ultimoCharMaiusculo, i).toLowerCase());
// Adiciona o valor passado no parametro do metodo em caixa baixa.
                  ultimoCharMaiusculo = i;
            }
      }
    listRetorno.add(valor.substring(ultimoCharMaiusculo, valor.length()).toLowerCase()); //
Adiciona o valor passado no parametro do metodo em caixa baixa.

      return listRetorno; // Retorna a lista editada acima

}
```

Porém apesar de passar no teste de duas palavras, retornou o erro no teste de uma palavra com a primeira letra maiúscula, por conta de interpretar a primeira letra em maiuscula como uma segunda palavra em vez de uma primeira palavra.

```
// Metodo base para interacao, unico metodo obrigatorio segundo o exercicio
public static List<String> converterCamelCase(String valor) {

	List<String> listRetorno = new ArrayList<String>(); // Instancia uma lista do tipo
Array para o objeto String.
	int ultimoCharMaiusculo = 0;

	for(int i = 0; i < valor.length(); i++) {
		char caractere = valor.charAt(i);
		if (Character.isUpperCase(caractere)) {
			String palavra = valor.substring(ultimoCharMaiusculo, i).toLowerCase();
			if (!palavra.equals("")) {
				listRetorno.add(palavra); // Adiciona o valor passado no parametro
do metodo em caixa baixa.
			}
			ultimoCharMaiusculo = i;
		}
	}
	listRetorno.add(valor.substring(ultimoCharMaiusculo, valor.length()).toLowerCase());
// Adiciona o valor passado no parametro do metodo em caixa baixa.

	return listRetorno; // Retorna a lista editada acima

}
```

### 3.3. Refatoração

Foi trocado o nome do parâmetro do método converterCamelCase e colocados comentários no método de forma a facilitar a leitura do código, também foi realizada a edição do código para manter as 10 linhas de código por função.

Código refatorado

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
public class Texto {

	// Metodo base para interacao, unico metodo obrigatorio segundo o exercicio
	public static List<String> converterCamelCase(String valor) {
		List<String> listRetorno = new ArrayList<String>(); // Instancia uma lista do
tipo Array para o objeto String.
		for (int i = proximoCharMaiusculo(valor); i < valor.length() && i > -1; i =
proximoCharMaiusculo(valor.substring(i))) { // Metodo FOR para rodar entre as palavras
			if (i > 0) { // Caso não seja o primeiro caractere, insere o dado na
lista
				listRetorno.add(valor.substring(0, i).toLowerCase()); // Adiciona o
valor passado no parametro do metodo em caixa baixa.
				valor = valor.substring(i);
			} else { // caso seja o primeiro caractere (primeira letra do primeiro
nome em maiuscula), prossegue para o proximo
				i = 1;
			}
		}
		listRetorno.add(valor.toLowerCase()); // Adiciona o valor passado no parametro do
metodo em caixa baixa.
		return listRetorno; // Retorna a lista editada acima
	}

	// Funcao para retornar o index do proximo caractere maiusculo do texto
	static int proximoCharMaiusculo (String valor) {
		for(int i = 0; i < valor.length(); i++) { // FOR para rodar cada caractere do
texto
		  if (Character.isUpperCase(valor.charAt(i))) { // Caso o caractere for
maiusculo, retorna o index
			return i;
		  }
		}
		return -1; // Caso nao encontre algum caractere maiusculo, retorna erro (-1)
	}
}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

4. **Converter duas palavras com a primeira letra maiuscula**
   **4.1. Teste**

Foi criado um teste para usar o método converterCamelCase com o valor “NomeComposto” e que deve retornar a lista [“nome”, “composto”]

Código do teste:

```
@Test
public void testeDuasPalavrasEmCamelCase2() {
      List<String> listRetorno = new ArrayList<String>();
      listRetorno.add("nome");
      listRetorno.add("composto");

      assertEquals(listRetorno,
Texto.converterCamelCase("NomeComposto"));
}
```

**4.2. Desenvolvimento**

O código do método converterCamelCase foi editado para editar o texto recebido (no caso “NomeComposto”), para esse item, tivemos problemas quanto ao uso do laço for na função principal, para corrigir tivemos que colocar alguns prints para debug e adaptar o código.

Código desenvolvido

