

Desafios



Sites:

- <https://codingbat.com/java>
- <https://www.codewars.com/>
- <https://www.hackerrank.com/>

Desafio blue Ticket

Você tem um bilhete de loteria azul, com as entradas a, b e c. Isso cria três pares, que chamaremos de ab, bc e ac. Considere a soma dos números em cada par.

Se qualquer par somar exatamente 10, o resultado será 10. Caso contrário, se a soma ab for exatamente 10 a mais que as somas bc ou ac, o resultado será 5. Caso contrário, o resultado será 0.

blueTicket(9, 1, 0) → 10

blueTicket(9, 2, 0) → 0

blueTicket(6, 1, 4) → 10

Expected	Run		
blueTicket(9, 1, 0) → 10	10	OK	
blueTicket(9, 2, 0) → 0	0	OK	
blueTicket(6, 1, 4) → 10	10	OK	
blueTicket(6, 1, 5) → 0	0	OK	
blueTicket(10, 0, 0) → 10	10	OK	
blueTicket(15, 0, 5) → 5	5	OK	
blueTicket(5, 15, 5) → 10	10	OK	
blueTicket(4, 11, 1) → 5	5	OK	
blueTicket(13, 2, 3) → 5	5	OK	
blueTicket(8, 4, 3) → 0	0	OK	
blueTicket(8, 4, 2) → 10	10	OK	
blueTicket(8, 4, 1) → 0	0	OK	
other tests		OK	

Desafio blue Ticket

```
public static int blueTicket(int a, int b, int c) {  
  
}
```

Expected	Run		
blueTicket(9, 1, 0) → 10	10	OK	
blueTicket(9, 2, 0) → 0	0	OK	
blueTicket(6, 1, 4) → 10	10	OK	
blueTicket(6, 1, 5) → 0	0	OK	
blueTicket(10, 0, 0) → 10	10	OK	
blueTicket(15, 0, 5) → 5	5	OK	
blueTicket(5, 15, 5) → 10	10	OK	
blueTicket(4, 11, 1) → 5	5	OK	
blueTicket(13, 2, 3) → 5	5	OK	
blueTicket(8, 4, 3) → 0	0	OK	
blueTicket(8, 4, 2) → 10	10	OK	
blueTicket(8, 4, 1) → 0	0	OK	
other tests		OK	

Desafio cigar Party

Quando esquilos se reúnem para uma festa, eles gostam de ter charutos. Uma festa de esquilo é bem-sucedida quando o número de charutos está entre 40 e 60, inclusive. A menos que seja o fim de semana, caso em que não há limite superior no número de charutos.

Retorne true se a parte com os valores fornecidos for bem-sucedida ou false caso contrário.

cigarParty(30, false) → false













cigarParty(50, false) → true

cigarParty(70, true) → true

Expected	Run		
cigarParty(30, false) → false	false	OK	
cigarParty(50, false) → true	true	OK	
cigarParty(70, true) → true	true	OK	
cigarParty(30, true) → false	false	OK	
cigarParty(50, true) → true	true	OK	
cigarParty(60, false) → true	true	OK	
cigarParty(61, false) → false	false	OK	
cigarParty(40, false) → true	true	OK	
cigarParty(39, false) → false	false	OK	
cigarParty(40, true) → true	true	OK	
cigarParty(39, true) → false	false	OK	
other tests		OK	

Desafio cigar Party

```
public boolean cigarParty(int cigars, boolean isWeekend) {  
  
}
```

Expected	Run		
cigarParty(30, false) → false	false	OK	
cigarParty(50, false) → true	true	OK	
cigarParty(70, true) → true	true	OK	
cigarParty(30, true) → false	false	OK	
cigarParty(50, true) → true	true	OK	
cigarParty(60, false) → true	true	OK	
cigarParty(61, false) → false	false	OK	
cigarParty(40, false) → true	true	OK	
cigarParty(39, false) → false	false	OK	
cigarParty(40, true) → true	true	OK	
cigarParty(39, true) → false	false	OK	
other tests		OK	

Desafio sumLimit

Dadas 2 entradas não negativas, a e b, retorne sua soma, desde que a soma tenha o mesmo número de dígitos que a. Se a soma tiver mais dígitos que a, basta retornar a sem b.

