estudar.com.vc

Python

Resumo e Exercícios P3

Resuminho Teórico

Classes e Objetos

Tudo em python é um *objeto*, objetos são *instâncias* de *classes,* como *int, float, list,* etc.

```
a = 5          # a é um objeto da classe int
b = 2.0        # b é um objeto da classe float
c = [3, 8, 10] # c é um objeto da classe list
```

É possível criar classes novas, que podem ter atributos e métodos. Atributos são variáveis internas a uma classe e todos os objetos tem essas variáveis. Métodos são funções que se aplicam sobre a classe e seus métodos. Existem métodos especiais, como o *__init__(self),* que é o *construtor* da classe ou o *__str__(self)*, que é um método chamado automaticamente quando o objeto é usado como uma string (como num print).

```
class Triangulo:
    def __init__(self, b, h): # construtor
        self.base = b   # b é um parâmetro do construtor, base é atributo da classe
        self.altura = h

    def __str__(self): # Representacao em string da classe
        return "Triangulo com base %.2f e altura %.2f" % (self.base, self.altura)

    def area(self): # Retorna a area do triangulo
        return self.base * self.altura/2

t = Triangulo(3, 4)
t.area()      # 6
t.base        # 3
t.altura = 8
t.area()      # 12
```

Dicionários

Um dicionário funciona como uma lista, porém, em vez de números (índices) associados a algum valor, existem *chaves* associadas a um valor, que podem ser de qualquer tipo básico (int, float, string)

```
dicionario = { 'banana': 2, 'maca': 3, 'uva': 5 }
dicionario['maca'] = 4
dicionario['banana'] # 2
dicionario['maca']   # 4
# Alguns metodos de dicionarios
dicionario.keys() # ['banana', 'maca', 'uva']
for k, v in dicionario.items(): # .items() retorna pares de (chave, valor)
    print(k, v) # Imprime 'banana 2', 'maca 3' e 'uva 5'
```

Exercício:

1. War

PSub 2015 – Questão 2

Nesta questão você escreverá um programa em Python que simula uma partida de uma versão simplificação de um jogo de cartas chamado War.

War é um jogo entre dois jogadores. No início de uma partida de War as cartas de um baralho são embaralhadas. Em seguida, cada um dos dois jogadores recebe metade das 52 cartas do baralho. Cada carta recebida por um jogador é colocada em um monte à sua frente. Cada partida do jogo é composta por 26 rodadas. Em cada rodada, simultaneamente, cada jogador pega a carta que está no topo do seu monte e a mostra ao adversário. O vencedor da rodada é o jogador que tiver a carta de maior valor. Em uma rodada pode haver empate. O vencedor da partida é o jogador que vencer mais rodadas. Em uma partida também pode haver empate. Nesta questão, os jogadores da partida de War simulada pelo seu programa serão dois físicos ilustres do Instituto de Tecnologia da Califórnia (Caltech): Sheldon Cooper (Sheldon) e Leonard Hofstadter (Leonard).

A seguir é exibido o resultado de uma partida de War jogada entre Sheldon e Leonard.

```
Sheldon: ♠K ♠Q ♠J ♦5 ♥3 ♦8 ♥2 ♠D ♦A ♣A ♦Q ♦7 ♥K ♣6 ♦9 ♣Q ♠6 ♣2 ♥7 ♥D ♦6 ♥9 ♣J ♠8 ♦D ♥J
Leonard: ♦4 ♣9 ♣D ♠2 ♠3 ♠4 ♣4 ♦3 ♣7 ♣K ♥5 ♥A ♥4 ♣5 ♦K ♠A ♦2 ♠9 ♠5 ♦J ♣3 ♠7 ♥8 ♥Q ♥6 ♣8
Venceu : S  S  S  S  E  S  L  S  S  S  S  L  S  S  L  L  S  L  S  L  S  S  S  L  S  S
```

As constantes definidas abaixo devem ser obrigatoriamente utilizadas nessa questão, sem precisar redefini-las. O valor de cada carta depende apenas de seu posto: 'A', '2', '3', '4', '5', '6', '7', '8', '9', 'D', 'J', 'Q', e 'K'. O dicionário VALOR tem como chave o posto de cada carta e apresenta o seu valor.

```
NAIPES  = ['♥', '♣', '♦', '♠']
POSTOS  = ['A', '2', '3', '4', '5', '6', '7', '8', '9', 'D', 'J', 'Q', 'K']
VALOR   = {'A': 14, '2': 2, '3': 3, '4': 4, '5': 5, '6': 6, '7': 7, '8': 8, '9': 9, 'D': 10,
'J': 11, 'Q': 12, 'K': 13}
```

a) Escreva uma classe **Carta** que será usada para criar um objeto que representa uma carta de um baralho. Essa classe deve obedecer as especificações listadas a seguir.