```
// Metodo base para interacao, unico metodo obrigatorio segundo o exercicio
public static List<String> converterCamelCase(String valor) {
      List<String> listRetorno = new ArrayList<String>(); // Instancia uma lista do tipo
Array para o objeto String.
    for (int i = proximoCharMaiusculo(valor); i < valor.length() && i > -1; ) { // Metodo FOR
para rodar entre as palavras

            System.out.printf("i: %d\n", i);
            System.out.printf("valor: %s\n", valor);

      if (i > 0) { // Caso não seja o primeiro caractere, insere o dado na lista
                  listRetorno.add(valor.substring(0, i).toLowerCase()); // Adiciona o valor
passado no parametro do metodo em caixa baixa.
                  System.out.printf("ADD: %s\n", valor.substring(0, i).toLowerCase());
                  valor = valor.substring(i);
                  i = proximoCharMaiusculo(valor.substring(i));
            } else { // caso seja o primeiro caractere (primeira letra do primeiro nome em
maiuscula), prossegue para o proximo
                  i  = proximoCharMaiusculo(valor.substring(1)) + 1;
            }
    }
    listRetorno.add(valor.toLowerCase()); // Adiciona o valor passado no parametro do metodo
em caixa baixa.
    System.out.printf("ADD: %s\n", valor.toLowerCase());
      return listRetorno; // Retorna a lista editada acima
}

// Funcao para retornar o index do proximo caractere maiusculo do texto
static int proximoCharMaiusculo (String valor) {
    System.out.println("\tPROX CHAR MAIUSCULO");
    for(int i = 0; i < valor.length(); i++) { // FOR para rodar cada caractere do texto
          System.out.printf("Char: %c\n", valor.charAt(i));
      if (Character.isUpperCase(valor.charAt(i))) { // Caso o caractere for maiusculo,
retorna o index
          System.out.println("DEU RUIM: -1");
          return i;
      }
    }
    System.out.println("DEU RUIM: -1");
    return -10; // Caso nao encontre algum caractere maiusculo, retorna erro (-1)
}
```

### 4.3. Refatoração

Neste caso, a primeira melhoria é a remoção dos comandos de print que foram utilizados para debug, após essa tratativa, corrigimos os comandos para reduzir as linhas de cada função para menor ou igual a 10.

Código refatorado

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
// Metodo base para interacao, unico metodo obrigatorio segundo o exercicio
public static List<String> converterCamelCase(String valor) {
      List<String> listRetorno = new ArrayList<String>(); // Instancia uma lista do tipo
Array para o objeto String.
    for (int i = proximoCharMaiusculo(valor, 1); i < valor.length() && i > -1; i =
proximoCharMaiusculo(valor.substring(i), i)) { // Metodo FOR para rodar entre as palavras
      if (i > 0) { // Caso não seja o primeiro caractere, insere o dado na lista
                  listRetorno.add(valor.substring(0, i).toLowerCase()); // Adiciona o valor
passado no parametro do metodo em caixa baixa.
                  valor = valor.substring(i);
            }
    }
    listRetorno.add(valor.toLowerCase()); // Adiciona o valor passado no parametro do
metodo em caixa baixa.
      return listRetorno; // Retorna a lista editada acima
}

// Funcao para retornar o index do proximo caractere maiusculo do texto
static int proximoCharMaiusculo (String valor, int anteriorMaiuscula) {
    for(int i = anteriorMaiuscula == 0 ? 1 : 0; i < valor.length(); i++) { // FOR para
rodar cada caractere do texto
      if (Character.isUpperCase(valor.charAt(i))) { // Caso o caractere for maiusculo,
retorna o index
            return i;
      }
    }
    return -10; // Caso nao encontre algum caractere maiusculo, retorna erro (-1)
}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

5. **Converter uma palavra com todas as letras maiusculas**
   **5.1. Teste**

Foi criado um teste para usar o método converterCamelCase com o valor “CPF” e que deve retornar a lista [“CPF”].

Código do teste:

```
      @Test
      public void testeUmaPalavraTodaEmMaiuscula() {
            List<String> listRetorno = new ArrayList<String>();
            listRetorno.add("CPF");

            assertEquals(listRetorno,
Texto.converterCamelCase("CPF"));
      }
```

**5.2. Desenvolvimento**

O código do método converterCamelCase foi refeito para poder atender a necessidade atual (novo teste).

Código desenvolvido

```
public static List<String> converterCamelCase (String valorInicial) {
      System.out.printf("CONVERTER CAMEL CASE: %s\n", valorInicial);
      List<String> valorConvertido = new ArrayList<String>(); // Instancia uma lista do tipo
Array para o objeto String.
      String palavra = "";
      int maiuscula = 0;

    for(int i = 0; i < valorInicial.length(); i++) { // FOR para rodar cada caractere do
texto
            System.out.printf("\tchar: %c\n", valorInicial.charAt(i));
      if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for maiusculo,
retorna o index
            maiuscula++;
                  System.out.printf("\t\tcounter maiuscula: %d\n", maiuscula);
      } else {
                  System.out.printf("\t\tMinuscula\n");
            if (maiuscula == 1) {
                  if (i != 1){
                              System.out.printf("\t\t\tPalavra formada (=1): %s\n",
palavra.substring(0, i-1));
                        valorConvertido.add(palavra.substring(0, i-1).toLowerCase());
                              System.out.printf("\t\t\tADD: %s\n", palavra.substring(0, i-
1).toLowerCase());
                        palavra = palavra.substring(i-1);
                  }
                        maiuscula = 0;
            } else if (maiuscula > 1) {
                        System.out.printf("\t\t\tPalavra formada (>1): %s\n", palavra);
                  valorConvertido.add(palavra);
                  System.out.printf("\t\t\tADD: %s\n", palavra);
                  palavra = "";
            }
      }
      palavra += valorInicial.charAt(i);
    }

    if (maiuscula > 1) {
            System.out.printf("\t\t\tPalavra formada (final): %s\n", palavra);
            valorConvertido.add(palavra);
            System.out.printf("\t\t\tADD: %s\n", palavra);
    } else {
            System.out.printf("\t\t\tPalavra formada (final): %s\n", palavra);
            valorConvertido.add(palavra.toLowerCase());
            System.out.printf("\t\t\tADD: %s\n", palavra.toLowerCase());
    }

      return valorConvertido;
}
```

**5.3. Refatoração**

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

Neste caso, a primeira melhoria é a remoção dos comandos de print que foram utilizados para debug, após essa tratativa, corrigimos os comandos para reduzir as linhas de cada função para menor ou igual a 10.

Código refatorado

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
public class Texto {
	static List<String> valorConvertido;
	static String palavra;
	static int maiuscula;

	// Funcao principal para converter uma String em CamelCase em uma lista com as
palavras separadas em itens
	public static List<String> converterCamelCase (String valorInicial) {
		valorConvertido = new ArrayList<String>(); // Instancia uma lista do tipo Array
para o objeto String.
		palavra = "";
		maiuscula = 0;

		buscaPalavras(valorInicial);

	    separaUltimaPalavra();

	    return valorConvertido;
	}

	// Faz o laco FOR para a busca de palavras
	static void buscaPalavras (String valorInicial) {
	    for(int i = 0; i < valorInicial.length(); i++) { // FOR para rodar cada caractere
do texto
		if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for
maiusculo, retorna o index
			maiuscula++;
		} else {
			separaPalavra(i);
		}
		palavra += valorInicial.charAt(i);
	    }
	}

	// Separa as palavras dentro do loop FOR
	static void separaPalavra(int i) {
		if (maiuscula == 1) {
			separaPalavraEmMaiusculo(i);
		} else if (maiuscula > 1) {
			valorConvertido.add(palavra);
			palavra = "";
		}
	}

	// Separa as palavras com apenas uma letra em maiuscula
	static void separaPalavraEmMaiusculo(int i) {
		if (i != 1){
			valorConvertido.add(palavra.substring(0, i-1).toLowerCase());
			palavra = palavra.substring(i-1);
		}
		maiuscula = 0;
	}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
	// Separa a ultima rodada de palavras (apos o laco FOR)
	static void separaUltimaPalavra () {
	    if (maiuscula > 1) {
	            valorConvertido.add(palavra);
	    } else {
	            valorConvertido.add(palavra.toLowerCase());
	    }
	}
}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

6. **Converter uma palavra minuscula e uma palavra com todas as letras maiusculas**
   **6.1. Teste**

Foi criado um teste para usar o método converterCamelCase com o valor “numeroCPF” e que deve retornar a lista [“numero”, “CPF”].

Código do teste:

```
	@Test
	public void testeUmaPalavraMinusculaUmaPalavraTodaEmMaiuscula() {
		List<String> listRetorno = new ArrayList<String>();
		listRetorno.add("numero");
		listRetorno.add("CPF");

		assertEquals(listRetorno, Texto.converterCamelCase("numeroCPF"));
	}
```

**6.2. Desenvolvimento**

O código do método converterCamelCase foi convertido para se adaptar ao teste, sendo necessário a alteração de algumas funções.

Código desenvolvido

```
public class Texto {
	static List<String> valorConvertido;
	static String palavra;
	static int maiuscula;

	public static void printLista (List<String> lista) {
		for(int i = 0; i < lista.size(); i++){
			System.out.printf("Lista[%d]: %s\n", i, lista.get(i));
		}
	}

	// Funcao principal para converter uma String em CamelCase em uma lista com as palavras
separadas em itens
	public static List<String> converterCamelCase (String valorInicial) {
		valorConvertido = new ArrayList<String>(); // Instancia uma lista do tipo Array
para o objeto String.
		palavra = valorInicial.charAt(0)+"";
		maiuscula =
Character.isUpperCase(valorInicial.charAt(0))&&Character.isUpperCase(valorInicial.charAt(1))?
1:0;
		buscaPalavras(valorInicial);
	    separaUltimaPalavra();
	    return valorConvertido;
	}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
      // Faz o laco FOR para a busca de palavras
      static void buscaPalavras (String valorInicial) {
          for(int i = 1; i < valorInicial.length(); i++) { // FOR para rodar cada caractere
do texto
            if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for
maiusculo, retorna o index
                  if (maiuscula == 0) {
                        separaPalavraAntesMaiuscula(i);
                  }
                  maiuscula++;
            }
            palavra += valorInicial.charAt(i);
          }
      }


      // Separa as palavras com apenas uma letra em maiuscula
      static void separaPalavraEmMaiusculo(int i) {
      System.out.printf("Separar palavra em maiusculo[%d]: %s\n", i, palavra);
            if (i != 1){
                  valorConvertido.add(palavra.substring(0, i-1).toLowerCase());
                  palavra = palavra.substring(i-1);
            }
            maiuscula = 0;
      }

      // Separa a ultima rodada de palavras (apos o laco FOR)
      static void separaUltimaPalavra () {
      System.out.printf("Separar ultima palavra: %s\n", palavra);
          if (maiuscula > 1) {
                  valorConvertido.add(palavra);
          } else {
                  valorConvertido.add(palavra.toLowerCase());
          }
      }

      // Separa palavra antes da maiuscula
      static void separaPalavraAntesMaiuscula (int i) {
      System.out.printf("Separar palavra antes maiuscula[%d]: %s\n", i, palavra);
            valorConvertido.add(palavra.substring(0, i).toLowerCase());
            palavra = palavra.substring(i);
      }
}
```

### 6.3. Refatoração

Neste caso, a primeira melhoria é a remoção dos comandos de print que foram utilizados para debug, após essa tratativa, corrigimos os comandos para reduzir as linhas de cada função para menor ou igual a 10.

Código refatorado

```
public class Texto {
	static List<String> valorConvertido;
	static String palavra;
	static int maiuscula;

	public static void printLista (List<String> lista) {
		for(int i = 0; i < lista.size(); i++){
			System.out.printf("Lista[%d]: %s\n", i, lista.get(i));
		}
	}

