Lista de Exercícios 2

Requisitos gerais:

- Cada questão deverá ser feita em seu próprio arquivo **.py**.
- Deverá ser entregue um único arquivo .zip que obviamente deve conter os arquivos .py. Nomeie o arquivo .zip da seguinte forma: <RA>_<primeiro nome>.zip.

1) Faça um programa em Python para construir uma **matriz identidade** ordem 100 (cem linhas e cem colunas) usando NumPy. Caso haja dificuldade na instalação, use https://onecompiler.com/python para testar seu programa antes de entregar (exe_01.py).

Resultado esperado: uma matriz numérica com dez mil elementos, todos os elementos da matriz devem possuir o valor 0, exceto os cem elementos da diagonal principal que devem possuir o valor 1:

```
[[1  0  0 ... 0  0  0]
 [0  1  0 ... 0  0  0]
 [0  0  1 ... 0  0  0]
 ...
 [0  0  0 ... 1  0  0]
 [0  0  0 ... 0  1  0]
 [0  0  0 ... 0  0  1]]
```

2) Escreva um código Python para combinar cada elemento da variável **cartas** com cada elemento da variável **naipes**, formando assim, os elementos de uma lista chamada **baralho52Cartas**:

exe_02.py:

```
cartas = ('A', '2', '3', '4', '5', '6', '7', '8', '9', '10', 'J', 'Q', 'K')
naipes = ('\u2666', '\u2660','\u2665','\u2663')

baralho52Cartas = []
#
# Início do seu código aqui
#


#
# Fim do seu código aqui
#
print(baralho52Cartas)
```

O resultado esperado é:

```
['A♦',  'A♠',  'A♥',  'A♣',  '2♦', '2♠', '2♥', '2♣', '3♦', '3♠', '3♥', '3♣',
 '4♦',  '4♠',  '4♥',  '4♣',  '5♦', '5♠', '5♥', '5♣', '6♦', '6♠', '6♥', '6♣',
 '7♦',  '7♠',  '7♥',  '7♣',  '8♦', '8♠', '8♥', '8♣', '9♦', '9♠', '9♥', '9♣',
 '10♦', '10♠', '10♥', '10♣', 'J♦', 'J♠', 'J♥', 'J♣', 'Q♦', 'Q♠', 'Q♥', 'Q♣',
 'K♦',  'K♠',  'K♥',  'K♣']
```

3) Faça um código para que a lista **baralho54Cartas** contenha todos elementos de **baralho52Cartas** juntamente com os elementos da variável **jokers**. (Altere apenas o valor de baralho54Cartas, não altere o valor de baralho52Cartas nem de jokers)

exe_03.py:

```
baralho54Cartas = []

jokers = (chr(127167), chr(127183))

baralho52Cartas = ['A♦', 'A♠',  'A♥',  'A♣',  '2♦',  '2♠', '2♥', '2♣',
 '3♦', '3♠', '3♥', '3♣', '4♦',  '4♠',  '4♥',  '4♣',  '5♦', '5♠', '5♥', '5♣',
 '6♦', '6♠', '6♥', '6♣', '7♦',  '7♠',  '7♥',  '7♣',  '8♦', '8♠', '8♥', '8♣',
 '9♦', '9♠', '9♥', '9♣', '10♦', '10♠', '10♥', '10♣', 'J♦', 'J♠', 'J♥', 'J♣',
 'Q♦', 'Q♠', 'Q♥', 'Q♣', 'K♦',  'K♠',  'K♥',  'K♣']
#
# Início do seu código aqui
#



#
# Fim do seu código aqui
#
print(baralho54Cartas)
```

