Questão **2**

Correto

Atingiu 2,00 de 2,00

Converter tuplas

Escreva uma função chamada **convert** que receberá uma lista de tuplas com dois valores, chave e valor, e retornará um dicionário, acumulando todos os valores de chaves iguais em uma lista, como nos exemplos.

Entrada

Não há entrada de dados, o teste chama a função **convert** para uma lista de tuplas.

Saída

Não há saída explícita e a função deverá retornar a lista atualizada. Não imprima nada, apenas retorne a resposta.

Observação

- Observando as entradas do testes, implemente somente a função **convert(l)** .
- No primeiro caso de teste, a lista inicial tem um tupla com chave igual a 3, duas tuplas com chave igual a 4 e três tuplas com chave igual a 1. Portanto o resultado é um dicionário com três pares chave-valor.
- Submeta somente o que foi solicitado.

Particularidade do Tópico

Atenção, a criação de uma função com o nome determinado pelo enunciado é fundamental para a prática do aluno e o Moodle irá descontar pontos, caso a criação não tenha sido feita corretamente (sendo ***case-sensitive*** o nome da função).

For example:

Test	Result

`l = [(3, 91), (4, 69), (1, 85), (1, 96), (1, 7), (4, 94)]`
`resposta = convert(l)`
`print(resposta)`
	`{3: [91], 4: [69, 94], 1: [85, 96, 7]}`

`l = [(1, 69), (2, 94), (2, 15), (4, 59), (4, 65), (4, 17)]`
`resposta = convert(l)`
`print(resposta)`
	`{1: [69], 2: [94, 15], 4: [59, 65, 17]}`

`l = [(4, 63), (4, 60), (2, 4), (4, 18), (4, 62)]`
`resposta = convert(l)`
`print(resposta)`
	`{4: [63, 60, 18, 62], 2: [4]}`

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
def convert(l):
    dic = {}
    for tupla in l:
        if tupla[0] in dic.keys():
            dic[tupla[0]] = dic[tupla[0]] +[tupla[1]]
        else:
            dic[tupla[0]] = [tupla[1]]

