INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
CEARÁ

DEPARTAMENTO DE TELEMÁTICA
DISCIPLINA: PROGRAMAÇÃO ORIENTADA A OBJETO
LISTA EXERCICIO

ALUNO: Victor dos Santos Bessa Ribeiro **Data: 07/ 10/2021**

1ª Questão (10 Escores). Associe a cada item da 2ª coluna um valor que corresponde a um item da 1ª coluna.

a)	Permite que um objeto seja usado no lugar de outro.	(C)	Encapsulamento
b)	Define a representação de um objeto.	(H)	Mensagem
c)	Separação de interface e implementação que permite que usuários de objetos possam utilizá-los sem conhecer detalhes de seu código.	(I)	Herança
d)	Possui tamanho fixo.	(A)	Polimorfismo
e)	Instância de uma classe.	(F)	Dependência
f)	Forma de relacionamento entre classes onde objetos são instanciados no código.	(J)	Lista
g)	Forma de relacionamento entre classes implementado por meio de coleções.	(B)	Classe
h)	Forma de chamar um comportamento de um objeto.	(E)	Objeto
i)	Reuso de código na formação de hierarquias de classes.	(G)	Composição
j)	Permite inserções e remoções.	(D)	Array

2ª Questão (10 Escores). Aplique V para as afirmações verdadeiras e F para as afirmações falsas.

a) Métodos construtores devem sempre ser explícitos. (F)

b) A classe **Professor** tem um relacionamento de agregação com a classe **Disciplina.** (V)

c) Quando uma classe possui como atributo uma referência para um objeto temos uma dependência. (V)

d) Membros de classes static existem mesmo quando nenhum objeto dessa classe exista. (V)

e) Um relacionamento ***´tem um´*** é implementado via herança. (F)

f) Uma classe **Funcionário** tem um relacionamento **´é um´** com a classe **Dependente.** (F)

g) Uma classe abstract pode ser instanciada. (F)

h) Relacionamentos TODO-PARTE são tipos de associações. (V)

i) Você implementa uma interface ao subscrever apropriada e concretamente todos os métodos definidos pela interface. (V)

j) Um método ***static*** não é capaz de acessar uma variável de instância. (F)

3ª Questão (40 Escores). Escreva exemplos de código Python onde seja possível identificar os seguintes conceitos de POO.

a) Herança;

```python
class Roupa():
    def _init_(self, tecido, cor, tipo):
       self.tecido = tecido
       self.cor = cor
       self.tipo = tipo
    def printar(self):
        print("Tecido:", self.tecido)
        print("Cor:", self.cor)
        print("Tipo:", self.tipo)

class Batina(Roupa):
    def _init_(self, tecido, cor, tipo, botoes):
        super(Batina, self)._init_(tecido, cor, tipo)
        self.botoes = botoes

    def printar(self):
        super(Batina,self).printar()
        print("Botões:", self.botoes)

batina_do_joão = Batina("Gabardine","Preta","Talar","33")
batina_do_joão.printar()
```

b) Encapsulamento;

```python
class Bilas():
    def _init_(self,num_Bilas):
       self._num_Bilas = num_Bilas

    def perder(self,quantidade):
        self._num_Bilas -= quantidade

    def ganhar(self,quantidade):
        self._quantidade += quantidade

    def getNum_Bilas(self):
        return self._num_Bilas

    def setNum_Bilas(self, atualizado):
        self._num_Bilas = atualizado

juninho = Bilas(50)

juninho.ganhar(10)
print(juninho.getNum_Bilas())

juninho.perder(5)
print(juninho.getNum_Bilas())
```

c) Polimorfismo;

```
class Veiculo():
    def _init_(self, cor, rodas, marca):
       self.cor = cor
       self.rodas = rodas
       self.marca= marca
    def printar(self):
        print("Cor:", self.cor)
        print("Rodas:", self.rodas)
        print("Marca:", self.marca)

class Moto(Veiculo):
    def _init_(self, cor, rodas, marca, tamanho):
        super(Moto, self)._init_(cor, rodas, marca)
        self.tamanho = tamanho

    def printar(self):  # Polimorfismo
        super(Moto,self).printar()
        print("Tamanho:", self.tamanho)

moto1 = Moto("Azul","2","Yamaha","2")
```

d) Variáveis de Instância;

```
class Bilas():
    Saldo = 50 # Variável é comum a todos os objetos e pode ser alterada por qualquer
instancia
    def _init_(self,num_Bilas):
       self._num_Bilas = num_Bilas

    def perder(self,quantidade):
        self._num_Bilas -= quantidade

    def ganhar(self,quantidade):
        self._quantidade += quantidade

    def getNum_Bilas(self):
        return self._num_Bilas

    def setNum_Bilas(self, atualizado):
        self._num_Bilas = atualizado

juninho = Bilas(50)

juninho.ganhar(10)
print(juninho.getNum_Bilas())
```

```
juninho.perder(5)
print(juninho.getNum_Bilas())

print(Bilas.Saldo)
```

e) Métodos construtores

```
class Cachorro():
    def _init_(raca,cor): # Metodo construtor
      self.raca = raca
      self.cor = cor

Wattson = Cachorro("vira lata", "castanho")
```

f) Dependência

g) Associação

h) Relacionamento TODO-PARTE

4ª Questão (20 Escores)

Escreva em Python uma classe Ponto que possui os atributos inteiros x e y. Escreva uma classe Reta que possui dois pontos a e b. Escreva os métodos construtores para a classe Ponto e para a Classe Reta. Escreva os métodos get e set para acessar e alterar os atributos da classe Ponto e da classe Reta. Escreva um método distancia que retorna um valor real da distancia entre os dois pontos da reta.

```
class Ponto():
    def __init__(self, x , y):
        self.x = x
        self.y = y
    def setX(self,atualizado):
        self.x = atualizado
    def setY(self,atualizado):
        self.y = atualizado
    def getX(self):
        return self.x
    def getY(self):
        return self.y
class Reta():
    def __init__(self,ponto1,ponto2):
        self.ponto1 = ponto1
        self.ponto2 = ponto2
    def setPonto1(self,atualizado):
        self.ponto1 = atualizado
    def setPonto2(self,atualizado):
```

```
        self.ponto2 = atualizado
    def getPonto1(self):
        return self.ponto1
    def getPonto2(self):
        return self.ponto2
    def distancia(self):
        componenteX = (self.getPonto2().getX() - self.getPonto1().getX()) **2
        componenteY = (self.getPonto2().getY() - self.getPonto1().getY()) **2
        resultado = (componenteX + componenteY)/2

        return resultado

ponto1 = Ponto(0,1)
ponto2 = Ponto(8,1)

reta = Reta(ponto1,ponto2)

print("Distância:", reta.distancia())
```