

SME0827 - Estruturas de Dados



Tuplas e Dicionários Aula 12



Professor: André C. P. L. F. de Carvalho, ICMC-USP
PAE: Moisés Rocha dos Santos
Monitor: Marília Costa Rosendo Silva



Exemplo

```
>>> t = 12345, 54321, 'alo!', # pode terminar com ,
>>> t[0]                      # indicando de forma
                              # explícita que é uma
>>> t                          # tupla

>>> # Podemos ter tuplas de tuplas
>>> u = t, (1, 2, 3, 4, 5)
>>> u

>>> z = t, 1, 2, 3, 4, 5
>>> z
```

© André de Carvalho - ICMC/USP

4



Aula de hoje

- Introdução
- Tuplas
- Empacotamento e desempacotamento
- Operações para sequências
- Dicionários
 - Métodos
 - Operadores
- Bancos de Dados

© André de Carvalho - ICMC/USP

2



Exemplo

```
>>> t = 12345, 54321, 'alo!', # pode terminar com ,
>>> t[0]                      # indicando de forma
12345                         # explícita que é uma
>>> t                          # tupla
(12345, 54321, 'alo!')

>>> # Podemos ter tuplas de tuplas
>>> u = t, (1, 2, 3, 4, 5)
>>> u
((12345, 54321, 'alo!'), (1, 2, 3, 4, 5))

>>> z = t, 1, 2, 3, 4, 5
>>> z
((12345, 54321, 'alo!'), 1, 2, 3, 4, 5)
```

© André de Carvalho - ICMC/USP

5



Tuplas

- Outro tipo de sequência
 - Também é uma coleção de itens (elementos)
- Sintaxe
 - Sequência de itens entre parênteses, separados por ",", (vírgula)
 - Quando atribuídos, podem vir sem parênteses
- Possíveis usos:
 - Coordenadas para georeferenciamento, valores de registros em um banco de dados, ...

© André de Carvalho - ICMC/USP

3



Tuplas

- Tipo imutável
 - Como em *strings*, não é possível atribuir valores a itens de uma tupla
 - Pode simular atribuição por meio de fatia e concatenação
- Podem ser formadas por itens mutáveis
 - Ex. Tupla de listas
- Observações:
 - Tupla vazia: ()
 - Tupla com apenas um item: (x,)

© André de Carvalho - ICMC/USP

6



Exemplo 1

```
>>> vazio = ()
>>> tup = 'ola'
>>> len(vazio)

>>> len(tup)

>>> tup

>>> a = [1, 2], [3, 4]
>>> a
```

Para indicar que é uma tupla
(necessário quando tem apenas
um item)



Exemplo

```
>>> t = 12345, 54321, 'alo!' # Empacotamento de tupla
>>> x, y, z = t               # Desempacotamento de tupla
>>> x

>>> z

>>> t
```



Exemplo 1

```
>>> vazio = ()
>>> tup = 'ola'
>>> len(vazio)
0
>>> len(tup)
1
>>> tup
('ola',)
>>> a = [1, 2], [3, 4]
>>> a
([1, 2], [3, 4])
```

Para indicar que é uma tupla
(necessário quando tem apenas
um item)



Exemplo

```
>>> t = 12345, 54321, 'alo!' # Empacotamento de tupla
>>> x, y, z = t               # Desempacotamento de tupla
>>> x
12345
>>> z
'alo!'
>>> t
(12345, 54321, 'alo')
```



Tuplas

- Empacotamento de tupla
 - Empacota itens em uma tupla
 - Sempre cria uma tupla
- Desempacotamento de sequência
 - Recupera itens de uma tupla
 - Número de variáveis no lado esquerdo deve ser igual ao número de itens da tupla
- Operações válidas para qualquer tipo sequência



Operadores para sequências

- Comparação de sequências
 - Utiliza operadores convencionais para comparação
 - ==, !=, <, >, <=, >=,
- seq1 + seq2
 - Constrói uma nova sequência (tupla)
 - Concatenando as sequências (tuplas) nos operandos