O resultado esperado é:

```
['A♦',  'A♠',  'A♥',  'A♣',  '2♦', '2♠', '2♥', '2♣', '3♦', '3♠', '3♥', '3♣',
 '4♦',  '4♠',  '4♥',  '4♣',  '5♦', '5♠', '5♥', '5♣', '6♦', '6♠', '6♥', '6♣',
 '7♦',  '7♠',  '7♥',  '7♣',  '8♦', '8♠', '8♥', '8♣', '9♦', '9♠', '9♥', '9♣',
 '10♦', '10♠', '10♥', '10♣', 'J♦', 'J♠', 'J♥', 'J♣', 'Q♦', 'Q♠', 'Q♥', 'Q♣',
 'K♦',  'K♠',  'K♥',  'K♣',  '🂿',  '🃏']
```

4) Construa um conjunto (SET) chamado **baralhoTruco** contendo apenas as cartas usadas no jogo de truco. Use os elementos preexistentes na lista baralho54cartas.

Elimine as cartas desnecessárias: cartas que começam com 8, 9, 10 e os curingas(joker).

exe_04.py:

```
baralho54cartas = ['A♦', 'A♠',  'A♥',  'A♣',  '2♦',  '2♠', '2♥', '2♣',
 '3♦', '3♠', '3♥', '3♣', '4♦',  '4♠',  '4♥',  '4♣',  '5♦', '5♠', '5♥', '5♣',
 '6♦', '6♠', '6♥', '6♣', '7♦',  '7♠',  '7♥',  '7♣',  '8♦', '8♠', '8♥', '8♣',
 '9♦', '9♠', '9♥', '9♣', '10♦', '10♠', '10♥', '10♣', 'J♦', 'J♠', 'J♥', 'J♣',
 'Q♦', 'Q♠', 'Q♥', 'Q♣', 'K♦',  'K♠',  'K♥', 'K♣', chr(127167), chr(127183)]
#
# Início do seu código aqui
#




#
# Fim do seu código aqui
#
print(baralhoTruco)
```

O resultado esperado é:

```
{'A♦',  'A♠',  'A♥',  'A♣',  '2♦', '2♠', '2♥', '2♣', '3♦', '3♠', '3♥', '3♣',
 '4♦',  '4♠',  '4♥',  '4♣',  '5♦', '5♠', '5♥', '5♣', '6♦', '6♠', '6♥', '6♣',
 '7♦',  '7♠',  '7♥',  '7♣',  'J♦', 'J♠', 'J♥', 'J♣', 'Q♦', 'Q♠', 'Q♥', 'Q♣',
 'K♦',  'K♠',  'K♥',  'K♣'}
```

(Obs.: no conjunto a ordem não importa)

5) Construa um dicionário **dictCartaValor** para associar o valor correspondente de cada carta do jogo de truco. Considere as seguintes informações:

- Use como chave os elementos contidos no conjunto **baralhoTruco**.
- Tradicionalmente, usa-se a ordem Dama-Valete-Rei no jogo, assim, cartas numéricas de **4 a 7** assumem seu próprio valor e outras cartas, assumem:
  - Q(Dama)=8, J(Valete)=9, K(Rei)=10, A(Ás)=11, 2(Dois)=12, 3(Três)=13.

- Em algumas situações do jogo, os naipes influenciam, dessa forma, no valor de cada carta deve ser somado:
  - 0.3 para paus (♣),
  - 0.2 para copas (♥),
  - 0.1 para espadas (♠), e
  - 0.0 para ouro (♦).

- Assim, durante uma partida as cartas poderão ter os valores comparados considerando apenas a parte inteira, ou quando for conveniente, pode-se considerar o valor todo.

