Lista de Exercício - Exercício VII

Rafael Gonçalves de Oliveira Viana

2º semestre de 2017

VII Qual é a diferença entre um Set, Dicionário e Lista ?. Implemente um exemplo de cada

Set

É uma coleção desordenada de dados, sem elementos duplicados. Usos comuns para isso incluem verificações da existência de objetos em outros sequências e eliminação de items duplicados.

Há uma grande quantidade de setoperações, incluindo union (|), intersecção (&), diferença (-), diferença simetrica ($\hat{}$). Estas são operações incomuns, porém possíveis.

```
# -*- coding: UTF-8 -*-
# Autor: Rafael Viana

#Cria grupo v0
v0=set( (1,1,2,3,5,8,13) )
#Cria grupo v1
v1=set( (2,3,5,7,11,13) )

# União
print v0 | v1
# Intersecção
print v0 & v1
# Diferença
print v0 - v1
#Diferença simétrica
print v0 ^ v1
```

Dicionário

É também conhecidos como "memória associativa", ou "vetor associativo". Diferentemente de sequências que são indexadas por inteiros, dicionário é indexado por uma chave, que podem ser de qualquer tipo imutável (como strings e inteiros). Tuplas de strings e inteiros podem ser utilizado com chave. Listas não podem ser usadas como chaves porque são mutáveis, o melhor exemplo de um dicionário é um conjunto não ordenado de pares chave-valor, onde as chaves são únicas em uma dada instância do dicionário.

Dicionários são delimitados por :{ }. Uma lista de pares 'chave:valor' separada por vírgulas dentro desse delimitadores define a constituição inicial do dicionário. Dessa forma também será impresso o conteúdo de um dicionário em uma seção de depuração.

As principais operações em um dicionário são armazenar e recuperar valores a partir de chaves. Também é possível remover um par chave:valor com o comando del. Se você armazenar um valor utilizando uma chave já presente, o antigo valor será substituído pelo novo. Se tentar recuperar um valor dada uma chave inexistente será gerado um erro.

O método keys() do dicionário retorna a lista de todas as chaves presentes no dicionário, em ordem arbitrária (se desejar ordená-las basta aplicar o método sort() na lista devolvida). Para verificar a existência de uma chave, utilize o método has_key() do dicionário ou a keyword in.

```
# -*- coding: UTF-8 -*-
# Autor: Rafael Viana

# esse dicionario e simples porém muito bom para guardar conjunto
# de chave valores

print "Exemplo 1"
# cria um dicionario utilizando chave valor dentro de { }
DicionarioDelinguagens = {'angular':'Angular é um framework da google',
                          'react':'React é um framework do facebook'}

#imprimir dicionario inteiro
print DicionarioDelinguagens
```

```
# imprimi apenas as chaves angular e react
print DicionarioDelinguagens.keys()
#ordenar dicionario, porem temos que guardar uma lista do dicionario
print "Ordenar Lista de Keys"
lista = DicionarioDelinguagens.keys()
lista.sort()
print lista
# Para verificar a existência de uma chave, utilize o método has_key()
print "Existe Angular no Dicionario: "+\
      str(DicionarioDelinguagens.has_key("angular"))


print "\nExemplo 2"
# outra forma de declarar com dict o dicionario
DicionarioDelinguagens2 = dict([('angular', 'Angular é google'),
                 ('React', 'React é um framework do facebook')])

print DicionarioDelinguagens2
```

Listas

A lista é uma estrutra de dados, que pode ser escrita como uma lista de valores separados por vírgula e entre colchetes. Mais importante, os valores contidos na lista não precisam ser do mesmo tipo.

```
# -*- coding: UTF-8 -*-
# Autor: Rafael Viana

# craindo Lista,
a = ['Python','JavaScript', 'NodeJs',"AngularX", "C#","C++","C","Perl"]
#imprimir sem ordenar
print a
# ordenando lista
a.sort()
#imprimir lista
print a
#remover item da lista
a.remove("C#")
```

Conclusão

Python possui diversas estruturas de dados nativas utilizadas para agrupar outros valores, cada uma delas pode ser aplicada, em situações diferentes, assim aproveitando todos os recursos da linguagem python disponível, porém a estrutura de dados mais versátil é sem duvida a lista.