- Construtor: deve receber como parâmetros um string naipe (da lista NAIPES) e um string posto (da lista POSTOS) que serão usados para criar a representação de uma carta.
- A classe deve possuir um método pegue_naipe() que retorna o naipe de um objeto da classe.
- A classe deve possuir um método pegue_posto() que retorna o posto de um objeto da classe.
- A classe deve possuir um método pegue_valor() que retorna o valor de um objeto da classe. Esse método deve utilizar o dicionário VALOR.

b) Escreva uma classe **Baralho** que será usada para criar um objeto que representa um baralho. Essa classe deve obedecer as especificações listadas a seguir.

- Construtor: o construtor da classe deve utilizar uma lista de objetos da classe Carta. A representação de cada uma das 52 cartas de um baralho deve estar presente na lista.
- A classe deve possuir um método embaralhe() que embaralha a lista de cartas de um dado objeto da classe. Para isso utilize a função random.shuffle() que recebe uma lista e embaralha os seu elementos. Não escreva a função random.shuffle(), apenas use-a.
- A classe deve possuir um método pegue_carta() que remove e retorna a carta no final da lista de cartas de um dado objeto da classe.

c) Escreva uma classe **Jogador** que será usada para criar um objeto que representa um jogador de War. Essa classe deve obedecer as especificações listadas a seguir.

- Construtor: recebe como parâmetro um string nome com o nome do jogador. Um jogador deve ser representado através de um string com o seu nome e uma lista contendo a representação das cartas no seu monte. No momento da criação, o monte de cartas do jogador está vazio. Cada uma das cartas no monte do jogador deve ser representada através de objetos da classe Carta.
- A classe deve possuir um método que permita que ao utilizarmos a função print() com um objeto da classe Jogador, o nome do jogador e todas as cartas no seu monte sejam mostrados como no exemplo acima.
- A classe deve possuir um método recebe_carta() que recebe como parâmetro um objeto carta da classe Carta e põe a carta no topo do monte de cartas de um dado objeto da classe.
- A classe deve possuir um método pegue_carta() que remove e retorna a carta no topo do monte de cartas de um dado objeto da classe.

d) Neste item você deverá escrever uma função main() que simula uma partida de War entre os jogadores Sheldon e Leonard. A sua função deverá utilizar objetos das classes Carta, Baralho e Jogador. A seguir está um exemplo de execução do programa.

```
War entre Sheldon e Leonard
Sheldon: ♣Q ♥6 ♥5 ♥7 ♦8 ♦7 ♥2 ♠K ♠4 ♠6 ♥D ♠D ♥4 ♣D ♣A ♥K ♦4 ♣6 ♦3 ♦J ♠Q ♦2 ♠J ♦K ♣J ♣5
Leonard: ♣4 ♦Q ♠A ♥3 ♠8 ♣9 ♣2 ♠7 ♠2 ♦9 ♠3 ♦6 ♥8 ♠5 ♣7 ♣3 ♥9 ♥Q ♥A ♦D ♦A ♣8 ♠9 ♣K ♦5 ♥J
Venceu :  S  L  L  S  E  L  E  S  S  L  S  S  L  S  S  S  L  L  L  S  L  L  S  E  S  L
Sheldon venceu a partida.
```

Na simulação de uma partida um baralho deve ser criado e embaralhado. Cada um dos jogadores deve receber metade das 52 cartas do baralho. Cada carta recebida por um jogador é colocada no seu monte. Linhas com os nomes dos jogadores seguidos pelas cartas nos seus montes devem ser impressas, como mostra o exemplo. Em cada uma das rodadas, as cartas no topo dos montes dos jogadores são comparadas. No final da partida devem impressas duas linhas: uma linha contendo o rótulo Venceu: seguido das marcas correspondentes aos vencedor de cada rodada (S ou L) ou ainda a marca representando empate (E) e uma linha indicando o jogador que venceu a partida ou se houve empate.

