Python - Conjuntos

Um conjunto é um dos tipos de dados integrados em Python. Em matemática, conjunto é uma coleção de objetos distintos. O tipo de dados definido é a implementação de um conjunto em Python. Os objetos em um conjunto podem ser de qualquer tipo de dados.

Defina em Python também um tipo de dados de coleção, como lista ou tupla. Porém, não se trata de uma coleção ordenada, ou seja, itens de um conjunto ou não acessíveis pelo seu índice posicional. Um objeto definido é uma coleção de um ou mais objetos imutáveis entre chaves {}.

Exemplo 1

Alguns exemplos de objetos definidos são fornecidos abaixo -

```
s1 = {"Rohan", "Physics", 21, 69.75}
s2 = {1, 2, 3, 4, 5}
s3 = {"a", "b", "c", "d"}
s4 = {25.50, True, -55, 1+2j}
print (s1)
print (s2)
print (s3)
print (s4)
```

Ele produzirá a seguinte **saída** -

```
{'Physics', 21, 'Rohan', 69.75}
{1, 2, 3, 4, 5}
{'a', 'd', 'c', 'b'}
{25.5, -55, True, (1+2j)}
```

O resultado acima mostra que a ordem dos objetos na atribuição não é necessariamente retida no objeto definido. Isso ocorre porque o Python otimiza a estrutura do conjunto para operações de conjunto.

Além da representação literal de set (mantendo os itens entre colchetes), a função set() integrada do Python também constrói o objeto set.

Função definir()

set() é uma das funções integradas. Toma qualquer objeto de sequência (lista, tupla ou string) como argumento e retorna um objeto definido

Sintaxe

```
Obj = set(sequence)
```

Parâmetros

- **sequência** - Um objeto do tipo lista, tupla ou str

Valor de retorno

A função set() retorna um objeto definido da sequência, descartando os elementos repetidos nela.

Exemplo 2

```
L1 = ["Rohan", "Physics", 21, 69.75]
s1 = set(L1)
T1 = (1, 2, 3, 4, 5)
s2 = set(T1)
string = "TutorialsPoint"
s3 = set(string)

print (s1)
print (s2)
print (s3)
```

Ele produzirá a seguinte **saída** -

```
{'Rohan', 69.75, 21, 'Physics'}
{1, 2, 3, 4, 5}
{'u', 'a', 'o', 'n', 'r', 's', 'T', 'P', 'i', 't', 'l'}
```

Exemplo 3

Set é uma coleção de objetos distintos. Mesmo que você repita um objeto na coleção, apenas uma cópia será retida nela.

```
s2 = {1, 2, 3, 4, 5, 3,0, 1, 9}
s3 = {"a", "b", "c", "d", "b", "e", "a"}
print (s2)
print (s3)
```

Ele produzirá a seguinte **saída** -

```
{0, 1, 2, 3, 4, 5, 9}
{'a', 'b', 'd', 'c', 'e'}
```

Exemplo 4

Somente objetos imutáveis podem ser usados para formar um objeto definido. Qualquer tipo de número, string e tupla é permitido, mas você não pode colocar uma lista ou dicionário em um conjunto.

```
s1 = {1, 2, [3, 4, 5], 3,0, 1, 9}
print (s1)
s2 = {"Rohan", {"phy":50}}
print (s2)
```

Ele produzirá a seguinte **saída** -

```
  s1 = {1, 2, [3, 4, 5], 3,0, 1, 9}
     ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
TypeError: unhashable type: 'list'
  s2 = {"Rohan", {"phy":50}}
     ^^^^^^^^^^^^^^^^^^^^^^
TypeError: unhashable type: 'dict'
```

Python gera TypeError com uma mensagem de tipos não laváveis 'list' ou 'dict'. Hashing gera um número exclusivo para um item imutável que permite uma pesquisa rápida na memória do computador. Python possui função hash() integrada. Esta função não é suportada por lista ou dicionário.

Mesmo que objetos mutáveis não sejam armazenados em um conjunto, o próprio conjunto é um objeto mutável. Python possui operadores especiais para trabalhar com conjuntos, e existem diferentes métodos na classe set para realizar operações de adição, remoção e atualização em elementos de um objeto de conjunto.