Adicionar elemento

Definição

Consiste em adicionar um elemento ao conjunto.

Nesse exemplo é possível criar um conjunto com alguns valores, para isso é preciso inserir um valor.

```
def adicionar(self, valor):
        if valor not in self.conjunto:
            self.conjunto.append(valor)
```

*Primeiro verifica se o valor (um elemento a ser inserido) já existe na lista, caso não exista, ele é inserido.

Remover elemento

Definição

```
def remover(self, valor):
        if valor in self.conjunto:
            self.conjunto.remove(valor)
```

Consiste em remover um elemento do conjunto.
*Verifica se o valor está no conjunto, se a condição for True, remove o elemento.

Verificar pertinência

Definição

Se um elemento a pertence ao conjunto A isso é representado como: $a \in A$. Caso contrário, se a não pertence a A, então representa-se como: $a \notin A$.

```
def pertinencia(self, valor):
        return valor in self.conjunto
```

Verifica se o valor (elemento a ser verificado) existe no conjunto

Continência

Definição

A continência permite introduzir os conceitos de subconjunto e igualdade de conjunto. Se todos os elementos de um conjunto A também são elementos de um conjunto B, então A está contido em B, o que é representado por:

$$A \subseteq B.$$

Isso também é lido como A é subconjunto de B.
Se $A \subseteq B$, mas há $b \in B$ tal que $b \notin A$, então pode-se dizer que A está contido propriamente em B, ou que A é subconjunto próprio de B. Isso é denotado por:

$$A \subset \mathrm{B}$$

```
def __Contido(self, conjunto):
        for e in self.conjunto:
            if not conjunto.pertinencia(e):
                return False
        return True
    def contido(self, conjunto):
        if len(conjunto.conjunto) == len(self.conjunto):
            return self.__Contido(conjunto)
        return False
    def contidoPropriamente(self, conjunto):
        if len(conjunto.conjunto) > len(self.conjunto):
            return self.__Contido(conjunto)
        return False
```

A função __Contido verifica se os elementos de um conjunto está contido no outro, para isso, ele verifica cada elemento de conjunto é pertinente no outro conjunto. A função contido verifica se os dois conjunto (conjunto)

União

Definição

Sejam A e B dois conjuntos. A união entre eles, $A \cup B$, é definida como:

$$A \cup B = \{x \mid x \in A \vee x \in B\}.$$

Considerando a lógica, o conjunto A pode ser definido como $x \in A$ e o conjunto B pode ser definido como $x \in B$. Ou seja, a propriedade de pertinência é utilizada para indicar uma proposição lógica. A união corresponde à operação lógica disjunção (símbolo $\vee$).

```
def uniao(self, subconjunto):
        conjuntoResultante = Conjunto(conjunto=self.conjunto)
        for e in subconjunto.conjunto:
            conjuntoResultante.adicionar(e)
        return conjuntoResultante
```

*Primeiro faz a copia do conjunto principal, executo um for no subconjunto e adiciona cada elemento na variável conjuntoResultante.

Intersecção

Definição

Sejam dois conjuntos A e B. A intersecção entre eles, $A \cap B$ é definida como:

$$A \cap B = \{x \mid x \in A \wedge x \in B\}.$$

A união corresponde à operação lógica conjunção (símbolo $\wedge$)

```
def interseccao(self, subconjunto):
        conjuntoResultante = Conjunto()
        for e in subconjunto.conjunto:
            if self.pertinencia(e):
                conjuntoResultante.adicionar(e)
        return conjuntoResultante
```

Diferença

Definição

Sejam os conjuntos A e B. A diferença dos conjuntos A e B, denotada por $A - B$ é definida como:

$$A - B = A \cap \sim B$$

ou

$$A - B = \{x \mid x \in A \land x \notin B\}$$

```
def diferenca(self, subconjunto):
        conjuntoResultante = Conjunto()
        for e in self.conjunto:
            if not subconjunto.pertinencia(e):
                conjuntoResultante.adicionar(e)
        return conjuntoResultante
```

