UNIVERSIDADE DE
VASSOURAS

UNIVERSIDADE DE VASSOURAS

Curso de Graduação em Engenharia de Software

Aula 5
18 MAR 2022

Laboratório de Programação de Interfaces com o Usuário

Prof. André Saraiva
Mestre em Sistemas de Computação
Especialista em Arquitetura e Projeto de Cloud Computing
Analista Blue Team em Cibersegurança pela Kimoshiro
Tutor EaD pela Universidade Federal Fluminense - UFF

UNIVERSIDADE DE VASSOURAS

Tópicos

Revisão de Python

- Lista
- Conjunto (Set)

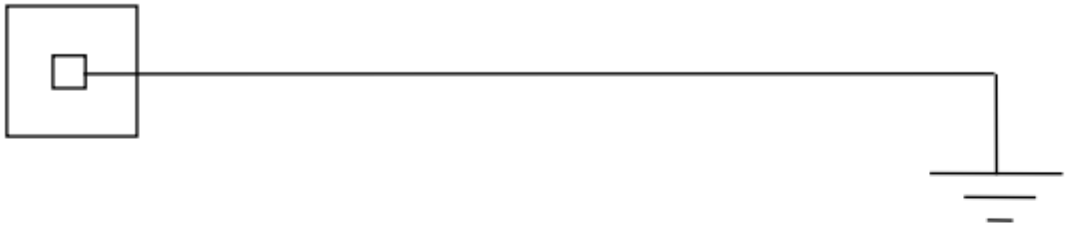
Listas

Uma lista é uma sequência ordenada pelo índice, de zero ou mais referências a objetos (ponteiros para objetos).

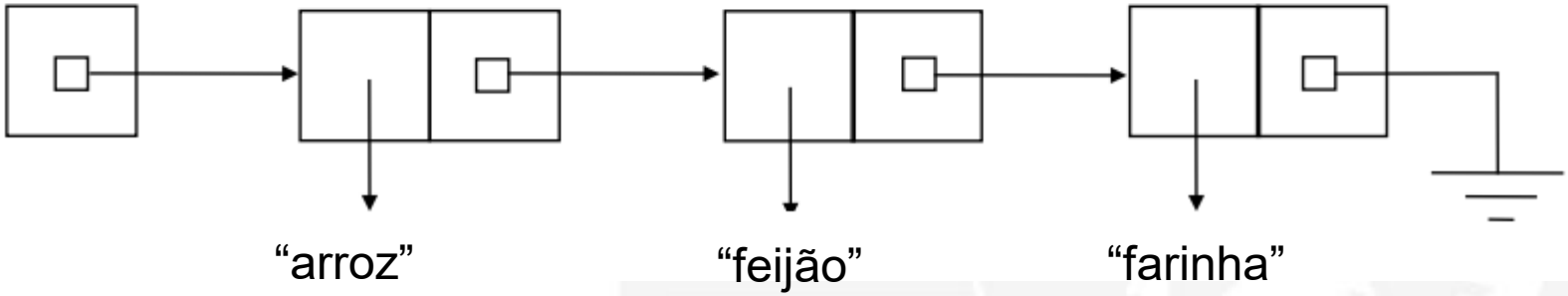
Listas são mutáveis, portanto podem receber novos elementos, substituir elementos existentes ou remover antigos elementos.

Listas

compras = []



compras = [“arroz”, “feijão”, “farinha”]



Operações para inclusão de novos elementos:

- **append(***novoElemento***)**: anexa o *novoElemento* no final da lista;

- **insert(***pos*, *novoElemento***)**: insere o *novoElemento* na posição *pos* da lista. Caso a lista tenha menos que *pos* elementos, o *novoElemento* é inserido no final da lista.

Listas

- **append(***novoElemento***)**: anexa o *novoElemento* no final da lista;

```
compras = [“arroz”, “feijão”, “farinha”]

compras.append(“ovo”)
compras = [“arroz”, “feijão”, “farinha”, “ovo’]
```

Listas

- **insert(***pos*, *novoElemento***)**: insere o *novoElemento* na posição *pos* da lista. Caso a lista tenha menos que *pos* elementos, o *novoElemento* é inserido no final da lista.

compras = [“arroz”, “feijão”, “farinha”, “ovo”]

compras.insert(1, “manteiga”)

compras = [“arroz”, “manteiga”, “feijão”, “farinha”, “ovo’]

Atividade Prática

- **Especificação do Problema:**

Crie uma lista de números que contenha a quantidade de números aleatórios passado pelo usuário e um intervalo de valores escolhidos também pelo usuário.

Exemplo:

Quantidade: 5

Menor valor: 1

Maior valor: 30

Saída: [4, 8, 11, 21, 28]

Operações para remoção de novos elementos:

- **pop()**: retorna e remove o último elemento da lista, o mais a direita.

- **remove(*x*)**: remove a primeira ocorrência do item *x*, da esquerda para a direita.

Operações para remoção de novos elementos:

- **pop()**: retorna e remove o último elemento da lista, o mais a direita.