	// Funcao principal para converter uma String em CamelCase em uma lista com as palavras separadas em itens
	public static List<String> converterCamelCase (String valorInicial) {
		valorConvertido = new ArrayList<String>(); // Instancia uma lista do tipo Array para o objeto String.
		palavra = valorInicial.charAt(0)+"";
		maiuscula = Character.isUpperCase(valorInicial.charAt(0))&&Character.isUpperCase(valorInicial.charAt(1))?1:0;
		buscaPalavras(valorInicial);
	   separaUltimaPalavra();
	   return valorConvertido;
	}

	}
}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
	// Faz o laco FOR para a busca de palavras
	static void buscaPalavras (String valorInicial) {
	   for(int i = 1; i < valorInicial.length(); i++) { // FOR para rodar cada caractere do texto
		if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for maiusculo, retorna o
index
			if (maiuscula == 0) {
				separaPalavraAntesMaiuscula(i);
			}
			maiuscula++;
		}
		palavra += valorInicial.charAt(i);
	   }
	}

	// Separa as palavras com apenas uma letra em maiuscula
	static void separaPalavraEmMaiusculo(int i) {
	System.out.printf("Separar palavra em maiusculo[%d]: %s\n", i, palavra);
		if (i != 1){
			valorConvertido.add(palavra.substring(0, i-1).toLowerCase());
			palavra = palavra.substring(i-1);
		}
		maiuscula = 0;
	}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
// Faz o laco FOR para a busca de palavras
static void buscaPalavras (String valorInicial) {
    for(int i = 1; i < valorInicial.length(); i++) { // FOR para rodar cada caractere do texto
     if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for maiusculo, retorna
o index
       if (maiuscula == 0) {
        separaPalavraAntesMaiuscula(i);
       }
       maiuscula++;
      }
      palavra += valorInicial.charAt(i);
    }
}

// Separa as palavras com apenas uma letra em maiuscula
static void separaPalavraEmMaiusculo(int i) {
    System.out.printf("Separar palavra em maiusculo[%d]: %s\n", i, palavra);
 if (i != 1){
  valorConvertido.add(palavra.substring(0, i-1).toLowerCase());
  palavra = palavra.substring(i-1);
 }
 maiuscula = 0;
}

// Separa a ultima rodada de palavras (apos o laco FOR)
static void separaUltimaPalavra () {
    System.out.printf("Separar ultima palavra: %s\n", palavra);
    if (maiuscula > 1) {
  valorConvertido.add(palavra);
    } else {
  valorConvertido.add(palavra.toLowerCase());
    }
}

// Separa palavra antes da maiuscula
static void separaPalavraAntesMaiuscula (int i) {
    System.out.printf("Separar palavra antes maiuscula[%d]: %s\n", i, palavra);
 valorConvertido.add(palavra.substring(0, i).toLowerCase());
 palavra = palavra.substring(i);
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

7. **Converter uma palavra minuscula, uma palavra com todas as letras maiusculas e outra minuscula**

### 7.1. Teste

Foi criado um teste para usar o método converterCamelCase com o valor “numeroCPFContribuinte” e que deve retornar a lista [“numero”, “CPF”, “contribuinte”].

Código do teste:

```
	@Test
	public void testeUmaPalavraMinusculaUmaPalavraTodaEmMaiusculaOutraMinuscula() {
		List<String> listRetorno = new ArrayList<String>();
		listRetorno.add("numero");
		listRetorno.add("CPF");
		listRetorno.add("contribuinte");

		assertEquals(listRetorno,
Texto.converterCamelCase("numeroCPFContribuinte"));
	}
```

### 7.2. Desenvolvimento

O código do método converterCamelCase foi convertido para se adaptar ao teste, sendo necessário a alteração de algumas funções.

