

```
package recursos.aprendizagem;
```

```
import java.util.Arrays;
```

```
import java.util.LinkedList;
```

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

```
import java.util.Scanner;
```

```
public class TesteView {
```

```
    public static void main(String[] args) {
```

```
        System.out.print("Cpf: ");
```

```
        List<Integer> cpf = configuraCpf();
```

```
        int valorMult = multiplicaValores(cpf, algoritmo(1));
```

```
        int primeiroDig = divideValores(valorMult);
```

```
        cpf.add(primeiroDig);
```

```
        valorMult = multiplicaValores(cpf, algoritmo(2));
```

```
        int segundoDig = divideValores(valorMult);
```

```
        cpf.add(segundoDig);
```

```
        imprimeResultado(cpf);
```

```
    }
```

```
    private static List<Integer> configuraCpf() {
```

```
        Scanner sc = new Scanner(System.in);
```

```
        String[] cpfBruto = sc.next().split("");
```

```
        List<Integer> cpf = new LinkedList<>();
```

```
        for(String digito:cpfBruto) {  
            cpf.add(Integer.parseInt(digito));  
        }  
  
        sc.close();  
        return cpf;  
    }  
}
```

```
private static int multiplicaValores(List<Integer> cpf, List<Integer> algoritmo) {  
    int total = 0;  
    for(int i = 0; i < algoritmo.size(); i++) {  
        total += (algoritmo.get(i) * cpf.get(i));  
    }  
    return total;  
}  
}
```

```
private static int divideValores(int valorMult) {  
    int primeiroDig = valorMult % 11;  
    if(primeiroDig < 2)  
        primeiroDig = 0;  
    else  
        primeiroDig = (11 - primeiroDig);  
  
    return primeiroDig;  
}  
}
```

```
private static List<Integer> algoritmo(int etapa) {  
    List<Integer> algoritmo = new LinkedList<>();  
    if(etapa == 1)  
        algoritmo = Arrays.asList(10, 9, 8, 7, 6, 5, 4, 3, 2);  
    else if (etapa == 2)
```

```

        algoritmo = Arrays.asList(11, 10, 9, 8, 7, 6, 5, 4, 3, 2);
    else
        throw new IllegalArgumentException("Número de etapa inválido!");
    return algoritmo;
}

private static void imprimeResultado(List<Integer> cpf) {
    int cont = 0;
    for(int numero:cpf) {
        if(cont == 3 | cont == 6)
            System.out.print(".");
        else if(cont == 9)
            System.out.print("-");
        System.out.print(numero);
        cont++;
    }
}
}

```

```

1 package recursos.aprendizagem;
2
3 import java.util.Arrays;
4 import java.util.LinkedList;
5 import java.util.List;
6 import java.util.Scanner;
7
8 public class TesteVow {
9
10     public static void main(String[] args) {
11         System.out.print("CPF: ");
12         List<Integer> cpf = configuraCPF();
13
14         int valorMult = multiplicaValores(cpf, algoritmo(1));
15         int primeiroDig = divideValores(valorMult);
16         cpf.add(primeiroDig);
17
18         valorMult = multiplicaValores(cpf, algoritmo(2));
19         int segundoDig = divideValores(valorMult);
20         cpf.add(segundoDig);
21
22         imprimeResultado(cpf);
23     }
24
25     private static List<Integer> configuraCPF() {
26         Scanner sc = new Scanner(System.in);
27         String[] cpfBruto = sc.next().split("");
28         List<Integer> cpf = new LinkedList<>();
29         for (String digito : cpfBruto) {
30             cpf.add(Integer.parseInt(digito));
31         }
32         sc.close();
33         return cpf;
34     }
35
36     private static int multiplicaValores(List<Integer> cpf, List<Integer> algoritmo) {
37         int total = 0;
38         for (int i = 0; i < algoritmo.size(); i++) {
39             total += (algoritmo.get(i) * cpf.get(i));
40         }
41         return total;
42     }
43
44     private static int divideValores(int valorMult) {
45         int primeiroDig = valorMult % 11;
46         if (primeiroDig < 2)
47             primeiroDig = 0;
48         else
49             primeiroDig = (11 - primeiroDig);
50     }
51 }

```

```

52     return primeiroDig;
53 }
54
55 private static List<Integer> algoritmo(int etapa) {
56     List<Integer> algoritmo = new LinkedList<>();
57     if (etapa == 1)
58         algoritmo = Arrays.asList(10, 9, 8, 7, 6, 5, 4, 3, 2);
59     else if (etapa == 2)
60         algoritmo = Arrays.asList(11, 10, 9, 8, 7, 6, 5, 4, 3, 2);
61     else
62         throw new IllegalArgumentException("Número de etapa inválido!");
63     return algoritmo;
64 }
65
66 private static void imprimeResultado(List<Integer> cpf) {
67     int cont = 0;
68     for (int numero : cpf) {
69         if (cont == 3 || cont == 6)
70             System.out.print("-");
71         else if (cont == 9)
72             System.out.print("-");
73         System.out.print(numero);
74         cont++;
75     }
76 }
77 }

```