- **remove(*x*)** ... ncia do item *x*, da esquerda p...

Lança uma exceção "IndexError: pop from empty list", se aplicado a uma lista vazia.

Listas

Operações pa ntos:

- **pop()**: reto ento da lista, o mais a direita.

Lança uma exceção "ValueError" se x não for encontrado.

- **remove(*x*)**: remove a primeira ocorrência do item *x*, da esquerda para a direita.

Listas

Operações úteis para listas:

- ✔ lista**[**pos**]**: retorna o elemento da lista na posição pos.
- ✔ **len(**lista**)**: retorna o comprimento da lista.
- ✔ lista.**count(**elemento**)**: retorna quantas vezes o elemento ocorre na lista.
- ✔ lista.**sort()**: ordena o conteúdo da lista, se os elementos forem todos do mesmo tipo.

Atividade Prática

- **Especificação do Problema:**

Utilizando a atividade 1, gere uma lista de 200 números aleatórios no intervalo 0 a 50. Crie uma função que remova todos os valores duplicados e mostre o conteúdo da lista na saída padrão

Operações de listas:

- compras[*posInicio* : *posAposFim*]: retorna um nova lista composta de referências para os elementos existentes, iniciando-se por elemento na posição *posInicio* e finalizando por elemento na posição anterior ao *posAposFim*

Listas

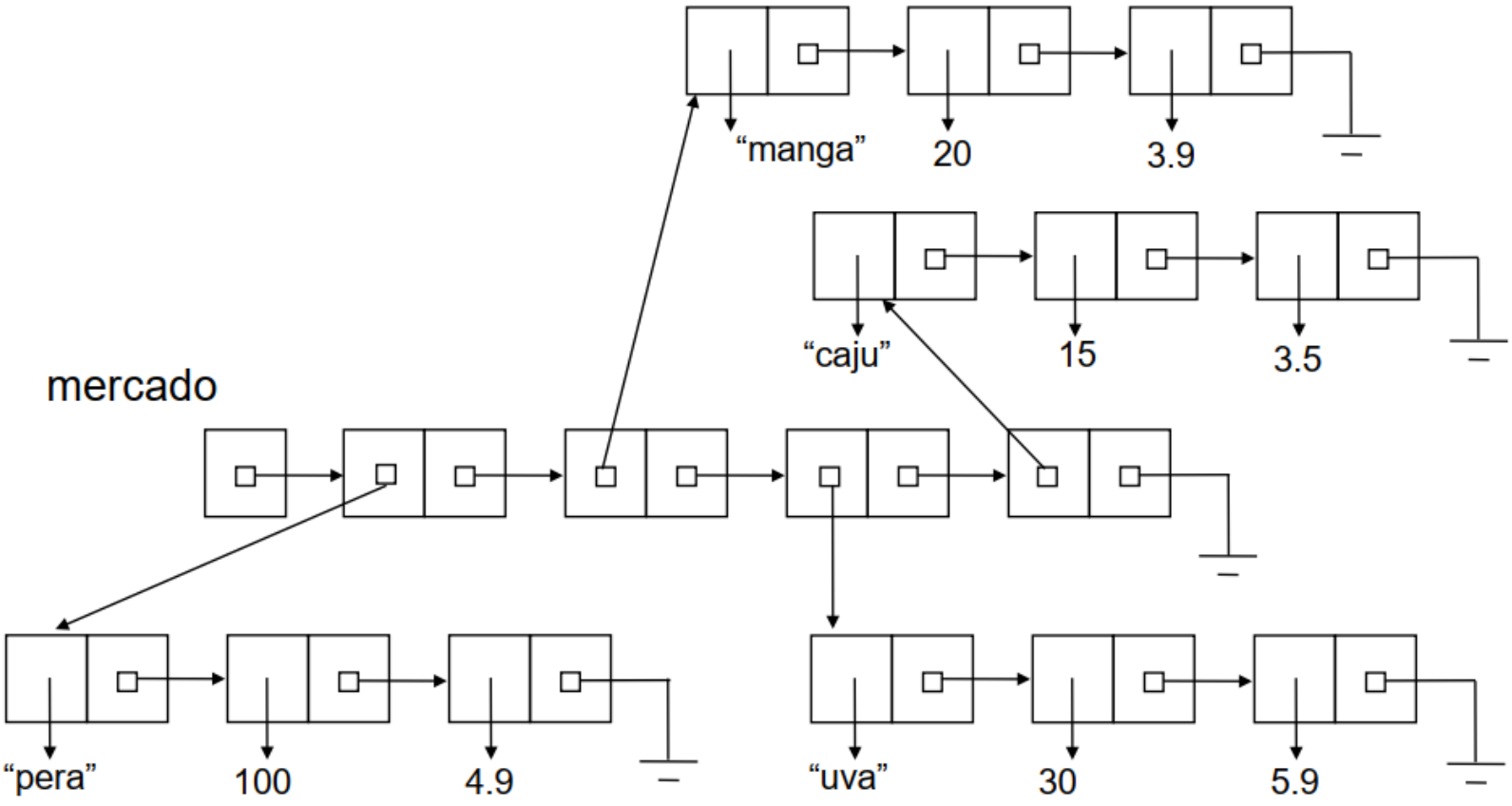
Operações de listas:

```
compras = [“arroz”, “manteiga”, “feijão”, “farinha”, “ovo”]

mercado = compras[1:4]
mercado = [“manteiga”, “feijão”, “farinha”]
```