Complemento

Definição

Considere o conjunto universo U. O complemento de um conjunto $A \subseteq U$, denotado por $\sim A$ é definido como:

$$\sim A = \{x \in U \mid x \notin A\}$$

```
def complementar(self, subconjunto):
        if len(self.conjunto) <= len(subconjunto.conjunto):
            return subconjunto.diferenca(self)
```

Conjunto das partes

Definição

Para qualquer conjunto A sabe-se que:

$$A \subseteq A$$
$$\varnothing \subseteq A$$

Para qualquer elemento $a \in A$, é visível que $\{a\} \subseteq A$ A operação unária chamada conjunto das partes, ao ser aplicada ao conjunto A, resulta no conjunto de todos os subconjuntos de A. Suponha um conjunto A. O conjunto das partes de A (ou conjunto potência), denotado por P(A) ou 2 A, é definido por:

$$\mathrm{P}(A) = \{X \mid X \subseteq A\}$$

```
def conjuntoDasPartes(self):
        conjuntoResultante = Conjunto()
        for i in range(1,len(self.conjunto)+1):
            for e in list(combinations(self.conjunto,i)):
                c = Conjunto()
                c.adicionar(e)
                conjuntoResultante.adicionar(c)
        c = Conjunto()
        conjuntoResultante.adicionar(c)
        return conjuntoResultante
```

Produto cartesiano

Definição

A operação produto cartesiano é uma operação binária que, quando aplicada a doi conjuntos A e B, resulta em um conjunto constituído de sequências de duas componentes (tuplas), sendo que a primeira componente de cada sequência é um elemento de A, e a segunda componente, um elemento de B.

Uma sequência de n componentes, denominada n-upla ordenada (lê-se: ênupla ordenada), consiste de n objetos (não necessariamente distintos) em uma ordem fixa. Por exemplo, uma 2-upla (tupla) ordenada é denominada par ordenado. Um par ordenado no qual a primeira componente é x e a segunda é y é definido como $\langle x, y\rangle$ ou (x, y). Uma n-upla ordenada é definida como:

$$\langle x1 , x2 , x3 , ..., xn\rangle.$$

Uma n-upla ordenada não deve ser confundida com um conjunto, pois a ordem das componentes é importante.

Assim:

$$\langle x, y\rangle \neq \langle y, x\rangle$$

O produto cartesiano dos conjuntos A e B, denotado por $A \times B$ é definido por:

$$A \times B = \{\langle a, b\rangle \mid a \in A \land b \in B\}$$

O produto cartesiano de um conjunto com ele mesmo é definido por:

$$A \times A = A2$$

```
def produtoCartesiano(self, subconjunto):
        conjuntoResultante = Conjunto()
        for e in self.conjunto:
            for e2 in subconjunto.conjunto:
                conjuntoResultante.adicionar((e,e2))
        return conjuntoResultante
```

União disjunta

Definição

Diferentemente da união, que desconsidera repetições de elementos no conjunto resultante, a união disjunta permite que os elementos do conjunto resultante sejam duplicados, uma vez que seja identificada a sua fonte. A união disjunta dos conjuntos A e B, denotada por $A + B$ ou $A \cup B$ é definida como:
$A + B = \{\langle a, A\rangle \mid a \in A\} \cup \{\langle b, B\rangle \mid b \in B\}$

```
def uniaoDisjunta(self, conjunto):
    conjuntoResultante = Conjunto()
    num = len(self.conjunto) if len(self.conjunto) >= len(conjunto.conjunto) else len(conjunto.conjunto)
    for i in range(num):
        try:
            conjuntoResultante.conjunto.append((self.conjunto[i], self.nome))
        except IndexError:
            pass
        try:
            conjuntoResultante.conjunto.append((conjunto.conjunto[i], conjunto.nome))
        except IndexError:
            pass
    return conjuntoResultante
```