Exemplo

```
>>> a = (1, 2, 3)
>>> b = (5, 6, 7)
>>> a < a

>>> a < b

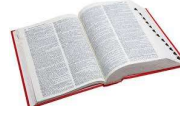
>>> a != b

>>> a + b
```

© André de Carvalho - ICMC/USP 13

Dicionário

- Semelhante a um dicionário onde você pode:
 - Consultar palavras e significados
 - Acrescentar palavras e significados
 - Alterar significados de palavras
 - Excluir palavras ou parte de seus significados



© André de Carvalho - ICMC/USP 16

Exemplo

```
>>> a = (1, 2, 3)
>>> b = (5, 6, 7)
>>> a < a
False
>>> a < b
True
>>> a != b
True
>>> a + b
(1, 2, 3, 5, 6, 7)
```

© André de Carvalho - ICMC/USP 14

Dicionário

- Também pode ser usado como um caderno de contatos, onde você pode:
 - Anotar nome, telefone, *email* e endereço
 - Encontrar *email* procurando pelo nome
- Cada item (entrada) é formado por um par:
 - Valor: um conjunto de valores (informações)
 - Chave: uma palavra-chave (ex.: CPF), que serve de indexador (identificação) de item

© André de Carvalho - ICMC/USP 17

Comparação de sequências

- Segue ordem lexicográfica
 - Comparação começa com os dois primeiros itens das sequências comparadas
 - Se forem diferentes, as sequências são diferentes
 - Senão, passa para os itens seguintes nas sequências

© André de Carvalho - ICMC/USP 15

Dicionário

- Chave e valor
 - Chave pode ser de qualquer tipo imutável
 - Strings e números podem sempre ser chaves
 - Tuplas podem ser chaves se contêm apenas strings, números ou tuplas
 - Listas não podem ser chaves (por ser um tipo mutável)
 - Valor pode ser de qualquer tipo, mutável ou imutável

© André de Carvalho - ICMC/USP 18



Tipos mutáveis x imutáveis

```
>>> b = [12, 34, 3]
>>> b

>>> b[2] = 6
>>> b

>>> b[1] = 2
>>> b
```

```
>>> x = "aulas"
>>> x[1]

>>> x[0] = 'z' # ERRO
```



Dicionários

- Podem ser vistos como um conjunto não ordenado de pares chave-valor
 - Ex.: {'Joao': 3245, 'Pedro': 2311}
 - Dicionário vazio: {}
- Chaves devem ser únicas
 - Armazenar novo item com chave existente escreve sobre o item que tem a mesma chave
- Acessar item com chave que não existe gera mensagem de erro

Chave Valor



Tipos mutáveis x imutáveis

```
>>> b = [12, 34, 3]
>>> b
[12, 34, 3]
>>> b[2] = 6
>>> b
[12, 34, 6]
>>> b[1] = 2
>>> b
[12, 2, 6]
```

```
>>> x = "aulas"
>>> x[1]
'u'
>>> x[0] = 'z' # ERRO

ERRO, pois valores
de um string não
podem ser mudados
```



Exemplo

```
>>> d = {'Andre':3, 'Joao':1}
>>> d
{'Andre': 3, 'Joao': 1}
>>> e = {('Andre', 4):3, ('Joao',5):1}
>>> e
{('Andre', 4): 3, ('Joao', 5): 1}
>>> f = e[('Joao',5)]
>>> f
1
>>> g = e['Joao']
>>> g
ERRO
>>> g = e[('Joao', 1)]
>>> g
ERRO
```



Dicionário

- A chave de cada item deve ser única
 - Mais de um item não pode ter a mesma chave
 - Ex.: CPF, email, nome completo, ...
 - Preferencialmente, chaves devem ser objetos simples
- Em Python, dicionários são instâncias (objetos) da classe *dict*