(Nota: uma maneira de calcular o número de dígitos de um int n não negativo é convertê-lo em uma string com `String.valueOf (n)` e depois verificar o comprimento da string.)

sumLimit(2, 3) → 5

sumLimit(8, 3) → 8

sumLimit(8, 1) → 9

Expected	Run		
sumLimit(2, 3) → 5	5	OK	
sumLimit(8, 3) → 8	8	OK	
sumLimit(8, 1) → 9	9	OK	
sumLimit(11, 39) → 50	50	OK	
sumLimit(11, 99) → 11	11	OK	
sumLimit(0, 0) → 0	0	OK	
sumLimit(99, 0) → 99	99	OK	
sumLimit(99, 1) → 99	99	OK	
sumLimit(123, 1) → 124	124	OK	
sumLimit(1, 123) → 1	1	OK	
sumLimit(23, 60) → 83	83	OK	
sumLimit(23, 80) → 23	23	OK	
sumLimit(9000, 1) → 9001	9001	OK	
sumLimit(90000000, 1) → 90000001	90000001	OK	
sumLimit(9000, 1000) → 9000	9000	OK	
other tests		OK	

Desafio sumLimit

```
public int sumLimit(int a, int b) {  
  
}
```

Expected	Run	OK	
sumLimit(2, 3) → 5	5	OK	
sumLimit(8, 3) → 8	8	OK	
sumLimit(8, 1) → 9	9	OK	
sumLimit(11, 39) → 50	50	OK	
sumLimit(11, 99) → 11	11	OK	
sumLimit(0, 0) → 0	0	OK	
sumLimit(99, 0) → 99	99	OK	
sumLimit(99, 1) → 99	99	OK	
sumLimit(123, 1) → 124	124	OK	
sumLimit(1, 123) → 1	1	OK	
sumLimit(23, 60) → 83	83	OK	
sumLimit(23, 80) → 23	23	OK	
sumLimit(9000, 1) → 9001	9001	OK	
sumLimit(90000000, 1) → 90000001	90000001	OK	
sumLimit(9000, 1000) → 9000	9000	OK	
other tests		OK	

Desafio luckySum

Dados 3 valores int, a b c, retorne sua soma. No entanto, se um dos valores for 13, ele não conta para a soma o valor e nem os valores à direita.

Assim, por exemplo, se b for 13, então b e c não contam.

luckySum(1, 2, 3) → 6

luckySum(1, 2, 13) → 3

luckySum(1, 13, 3) → 1

Expected	Run		
luckySum(1, 2, 3) → 6	6	OK	
luckySum(1, 2, 13) → 3	3	OK	
luckySum(1, 13, 3) → 1	1	OK	
luckySum(1, 13, 13) → 1	1	OK	
luckySum(6, 5, 2) → 13	13	OK	
luckySum(13, 2, 3) → 0	0	OK	
luckySum(13, 2, 13) → 0	0	OK	
luckySum(13, 13, 2) → 0	0	OK	
luckySum(9, 4, 13) → 13	13	OK	
luckySum(8, 13, 2) → 8	8	OK	
luckySum(7, 2, 1) → 10	10	OK	
luckySum(3, 3, 13) → 6	6	OK	
other tests		OK	

Desafio luckySum

```
public int luckySum(int a, int b, int c) {  
  
}
```

Expected	Run		
luckySum(1, 2, 3) → 6	6	OK	
luckySum(1, 2, 13) → 3	3	OK	
luckySum(1, 13, 3) → 1	1	OK	
luckySum(1, 13, 13) → 1	1	OK	
luckySum(6, 5, 2) → 13	13	OK	
luckySum(13, 2, 3) → 0	0	OK	
luckySum(13, 2, 13) → 0	0	OK	
luckySum(13, 13, 2) → 0	0	OK	
luckySum(9, 4, 13) → 13	13	OK	
luckySum(8, 13, 2) → 8	8	OK	
luckySum(7, 2, 1) → 10	10	OK	
luckySum(3, 3, 13) → 6	6	OK	
other tests		OK	