Código desenvolvido

```
// Faz o laco FOR para a busca de palavras
static void buscaPalavras (String valorInicial) {
   for(int i = 1; i < valorInicial.length(); i++) { // FOR para rodar cada caractere do texto
        if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for maiusculo, retorna o index
                if (maiuscula == 0) {
                        separaPalavraAntesMaiuscula(i);
                }
                maiuscula++;
        } else {
                if (maiuscula > 1) {
                        separaPalavraMaiusculaMinuscula(i);
                }
        }
        palavra += valorInicial.charAt(i);
   }
}

static void separaPalavraMaiusculaMinuscula (int i) {
System.out.printf("Separar palavra maiuscula e minuscula[%d]: %s\n", i, palavra);
        valorConvertido.add(palavra.substring(0, palavra.length()-1));
        palavra = palavra.substring(palavra.length()-1);
        maiuscula = 0;
}
```

### 7.3. Refatoração

Neste caso, a primeira melhoria é a remoção dos comandos de print que foram utilizados para debug, após essa tratativa, corrigimos os comandos para reduzir as linhas de cada função para menor ou igual a 10.

Código refatorado

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
public class Texto {
	static List<String> valorConvertido;
	static String palavra;
	static int maiuscula;

	public static void printLista (List<String> lista) {
		for(int i = 0; i < lista.size(); i++){
			System.out.printf("Lista[%d]: %s\n", i, lista.get(i));
		}
	}

	// Funcao principal para converter uma String em CamelCase em uma lista com as palavras separadas
em itens
	public static List<String> converterCamelCase (String valorInicial) {
		valorConvertido = new ArrayList<String>(); // Instancia uma lista do tipo Array para o objeto String.
		palavra = valorInicial.charAt(0)+"";
		maiuscula =
Character.isUpperCase(valorInicial.charAt(0))&&Character.isUpperCase(valorInicial.charAt(1))?1:0;
		buscaPalavras(valorInicial);
	   separaUltimaPalavra();
	   return valorConvertido;
	}

	}
}

	// Faz o laco FOR para a busca de palavras
	static void buscaPalavras (String valorInicial) {
	   for(int i = 1; i < valorInicial.length(); i++) { // FOR para rodar cada caractere do texto
		if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for maiusculo, retorna o
index
			if (maiuscula == 0) {
				separaPalavraAntesMaiuscula(i);
			}
			maiuscula++;
		}
		palavra += valorInicial.charAt(i);
	   }
	}

	// Separa as palavras com apenas uma letra em maiuscula
	static void separaPalavraEmMaiusculo(int i) {
	System.out.printf("Separar palavra em maiusculo[%d]: %s\n", i, palavra);
		if (i != 1){
			valorConvertido.add(palavra.substring(0, i-1).toLowerCase());
			palavra = palavra.substring(i-1);
		}
		maiuscula = 0;
	}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
// Faz o laco FOR para a busca de palavras
static void buscaPalavras (String valorInicial) {
    for(int i = 1; i < valorInicial.length(); i++) { // FOR para rodar cada caractere do texto
     if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for maiusculo, retorna o index
      if (maiuscula == 0) {
       separaPalavraAntesMaiuscula(i);
      }
      maiuscula++;
     }
     palavra += valorInicial.charAt(i);
    }
}

// Separa as palavras com apenas uma letra em maiuscula
static void separaPalavraEmMaiusculo(int i) {
    System.out.printf("Separar palavra em maiusculo[%d]: %s\n", i, palavra);
 if (i != 1){
  valorConvertido.add(palavra.substring(0, i-1).toLowerCase());
  palavra = palavra.substring(i-1);
 }
 maiuscula = 0;
}

// Separa a ultima rodada de palavras (apos o laco FOR)
static void separaUltimaPalavra () {
    System.out.printf("Separar ultima palavra: %s\n", palavra);
    if (maiuscula > 1) {
  valorConvertido.add(palavra);
    } else {
  valorConvertido.add(palavra.toLowerCase());
    }
}

// Separa palavra antes da maiuscula
static void separaPalavraAntesMaiuscula (int i) {
    System.out.printf("Separar palavra antes maiuscula[%d]: %s\n", i, palavra);
 valorConvertido.add(palavra.substring(0, i).toLowerCase());
 palavra = palavra.substring(i);
}
```

8. **Gerar um tratamento para separar palavras de números**

**8.1. Teste**

Foi criado um teste para usar o método converterCamelCase com o valor “recupera10Primeiros” e que deve retornar a lista [“recupera”, “10”, “primeiros”].

Código do teste:

```
@Test
public void testeUmaPalavraMinusculaUmNumeroOutraMinuscula() {
      List<String> listRetorno = new ArrayList<String>();
      listRetorno.add("recupera");
      listRetorno.add("10");
      listRetorno.add("primeiros");

      assertEquals(listRetorno,
Texto.converterCamelCase("recupera10Primeiros"));
}
```

**8.2. Desenvolvimento**

Foi alterada a função **buscaPalavras** para poder identificar números.

Código desenvolvido

```
// Faz o laco FOR para a busca de palavras
static void buscaPalavras (String valorInicial) {
    for(int i = 1; i < valorInicial.length(); i++) { // FOR para rodar cada caractere do
texto
      if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for
maiusculo, retorna o index
            if (maiuscula == 0) {
                  separaPalavraAntesMaiuscula(i);
            }
            maiuscula++;
      } else if (Character.isDigit(valorInicial.charAt(i))) { // Caso o caractere for
numero, retorna o index
            System.out.printf("Separar palavra de numero[%d]: %s\n", i, palavra);
                  if (numero == 0) {
                        valorConvertido.add(palavra.substring(0, palavra.length()));
                        palavra = palavra.substring(palavra.length());
            }
            numero++;

      } else {
            if (maiuscula > 1) {
                  separaPalavraMaiusculaMinuscula(i);
            }
            numero = 0;
      }
      palavra += valorInicial.charAt(i);
    }
}
```

### 8.3. Refatoração

Foi comentado o que cada função irá realizar.

Código refatorado

```
// Faz o laco FOR para a busca de palavras
static void buscaPalavras (String valorInicial) {
   for(int i = 1; i < valorInicial.length(); i++) { // FOR para rodar cada caractere do texto
        testeChar(valorInicial, i);
        testeNumero(valorInicial, i);
        palavra += valorInicial.charAt(i);
   }
  }
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

```
	// Faz os testes necessarios para tratar caracteres dentro do laco FOR
	static void testeChar (String valorInicial, int i) {
		if (Character.isUpperCase(valorInicial.charAt(i))) { // Caso o caractere for maiusculo,
retorna o index
		if (maiuscula == 0)
			separaPalavraAntesMaiuscula(i);
		maiuscula++;
	} else {
		if (maiuscula > 1)
			separaPalavraMaiusculaMinuscula(i);
	}

	// Faz os testes necessarios para tratar numeros dentro do laco FOR
	static void testeNumero (String valorInicial, int i) {
		if (Character.isDigit(valorInicial.charAt(i))) { // Caso o caractere for numero, retorna o index
		separaPalavraDeNumero(i);
		} else {
		numero = 0;
		}
	}

	// Separa a palavra que vem antes dos numeros
	static void separaPalavraDeNumero (int i) {
	System.out.printf("Separar palavra de numero[%d]: %s\n", i, palavra);
		if (numero == 0) {
			valorConvertido.add(palavra.substring(0, palavra.length()));
			palavra = palavra.substring(palavra.length());
		}
		numero++;
	}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

9. **Gerar um tratamento para gerar erro caso haja um número no primeiro caractere**
   **9.1. Teste**

Foi criado um teste para usar o método converterCamelCase com o valor “10Primeiros” e que deve retornar uma RuntimeException.

Código do teste:

```
@Test
public void testeErroNumeroPrimeiro() {
      try {
            Texto.converterCamelCase("10Primeiros");
            fail();
      } catch (NumeroPrimeiroException e) {}
}
```

**9.2. Desenvolvimento**

Foi alterada a função **converterCamelCase** para rodar inicialmente um teste que permita o retorno de uma RuntimeException caso haja um numero no primeiro caractere da String.

Código desenvolvido