Exemplo

```
>>> d = {'Andre':3, 'Joao':1}
>>> d
{'Andre': 3, 'Joao': 1}
>>> e = {('Andre', 4):3, ('Joao',5):1}
>>> e
{('Andre', 4): 3, ('Joao', 5): 1}
>>> f = e[('Joao',5)]
>>> f
1
>>> g = e['Joao']
>>> g
ERRO
>>> g = e[('Joao', 1)]
>>> g
ERRO
```



Dicionários

- Valores podem ser
 - Consultados
 - Indexar dicionário com a chave do item a ser consultado
 - Incluídos
 - Atribuir valor, indexando dicionário com uma nova chave
 - Excluídos
 - Elimina itens, utilizar para isso o comando *del*
 - Alterados
 - Dada uma chave existente, alterar valor associado



Métodos pré-definidos

- Método *nomedicionario.items()*
 - Retorna os pares chave-valor de um dicionário
 - Visão de itens
- Método *nomedicionario.keys()*
 - Retorna as chaves do dicionário (em ordem arbitrária)
 - Podem ser ordenadas pelo Método *sorted()*
- Método *nomedicionario.values()*
 - Retorna os valores do dicionário



Exemplo

Incluir novo item

Consultar dicionário

Eliminar item

```
>>> tel = {'joao': 4098, 'leo': 4139}
>>> tel['luis'] = 4127
>>> tel
{'joao': 4098, 'leo': 4139, 'luis': 4127}
>>> tel['joao']
4098
>>> del tel['leo']
>>> tel['ivo'] = 4127
>>> tel
{'joao': 4098, 'luis': 4127, 'ivo': 4127}
```



Pausa



Exemplo

Incluir novo item

Consultar dicionário

Eliminar item

```
>>> tel = {'joao': 4098, 'leo': 4139}
>>> tel['luis'] = 4127
>>> tel
{'joao': 4098, 'leo': 4139, 'luis': 4127}
>>> tel['joao']
4098
>>> del tel['leo']
>>> tel['ivo'] = 4127
>>> tel
{'joao': 4098, 'luis': 4127, 'ivo': 4127}
```



Motivação para uso de BDs

- Dicionários podem ser usados para criar um Banco de Dados, para:
 - Faz a matrícula em disciplinas
 - Retira um livro da biblioteca
 - Realiza operações em um caixa eletrônico
 - Compra um lanche na cantina

Informações precisam ser incluídas ou atualizadas

BD, sem saber você usa



© André de Carvalho - ICMC/USP 31

BDs Relacionais

- Coleção de tabelas
 - Colunas representam as características (atributos) dos dados
 - Todos os dados em uma coluna devem ser do mesmo tipo
 - Cada registro é armazenado em uma linha

	Nome	Endereco	Idade
José	Rua Sol 10	23	
Maria	Av. La	23	
Luiz	Rua Azul 20	57	

Coluna ou campo

Linha ou registro

© André de Carvalho - ICMC/USP 34

O que é um Banco de Dados?

- Definição:
 - Coleção de dados logicamente relacionados que tem algum propósito associado
- Projetado, construído e preenchido com dados
 - Para satisfazer um propósito ou público específico

© André de Carvalho - ICMC/USP 32

Terminologia

- **Relação:** tabela
- **Tupla:** cada linha da tabela (registro)
- **Atributo:** cada coluna da tabela (campo)
- **Valores** de cada atributo são **atômicos**
 - Indivisíveis e mono-valorados
- **Domínio** de um atributo
 - Conjunto de valores atômicos que o atributo pode assumir
 - Ex.: data de nascimento

© André de Carvalho - ICMC/USP 35

O que faz um Banco de Dados?