Listas

mercado = [[“pera”, 100, 4.9], [“manga”, 20, 3.9], [“uva”, 30, 5.9], [“caju”, 15, 3.5]]



Operações de listas:

mercado = [[“pera”, 100, 4.9], [“manga”, 20, 3.9], [“uva”, 30, 5.9], [“caju”, 15, 3.5]]

mercado[1][2] *= 0.5	# manga pela metade do preço
mercado[3][1] -= 10	# caju com dez quilos a menos
mercado.remove([“uva”,30,5.9])	# uva é removido do mercado
mercado.insert(1, [“kiwi”, 200, 1.99])	# o produto kiwi foi é inserida

mercado = [[“pera”, 100, 4.9], [“kiwi”, 200, 1.99], [“manga”, 20, 1.95], [“caju”, 5, 3.5]]

UNIVERSIDADE DE VASSOURAS

Conjuntos (set)

Conjunto (set) é uma estrutura de dados mutável, desordenada e sem elementos repetidos

Diferentemente de vetores, conjuntos não têm seus elementos acessados por índice.

No entanto, conjuntos são iteráveis, podendo seus elementos serem acessados por uma estrutura **for**

Um conjunto pode ser escrito diretamente no vídeo via comando **print**.

Conjuntos (set)

- A função **set()** associa um conjunto vazio a uma variável

```
numeros = set()
```

- A função **add()** adiciona um elemento ao conjunto, caso o elemento ainda não ocorra no conjunto.

```
numeros = set()
numeros.add(5)
numeros.add(15)
numeros.add(25)
numeros.add(35)          # numeros={5, 15, 25, 35}
```

Conjuntos (set)

- A função **add()** adiciona um elemento ao conjunto, caso o elemento ainda não ocorra no conjunto.

```
numeros.discard(15)
# numeros={5, 25, 35}
```

- A função **len()** retorna a cardinalidade do conjunto, isto é, seu tamanho

```
# numeros={5, 25, 35}
print(len(numeros))
```

Conjuntos (set)

✔ ***UNIÃO*** → **s.union(t)**: Retorna um novo conjunto resultante da união de dois conjuntos **s** e **t** é formado por todo elemento que pertence a **s** ou que pertence a **t** (ou a ambos)

{1, 3, 4}.union({1, 2, 4}) = {1, 2, 3, 4}
{1, 3}.union({2, 4}) = {1, 2, 3, 4}
{‘A’, ‘C’, ‘E’}.union({‘B’, ‘C’, ‘D’} = {‘A’, ‘B’, ‘C’, ‘D’, ‘E’}
{‘C’}.union({‘B’, ‘C’, ‘D’} = {‘B’, ‘C’, ‘D’}

Conjuntos (set)

✔ ***INTERSEÇÃO*** → **s.intersection(t)**: Retorna um novo conjunto resultante da interseção de dois conjuntos **s** e **t** é formado por todo elemento que pertence a **s** e que pertence a **t**

{1, 3, 4}.intersection({1, 2, 4}) = {1, 4}
{1, 3}.intersection({2, 4}) = { }
{‘A’, ‘C’, ‘E’ }.intersection({‘B’, ‘C’, ‘D’}) = {‘C’}
{‘C’ }.intersection({‘B’, ‘C’, ‘D’}) = {‘C’}

Conjuntos (set)

✔ ***DIFERENÇA*** → **s.difference(t)**: Retorna um novo conjunto resultante da diferença entre dois conjuntos **s** e **t**. O resultado é formado por todo elemento que pertence a **s** e que não pertence a **t**

{1, 3, 4}.difference({1, 2, 4}) = {3}
{1, 3, 4} - {1, 2, 4} = {3}
{1, 3}.difference({2, 4}) = {1, 3}
{‘A’, ‘C’, ‘E’}.difference({‘B’, ‘C’, ‘D’}) = {‘A’, ‘E’}
{‘C’}.difference({‘B’, ‘C’, ‘D’}) = { }

Atividade Prática para P1

- **Especificação do Problema:**

Escreva um programa que gera e imprime os números primos entre 2 e N, escolhido pelo usuário, usando o algoritmo chamado de "**Crivo de Eratóstenes**"

Prazo: 24/03 às 23:59h

Valor: 1,0 (até 24/03) e 1,5 (até 19/03 às 23:59hs)

UNIVERSIDADE DE VASSOURAS

Próximos Tópicos

Revisão de Python

- Dicionário (dict)
- Arquivos Texto
- Revisão

Contato

Professor:
André Saraiva, MSc

E-mail:
andre.saraiva@universidadedevassouras.edu.br