Introdução à Ciência da Computação

Disciplina: 113913

Adaptado das aulas de APC da Profa. Carla Denise Castanho

Universidade de Brasília – UnB

Instituto de Ciências Exatas – IE

Departamento de Ciência da Computação – CIC

7. LISTAS

Conjuntos Homogêneos

- Imagine que você precisa fazer um programa para calcular as médias finais dos alunos de ICC.
- Para cada aluno, você deve ler as notas de 3 provas. Deve então e calcular e mostrar a média de cada um.

Conjuntos Homogêneos

- Com o que aprendemos até agora, é possível resolver esse problema. No entanto, você precisaria de:

  40 x 3 provas = **120 variáveis**!!

Conjuntos Homogêneos

Exemplo - Variáveis!! Infinitas variáveis!!!

```
Algoritmo NotaICC

Variáveis
    p1_a1, p1_a2, ..., p1_a40 : real
    p2_a1, p2_a2, ..., p2_a40 : real
    p3_a1, p3_a2, ..., p3_a40 : real

Início
    Escreva ("Informe a nota da Prova 1 do Aluno 1:")
    Leia (p1_a1)
    ...
Fim
```

120 VARIÁVEIS!!! ʘ‿ʘ

120 Leituras!!! 40 Cálculos e Escritas!!! ▫

Conjuntos Homogêneos

- E se fosse um programa para calcular as médias da UnB inteira?
- E se você precisar adicionar uma nota? Ou fazer qualquer alteração no cálculo?
- Certamente deve existir um jeito melhor de resolver esse problema...

Conjuntos Homogêneos

- Para trabalhar com muitos dados de **mesmo tipo**, nós usamos a noção de **conjuntos homogêneos**.
- Linguagens de Programação, onde a declaração da tipificação da variável é sintaticamente obrigatória, possuem um tipo de **estrutura de dados** denominado **vetores** para tratar estes conjuntos.
  - O vetor tem um **tamanho** definido (número de elementos) e cada elemento é referenciado por meio de um **índice**.
- Como a tipificação da variável em Python não é obrigatória e é dinâmica, ela oferece uma outra **estrutura de dados** denominada **listas**, **que pode ser usada para implementar vetores**.

Listas

- Por exemplo, para nos referirmos ao **i-ésimo** elemento, usamos **lista[i]** – lê-se *"lista, índice i"*.
- Em Python, o primeiro elemento da lista é a **lista[0]**. Por uma questão de eficiência, **começamos a contar do zero** e vamos até (tamanho da lista -1)!
- Na linguagem Python, por exemplo, um vetor de **40** elementos vai de **lista[0]** até **lista[39]**...
- Vamos usar essa mesma convenção em pseudocódigo.

Listas

- Em Python não se declara variáveis. No entanto, como o pseudocódigo é feito independentemente da linguagem de programação podemos declarar um vetor em pseudocódigo, utilizando a seguinte notação:

Sintaxe para declarar um Vetor

```
<nome> : vetor [<tamanho>] de <tipo>
```

Exemplos

```
Variáveis
    primos : vetor [10] de inteiros
    notas : vetor [100] de reais
    sexo : vetor [30] de caracteres
```

- mesmo tipo!
- Ou de forma mais direta:

Sintaxe para declarar um Vetor

```
<nome> : lista
```

Listas

- Note que a sintaxe para declarar listas é muito simples:

Sintaxe para declarar um Vetor
<nome> **: lista**

- Listas não tem tamanho fixo, é uma estrutura de dados dinâmica (aumenta ou diminui de tamanho de acordo com a necessidade)
- Os tipos da lista de Python são dinâmicos
- Ou seja, dá uma flexibilidade muito grande para o programador
  - Isto significa que se você quer usar uma lista como vetor, cabe a você (programador) cuidar para que a lista contenha dados de apenas um tipo!

Listas - Exemplo

Exemplo - Cálculo de Notas de CB

```
Algoritmo NotaICC_Esperto

Variáveis
    prova1 : lista
    prova2 : lista
    prova3 : lista
    media : lista
    i : inteiro

Início
    Para i ← 0 até 39 faça
        Leia (prova1[i])
        Leia (prova2[i])
        Leia (prova3[i])
    Fim-para
    Para i ← 0 até 39 faça
        media[i] ← (prova1[i] + prova2[i] + prova3[i]) / 3
    Fim-para
    Para i ← 0 até 39 faça
        Escreva ("Média do Aluno ", i+1, ": ", media[i])
    Fim-para
Fim
```

UnB

Listas - Exemplo

Exemplo - Cálculo de Notas de CB

```
Algoritmo NotaICC_Esperto

Variáveis
    prova1 : lista
    prova2 : lista
    prova3 : lista
    media : lista
    i : inteiro

Início
    Para i ← 0 até 39 faça
        Leia (prova1[i])
        Leia (prova2[i])
        Leia (prova3[i])
    Fim-para
    Para i ← 0 até 39 faça
        media[i] ← (prova1[i] + prova2[i] + prova3[i]) / 3
    Fim-para
    Para i ← 0 até 39 faça
        Escreva ("Média do Aluno ", i+1, ": ", media[i])
    Fim-para
Fim
```