    return dic

```

PRECHECK VERIFICAR

	Test	Expected	Got	

✓	`l = [(3, 91), (4, 69), (1, 85), (1, 96), (1, 7), (4, 94)]`
	`resposta = convert(l)`
	`print(resposta)`
		`{3: [91], 4: [69, 94], 1: [85, 96, 7]}`	`{3: [91], 4: [69, 94], 1: [85, 96, 7]}`	✓

✓	`l = [(1, 69), (2, 94), (2, 15), (4, 59), (4, 65), (4, 17)]`
	`resposta = convert(l)`
	`print(resposta)`
		`{1: [69], 2: [94, 15], 4: [59, 65, 17]}`	`{1: [69], 2: [94, 15], 4: [59, 65, 17]}`	✓

✓	`l = [(4, 63), (4, 60), (2, 4), (4, 18), (4, 62)]`
	`resposta = convert(l)`
	`print(resposta)`
		`{4: [63, 60, 18, 62], 2: [4]}`	`{4: [63, 60, 18, 62], 2: [4]}`	✓

✓	`l = [(3, 15), (1, 11), (4, 98), (2, 32), (2, 95), (3, 62), (1, 67), (3, 77), (2, 37), (2, 33)]`
	`resposta = convert(l)`
	`print(resposta)`
		`{3: [15, 62, 77], 1: [11, 67], 4: [98], 2: [32, 95, 37, 33]}`	`{3: [15, 62, 77], 1: [11, 67], 4: [98], 2: [32, 95, 37, 33]}`	✓

✓	`l = [(1, 287), (19, 236), (16, 290), (19, 511), (10, 286), (16, 957), (12, 640), (5, 937), (5, 603), (8, 131), (16, 741), (4, 455), (18, 731), (11, 400), (7, 771), (2, 843), (12, 341), (8, 79), (17, 531), (16, 278), (1, 918), (6, 385), (7, 706), (17, 659), (8, 302), (2, 991), (2, 577), (2, 906), (9, 978), (11, 674), (10, 628), (16, 563), (4, 65), (2, 37), (19, 945), (11, 989), (15, 799), (19, 590), (11, 598), (13, 270), (8, 579), (7, 637), (18, 704), (7, 820), (14, 714), (1, 373), (18, 750), (18, 751), (15, 917), (18, 416), (16, 174), (13, 379), (2, 325), (5, 832), (16, 76), (16, 440), (15, 797), (16, 954), (5, 622), (19, 897), (8, 839), (12, 751), (14, 764), (1, 654), (1, 637), (10, 51), (6, 727), (15, 783), (17, 384), (17, 961), (15, 220), (5, 97), (5, 869), (9, 386), (3, 22), (14, 397), (7, 477), (5, 841), (19, 237), (16, 138), (7, 251), (18, 462), (17, 400), (14, 680), (13, 155), (9, 828), (2, 336), (18, 143), (8, 478), (19, 337), (7, 877), (12, 736), (10, 174), (6, 317), (16, 431), (4, 622), (10, 536), (19, 88), (15, 306), (10, 4)]`
	`resposta = convert(l)`
	`print(resposta)`
		`{1: [287, 918, 373, 654, 637], 19: [236, 511, 945, 590, 897, 237, 337, 88], 16: [290, 957, 741, 278, 563, 174, 76, 440, 954, 138, 431], 10: [286, 628, 51, 174, 536, 4], 12: [640, 341, 751, 736], 5: [937, 603, 832, 622, 97, 869, 841], 8: [131, 79, 302, 579, 839, 478], 4: [455, 65, 622], 18: [731, 704, 750, 751, 416, 462, 143], 11: [400, 674, 989, 598], 7: [771, 706, 637, 820, 477, 251, 877], 2: [843, 991, 577, 906, 37, 325, 336], 17: [531, 659, 384, 961, 400], 6: [385, 727, 317], 9: [978, 386, 828], 15: [799, 917, 797, 783, 220, 306], 13: [270, 379, 155], 14: [714, 764, 397, 680], 3: [22]}`	`{1: [287, 918, 373, 654, 637], 19: [236, 511, 945, 590, 897, 237, 337, 88], 16: [290, 957, 741, 278, 563, 174, 76, 440, 954, 138, 431], 10: [286, 628, 51, 174, 536, 4], 12: [640, 341, 751, 736], 5: [937, 603, 832, 622, 97, 869, 841], 8: [131, 79, 302, 579, 839, 478], 4: [455, 65, 622], 18: [731, 704, 750, 751, 416, 462, 143], 11: [400, 674, 989, 598], 7: [771, 706, 637, 820, 477, 251, 877], 2: [843, 991, 577, 906, 37, 325, 336], 17: [531, 659, 384, 961, 400], 6: [385, 727, 317], 9: [978, 386, 828], 15: [799, 917, 797, 783, 220, 306], 13: [270, 379, 155], 14: [714, 764, 397, 680], 3: [22]}`	✓

	Test	Expected	Got	

✓	l = [(12, 303), (16, 867), (19, 8), (5, 533), (7, 842), (3, 136), (13, 961), (17, 798), (17, 488), (3, 895), (18, 647), (19, 187), (4, 657), (6, 953), (9, 199), (14, 562), (20, 237), (10, 928), (13, 718), (11, 785), (13, 317), (8, 495), (8, 269), (8, 437), (12, 65), (9, 91), (6, 888), (9, 600), (12, 723), (5, 187), (20, 424), (4, 549), (4, 677), (9, 54), (5, 293), (15, 153), (1, 112), (1, 951), (4, 993), (17, 216), (7, 212), (15, 49), (4, 639), (5, 600), (20, 383), (14, 243), (13, 811), (3, 241), (11, 927), (4, 453), (18, 776), (8, 843), (20, 276), (13, 907), (16, 727), (2, 811), (15, 205), (11, 828), (3, 510), (19, 850), (12, 209), (6, 902), (14, 343), (7, 50), (4, 786), (8, 677), (7, 768), (12, 198), (7, 474), (17, 911), (3, 300), (11, 666), (14, 910), (20, 187), (12, 747), (18, 463), (16, 905), (19, 784), (1, 882), (19, 379), (19, 880), (19, 689), (2, 950), (18, 523), (10, 458), (10, 157), (10, 393), (12, 640), (19, 212), (10, 720), (18, 570), (19, 633), (20, 244), (11, 683), (14, 808), (11, 959), (16, 766), (10, 191), (7, 341), (12, 524)]
	resposta = convert(l)
	print(resposta)
		{12: [303, 65, 723, 209, 198, 747, 640, 524], 16: [867, 727, 905, 766], 19: [8, 187, 850, 784, 379, 880, 689, 212, 633], 5: [533, 187, 293, 600], 7: [842, 212, 50, 768, 474, 341], 3: [136, 895, 241, 510, 300], 13: [961, 718, 317, 811, 907], 17: [798, 488, 216, 911], 18: [647, 776, 463, 523, 570], 4: [657, 549, 677, 993, 639, 453, 786], 6: [953, 888, 902], 9: [199, 91, 600, 54], 14: [562, 243, 343, 910, 808], 20: [237, 424, 383, 276, 187, 244], 10: [928, 458, 157, 393, 720, 191], 11: [785, 927, 828, 666, 683, 959], 8: [495, 269, 437, 843, 677], 15: [153, 49, 205], 1: [112, 951, 882], 2: [811, 950]}	{12: [303, 65, 723, 209, 198, 747, 640, 524], 16: [867, 727, 905, 766], 19: [8, 187, 850, 784, 379, 880, 689, 212, 633], 5: [533, 187, 293, 600], 7: [842, 212, 50, 768, 474, 341], 3: [136, 895, 241, 510, 300], 13: [961, 718, 317, 811, 907], 17: [798, 488, 216, 911], 18: [647, 776, 463, 523, 570], 4: [657, 549, 677, 993, 639, 453, 786], 6: [953, 888, 902], 9: [199, 91, 600, 54], 14: [562, 243, 343, 910, 808], 20: [237, 424, 383, 276, 187, 244], 10: [928, 458, 157, 393, 720, 191], 11: [785, 927, 828, 666, 683, 959], 8: [495, 269, 437, 843, 677], 15: [153, 49, 205], 1: [112, 951, 882], 2: [811, 950]}	✓

Passou em todos os teste! ✓

Correto

Notas para este envio: 2,00/2,00.