exe_05.py:

```
baralhoTruco =  { 'A♦', 'A♠', 'A♥', 'A♣', '2♦', '2♠', '2♥', '2♣',
'3♦', '3♠', '3♥', '3♣', '4♦', '4♠', '4♥', '4♣', '5♦', '5♠', '5♥', '5♣',
'6♦', '6♠', '6♥', '6♣', '7♦', '7♠', '7♥', '7♣', 'J♦', 'J♠', 'J♥', 'J♣',
'Q♦', 'Q♠', 'Q♥', 'Q♣', 'K♦', 'K♠', 'K♥', 'K♣' }

valCarta = { '4': 4,  '5': 5, '6': 6,  '7': 7,
             'Q': 8,  'J': 9, 'K': 10, 'A': 11,
             '2': 12, '3': 13 }

valNaipe = { '♦':0.0, '♠':0.1, '♥':0.2, '♣':0.3 }

dictCartaValor = dict()
#
# Início do seu código aqui
#



#
# Fim do seu código aqui
#
print(dictCartaValor)
```

Resultado esperado (ordem não importa):

```
{'7♥': 7.2, 'A♥': 11.2, '3♥': 13.2, 'A♣': 11.3, '4♦': 4.0, '4♥': 4.2, 'Q♥': 8.2, 'J♣': 9.3, '6♦': 6.0, '2♣': 12.3,
'2♥': 12.2, 'J♦': 9.0, 'A♠': 11.1, '7♦': 7.0, '3♦': 13.0, 'J♠': 9.1, '6♥': 6.2, 'J♥': 9.2, 'Q♣': 8.3, '2♠': 12.1,
'A♦': 11.0, '5♣': 5.3, '3♠': 13.1, '5♠': 5.1, '5♦': 5.0, '7♠': 7.1, '4♠': 4.1, '7♣': 7.3, '6♣': 6.3, 'K♥': 10.2,
'K♣': 10.3, 'Q♠': 8.1, 'K♦': 10.0, '5♥': 5.2, 'Q♦': 8.0, '3♣': 13.3, '2♦': 12.0, 'K♠': 10.1, '6♠': 6.1, '4♣': 4.3}
```

6) Pesquise sobre a função **sample** do módulo **random** (random.sample()) e considere o código abaixo:

linha	Código
01	`import random`
02	
03	`meuBaralho = {'A♦', 'A♠', 'A♥', 'A♣', '2♦', '2♠', '2♥', '2♣',`
04	`'3♦', '3♠', '3♥', '3♣', '4♦', '4♠', '4♥', '4♣',`
05	`'5♦', '5♠', '5♥', '5♣', '6♦', '6♠', '6♥', '6♣',`
06	`'7♦', '7♠', '7♥', '7♣', 'Q♦', 'Q♠', 'Q♥', 'Q♣',`
07	`'J♦', 'J♠', 'J♥', 'J♣', 'K♦', 'K♠', 'K♥', 'K♣'}`
08	
09	`qtdeCartas = 3`
10	`jogador1 = set()`
11	`jogador2 = set()`
12	
13	`for n in range(0, qtdeCartas):`
14	`    carta = random.sample(list(meuBaralho), 1)[0]`
15	`    jogador1.add(carta)`
16	`    meuBaralho.remove(carta)`
17	`    carta = random.sample(list(meuBaralho), 1)[0]`
18	`    jogador2.add(carta)`
19	`    meuBaralho.remove(carta)`
20	
21	`tombo = random.sample(list(meuBaralho), 1)[0]`
22	`meuBaralho.remove(tombo)`
23	
24	`print('Tombo:', tombo)`
25	`print('Cartas do Jogador 1:', jogador1)`
26	`print('Cartas do Jogador 2:', jogador2)`
27	`print('Qtde Restante:', len(meuBaralho))`
28	`print('Restante:', meuBaralho)`
29	
30	

6.a) Descreva em linhas gerais para que serve o código contido no **for** (linha13 à linha19).

6.b) O que faz o segundo parâmetro da função random.sample (ou seja, o que faz o número 1 que aparece nas linhas: 14, 17 e 21)

6.c) O que faz o número zero entre colchetes [0] (nas linhas: 14, 17 e 21), por que é necessário usá-lo?

Obs. Entregue as respostas do exercício 6 também usando um arquivo **.py** (exe_06.py), de forma a facilitar a correção pelo professor no mesmo ambiente.