A seguir estão alguma constantes que podem ser usadas pela sua função `main()`.

```
SHELDON = 'S'
LEONARD = 'L'
EMPATE  = 'E'

N_CARTAS   = 52
N_RODADAS  = 26
```

Gabarito:

War

```
import random

NAIPES  = ['♥', '♣', '♦', '♠']
POSTOS  = ['A', '2', '3', '4', '5', '6', '7', '8', '9', 'D', 'J', 'Q', 'K']
VALOR   = {'A': 14, '2': 2, '3': 3, '4': 4, '5': 5, '6': 6, '7': 7, '8': 8, '9': 9, 'D': 10,
'J': 11, 'Q': 12, 'K': 13}

# resultados de uma rodada
SHELDON = 'S'
LEONARD = 'L'
EMPATE  = 'E'

# número de cartas e rodadas
N_CARTAS   = 52
N_RODADAS  = 26

# a
class Carta:
    def __init__(self, naipe, posto):
        self.naipe = naipe
        self.posto = posto

    def pegue_naipe(self):
        return self.naipe

    def pegue_posto(self):
        return self.posto

    def pegue_valor(self):
        return VALOR[self.posto]

# b
class Baralho:
```

```
    def __init__(self):
        # Adiciona todas as combinações de naipe e posto no baralho
        self.baralho = []
        for naipe in NAIPES:
            for posto in POSTOS:
                carta = Carta(naipe,posto)
                self.baralho.append(carta)

    def embaralhe(self):
        # Utiliza a função fornecida para embaralhar as cartas
        random.shuffle(self.baralho)

    def pegue_carta(self):
        # pop() é um método da classe list que retorna o último elemento da lista e o remove
dela
        carta = self.baralho.pop()
        return carta

# c
class Jogador:
    def __init__(self, nome):
        self.nome  = nome
        self.monte = []

    def __str__(self):
        # Monta uma string como especificado e a retorna
        no_cartas = len(self.monte)
        s = "%s: " %(self.nome)
        for i in range(no_cartas-1,-1,-1):
            carta = self.monte[i]
            s += carta.pegue_naipe() + carta.pegue_posto() + ' '
        return s

    def recebe_carta(self,carta):
        self.monte.append(carta)

    def pegue_carta(self):
        carta = self.monte.pop()
        return carta

# d
def main():
    print("War entre Sheldon e Leonard")

    # Cria os jogadores da Classe 'Jogador', passando seus nomes para o construtor
    sheldon = Jogador("Sheldon")
    leonard = Jogador("Leonard")
```

```
    baralho = Baralho() # Cria o baralho
    baralho.embaralhe() # Embaralha as cartas do baralho

    # Distribui as cartas para os jogadores
    for rodadas in range(N_RODADAS):
        carta = baralho.pegue_carta()
        sheldon.recebe_carta(carta)
        carta = baralho.pegue_carta()
        leonard.recebe_carta(carta)

    # Imprime as cartas dos jogadores
    # Aqui, o método __str__ da classe Jogador é usado automaticamente
    print(sheldon)
    print(leonard)

    # Conta as cartas para determinar o vencedor
    cont_sheldon = 0 # contador de rodadas vencidas por sheldon
    cont_leonard = 0 # contador de rodadas vencidas por leonard
    vencedores = 'Venceu :' # String com os vencedores
    for rodada in range(N_RODADAS):
        # Pega as cartas dos montes e os valores
        carta_sheldon = sheldon.pegue_carta()       # Pega a ultima carta
        valor_sheldon = carta_sheldon.pegue_valor() # Pega o valor dessa carta
        carta_leonard = leonard.pegue_carta()
        valor_leonard = carta_leonard.pegue_valor()

        # Verifica o vencedor
        if valor_sheldon == valor_leonard:
            vencedores += '  ' + EMPATE
        elif valor_sheldon  > valor_leonard:
            vencedores += '  ' + SHELDON
            cont_sheldon += 1
        else:
            vencedores += '  ' + LEONARD
            cont_leonard += 1

    # Imprime a string com os vencedores
    print(vencedores)

    # Imprime a string indicando o vencedor
    if   cont_sheldon == cont_leonard:
        print("Sheldon e Leonard empataram a partida.")
    if   cont_sheldon > cont_leonard:
        print("Sheldon venceu a partida.")
    else: # cont_sheldon < cont_leonard:
        print("Leonard venceu a partida.")

main()
```