



universidade de aveiro
escola superior de tecnologia e
gestão de águeda

Introdução à Programação

Dicionários

Dicionários

Um dicionário funciona como uma lista, no entanto os índices não precisam de ser obrigatoriamente inteiros.

Um dicionário contém uma coleção de índices, que são chamados de chaves, e uma coleção de valores.

Cada chave está associada a um único valor.

A associação de uma chave e um valor é chamada de par **chave-valor**.

```
eng2sp = dict()  
>>> eng2sp  
{}
```

A função **dict** cria um novo dicionário sem itens.

As **chavetas** `{}` representam um dicionário vazio. A adição de um elemento ao dicionário é feito da seguinte forma:

```
>>> eng2sp['one'] = 'uno'
```

Esta linha cria um item que mapeia a chave 'one' para o valor 'uno'

Dicionários

A ordem dos itens num dicionário é imprevisível.

Ao criar o dicionário podemos dar-lhe a seguinte ordem

```
>>> eng2sp = {'one': 'uno', 'two': 'dos', 'three': 'tres'}
```

No entanto ao imprimir podemos ter uma ordem diferente

```
>>> eng2sp  
{'one': 'uno', 'three': 'tres', 'two': 'dos'}
```

Isto não é um problema uma vez que os dicionários nunca são indexados com índices inteiros. Nos dicionários usamos as chaves para obter o valor correspondente.

```
>>> eng2sp['two']  
'dos'
```

Se uma chave não estiver definida no dicionário é lançada uma exceção.

```
>>> eng2sp['four']  
KeyError: 'four'
```

Dicionários

A função **len** também funciona com dicionários.

```
>>> eng2sp = {'one': 'uno', 'two': 'dos', 'three': 'tres'}  
>>> len(eng2sp)  
3
```

Também se pode usar o operador **in** com os dicionários.

```
>>> 'one' in eng2sp  
True  
>>> 'uno' in eng2sp  
False
```

Para verificar se um valor aparece como um valor num dicionário podemos usar o método **values**, que retorna uma coleção de valores e em seguida usar o operador **in**:

```
>>> vals = eng2sp.values()  
>>> 'uno' in vals  
True
```

Resumo dos conceitos abordados

Conceito	Descrição	Exemplo em Python
Dicionário (dict)	Estrutura de dados que armazena pares <i>chave:valor</i> .	<code>aluno = {"nome": "Ana", "nota": 18}</code>
Chave	Identificador único usado para aceder a um valor.	<code>"nome"</code>
Valor	Informação associada à chave.	<code>"Ana"</code>
Aceder a um valor	Usar a chave entre parêntises rectos.	<code>aluno["nota"] → 18</code>
Adicionar/alterar elemento	Atribuição direta de um valor a uma chave.	<code>aluno["curso"] = "CS"</code>
Remover elemento	Pode ser com <code>del</code> ou <code>pop()</code> .	<code>del aluno["nota"]</code> ou <code>aluno.pop("nota")</code>
Iterar sobre o dicionário	Usar <code>for</code> em chaves , valores ou pares .	<code>for k, v in aluno.items(): print(k, v)</code>
Métodos úteis	<code>keys()</code> , <code>values()</code> , <code>items()</code> , <code>get()</code> , <code>clear()</code> .	<code>aluno.keys() → dict_keys(['nome', 'curso'])</code>
Verificar existência de chave	Usar o operador <code>in</code> .	<code>"nome" in aluno → True</code>
Dicionários dentro de dicionários.	É possível ter dicionários dentro de dicionários.	<code>turma = {"A1": {"nome": "Ana", "nota": 18}}</code>