```
// Funcao principal para converter uma String em CamelCase em uma lista com as palavras
separadas em itens
public static List<String> converterCamelCase (String valorInicial) {
      if ( Character.isDigit(valorInicial.charAt(0)) ) {
            throw new NumeroPrimeiroException("Não é permitido que o primeiro valor seja
numérico.");
      }

      valorConvertido = new ArrayList<String>(); // Instancia uma lista do tipo Array para o
      objeto String.
      palavra = valorInicial.charAt(0)+"";
      maiuscula =
Character.isUpperCase(valorInicial.charAt(0))&&Character.isUpperCase(valorInicial.charAt(1)
)?1:0;
      buscaPalavras(valorInicial);
    separaUltimaPalavra();
    return valorConvertido;
}
```

**9.3. Refatoração**

A função **`converterCamelCase`** foi alterada e criada mais uma função chamada **testaFalhas** para poder manter o máximo de 10 linhas em cada função.

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

Código refatorado

```
// Funcao principal para converter uma String em CamelCase em uma lista com as palavras separadas em itens
public static List<String> converterCamelCase (String valorInicial) {
	testaFalhas(valorInicial);
	valorConvertido = new ArrayList<String>(); // Instancia uma lista do tipo Array para o objeto String.
	palavra = valorInicial.charAt(0)+"";
	maiuscula = Character.isUpperCase(valorInicial.charAt(0))&&Character.isUpperCase(valorInicial.charAt(1))?1:0;
	buscaPalavras(valorInicial);
   separaUltimaPalavra();
   return valorConvertido;
}

// Testa todas as falhas
static void testaFalhas (String valorInicial) {
	if ( Character.isDigit(valorInicial.charAt(0)) ) {
		throw new NumeroPrimeiroException("Não é permitido que o primeiro valor seja numérico.");
}
```

Aluno: Felipe Augusto Pereira dos Santos
Relatório de conversor de String em Camel Case

10. **Gerar um tratamento para gerar erro caso haja um número no primeiro caractere**

### 10.1. Teste

Foi criado um teste para usar o método converterCamelCase com o valor “`nome#Composto`” e que deve retornar uma RuntimeException.

Código do teste:

```
@Test
public void testeErroCaractereEspecial() {
	try {
		Texto.converterCamelCase("nome#Composto");
		fail();
	} catch (CaracteresEspeciaisException e) {}
}
```

### 10.2. Desenvolvimento

Foi alterada a função **converterCamelCase** para rodar inicialmente um teste que permita o retorno de uma RuntimeException caso haja um numero no primeiro caractere da String.

Código desenvolvido

```
// Testa todas as falhas
static void testaFalhas (String valorInicial) {
	if ( Character.isDigit(valorInicial.charAt(0)) ) {
		throw new NumeroPrimeiroException("Não é permitido que o primeiro valor seja
numérico.");
	}

	if ( temCaracteresEspeciais(valorInicial) ) {

	}

	for(int i = 1; i < valorInicial.length(); i++) {
		if ( !(Character.isAlphabetic(valorInicial.charAt(i)) ||
Character.isDigit(valorInicial.charAt(i)) ) ) {
			throw new CaracteresEspeciaisException("Não é permitido que tenha
caracteres especiais.");
		}
	}
}
```

### 10.3. Refatoração

A função **converterCamelCase** foi alterada a função chamada **testaFalhas** e criada a função **temCaracteresEspeciais** para poder manter o máximo de 10 linhas em cada função.

Código refatorado

```
// Testa todas as falhas
static void testaFalhas (String valorInicial) {
      if ( Character.isDigit(valorInicial.charAt(0)) ) {
            throw new NumeroPrimeiroException("Não é permitido que o primeiro valor
seja numérico.");
      }

      if ( temCaracteresEspeciais(valorInicial) ) {
            throw new CaracteresEspeciaisException("Não é permitido que tenha
caracteres especiais.");
      }
}

// Testa se a String inicial contem algum caractere que nao seja permitido
static boolean temCaracteresEspeciais (String valorInicial) {
      for(int i = 1; i < valorInicial.length(); i++) {
            if ( !(Character.isAlphabetic(valorInicial.charAt(i)) ||
Character.isDigit(valorInicial.charAt(i)) ) ) {
                  return true;
            }
      }
      return false;
}
```