- Organiza informações por meio de uma estrutura específica
 - Facilita acesso aos e alteração dos dados
- Como organiza?
 - Em um BD, os informações podem ser estruturadas por vários modelos
 - Ex.: relacional (por tabelas)

© André de Carvalho - ICMC/USP 33

Exemplo

Atributo

Domínio(Nome) = cadeia de caracteres (string)

Domínio(Ano) = números com 4 dígitos

Tabela Pessoal

Sobrenome	Nome	Endereco	Cidade	Ano
Silva	José	Rua Sol 10	São Carlos	1960
Novaes	Maria	Av. La 23	São Carlos	1993
Lima	Luiz	Rua Azul 20	Curitiba	1985
Pereira	Pedro	Av. Norte 4	São Paulo	1992

Tupla

© André de Carvalho - ICMC/USP 36



SQL

- Structure Query language
- É uma linguagem de computador que segue o padrão ANSI
- Possui duas linguagens
 - Linguagem de definição de dados (DDL)
 - Cria tabelas
 - Linguagem de manipulação de dados (DML)



SQL DDL

- Principais comandos:
 - CREATE TABLE: cria uma nova tabela de BD
 - ALTER TABLE: altera uma tabela de BD
 - DROP TABLE: deleta uma tabela de BD
 - CREATE INDEX: cria uma chave de busca
 - DROP INDEX: deleta uma chave de busca



SGBDs que usam SQL

- Produtos comerciais
 - Microsoft ACCESS (Microsoft Office)
 - Microsoft SQLserver
 - Oracle
- Freeware
 - **MySQL**
 - PostgreSQL
 - MiniSQL



Criar um BD com MySQL

- Comando MySQL permite a criação de tabelas
 - Sintaxe:

```
CREATE TABLE nome-tabela
(
  nome-coluna1 tipo,
  nome-coluna2 tipo,
  ...
)
```



SQL DDL

- Linguagem de definição de dados SQL
 - *Data Definition Language*
- Permite que tabelas sejam criadas ou eliminadas
- Permite ainda:
 - Definir chaves
 - Especificar ligação entre tabelas
 - Impor restrições entre tabelas de BD



Criar um BD com MySQL

```
DROP TABLE IF EXISTS Pessoa
CREATE TABLE Pessoa
(
  sobrenome varchar ,
  nome varchar,
  endereco varchar,
  cidade varchar,
  ano int
);
```



Criar um BD com MySQL

```
DROP TABLE IF EXISTS Pessoal
CREATE TABLE Pessoal
(
  sobrenome varchar(30),
  nome varchar,
  endereco varchar,
  cidade varchar,
  ano int(4)
);
```

Tamanho
máximo



Consultas MySQL

Consulta:

SELECT sobrenome FROM Pessoal;

Em alguns sistemas, um ";" deve
ser colocado após um comando

Sobrenome	Nome	Endereco	Cidade	Ano
Silva	José	Rua Sol 10	São Carlos	1960
Novaes	Maria	Av. La 23	São Carlos	1993
Lima	Luiz	Rua Azul 20	Curitiba	1985
Pereira	Pedro	Av. Norte 4	São Paulo	1992



SQL DML

- Linguagem de manipulação de dados SQL
 - *Data Manipulation Language*
- Inclui comandos para consultar e atualizar registros de uma BD
 - SELECT: extrai dados de uma tabela de BD
 - UPDATE: atualiza dados em uma tabela de BD
 - DELETE: deleta dados de uma tabela de BD
 - INSERT INTO: insere novos dados em uma tabela de BD



Consultas MySQL

Consulta:

SELECT sobrenome FROM Pessoal;

Em alguns sistemas, um ";" deve
ser colocado após um comando

Sobrenome	Nome	Endereco	Cidade	Ano
Silva	José	Rua Sol 10	São Carlos	1960
Novaes	Maria	Av. La 23	São Carlos	1993
Lima	Luiz	Rua Azul 20	Curitiba	1985
Pereira	Pedro	Av. Norte 4	São Paulo	1992

Sobrenome
Silva
Novaes
Lima
Pereira



Comando SELECT

- Usado para selecionar dados de uma tabela
 - Resultado é armazenado em uma tabela de resultado (conjunto-resultado)
 - Sintaxe:

SELECT nome-coluna(s) FROM nome-tabela



Conclusão

- Introdução
- Tuplas
- Enpacotamento e desempacotamento
- Dicionários
- Bancos de Dados
- MySQL



Perguntas