DECLARAÇÃO DE VARIÁVEIS
EM PYTHON POSSO CRIAR UMA VARIÁVEL EM QQ LUGAR, MAS SE TENTAR ACESSAR OU MANIPULAR SEM ELA TER SIDO CRIADA VAI DA ERRO. VIDE EXEMPLO MAIS À FRENTE

ENTRADA DE DADOS

PROCESSAMENTO

SAÍDA DE DADOS

UnB

Listas na Linguagem Python

- Para declarar uma lista em Python, utilizamos a seguinte notação:

Sintaxe para declarar uma Lista em Python
<nome> **= []**

Exemplos

`primos = []`
`notas = []`
`nome = []`

Listas na Linguagem Python

- Podemos declarar listas com elementos nela:

Sintaxe para declarar uma Lista com elementos em Python
<nome> **=** **[** *elemento1, elemento2, . . .***]**

Exemplos
primos = [1,2,3,5,7] notas = [1.0, 3.5, 7.0] nome = [“João”, “José”, “Maria”]

Listas na Linguagem Python

- Podemos acessar um elemento da lista

Exemplos

```
>>>primos = [1,2,3,5,7]
>>>notas = [1.0, 3.5, 7.0]
>>>nome = [“João”, “José”, “Maria”]
>>>
>>>primos[0]
1
>>>notas[2]
7.0
>>>nome[2]
José
>>>
```

Listas na Linguagem Python

- Podemos modificar um elemento da lista

Exemplos

```
>>>primos = [1,2,3,5,7]
>>>notas = [1.0, 3.5, 7.0]
>>>nome = [“João”, “José”, “Maria”]
>>>
>>>notas[0]=5.0
>>>notas[0]
5.0
>>>notas
[5.0, 3.5, 7.0]
>>>
```

Mostra toda a lista

Listas na Linguagem Python

- Podemos colocar um elemento da lista

Exemplos

```
>>>primos = [1,2,3,5,7]
>>>notas = [1.0, 3.5, 7.0]
>>>nome = [“João”, “José”, “Maria”]
>>>
>>>notas.append(11)
>>>notas
[1, 2, 3, 5, 7, 11]
>>>
```

Listas na Linguagem Python

- Ou remover um elemento da lista

Exemplos

```
>>>primos = [1,2,3,5,7]
>>>notas = [1.0, 3.5, 7.0]
>>>nome = [“João”, “José”, “Maria”]
>>>
>>>del nome[0]
>>>nome
[“José”, “Maria”]
>>>
```

Listas na Linguagem Python

Exemplo - Cálculo de Notas de ICC em Python

```
prova1 = []
prova2 = []
prova3 = []
Media = []

for i in range(0, 40):
    print("Informe as notas do aluno %d:", %i)
    prova1.append(float(input()))
    prova2.append(float(input()))
    prova3.append(float(input()))

for i in range(0, 40):
    media.append( ((prova1[i] + prova2[i] + prova3[i]) / 3) )

for i in range (0, 40):
    print("Media do aluno %d: %2.2f." % ( i+1, media[i]) )
```

DECLARO AS LISTAS VAZIAS!
PARA MANIPULAR MAIS À FRENTE”

UnB

Número Variável de Elementos

Exemplo - Cálculo de Notas de ICC

```
Algoritmo NotaICC_Variável

Variáveis
    prova1 : lista
    prova2 : lista
    prova3 : lista
    media : lista
    i, nro_alunos : inteiro

Início
    Escreva ("Informe o número de alunos:")
    Leia (nro_alunos)
    Para i ← 0 até nro_alunos - 1 faça
        Leia (prova1[i])
        Leia (prova2[i])
        Leia (prova3[i])
    Fim-para
    Para i ← 0 até nro_alunos - 1 faça
        media[i] ← ((prova1[i] + prova2[i] + prova3[i]) / 3)
    Fim-para
    Para i ← 0 até nro_alunos - 1 faça
        Escreva ("Média do Aluno ", i+1, ": ", media[i])
    Fim-para
Fim
```

UnB

Número Variável de Elementos

Exemplo - Cálculo de Notas de ICC

```
Algoritmo NotaICC_Variável

Variáveis
    prova1 : lista
    prova2 : lista
    prova3 : lista
    media : lista
    i, nro_alunos : inteiro

Início
    Escreva ("Informe o número de alunos:")
    Leia (nro_alunos)
    Para i ← 0 até nro_alunos - 1 faça
        Leia (prova1[i])
        Leia (prova2[i])
        Leia (prova3[i])
    Fim-para
    Para i ← 0 até nro_alunos - 1 faça
        media[i] ← ((prova1[i] + prova2[i] + prova3[i]) / 3)
    Fim-para
    Para i ← 0 até nro_alunos - 1 faça
        Escreva ("Média do Aluno ", i+1, ": ", media[i])
    Fim-para
Fim
```

UnB

Listas na Linguagem Python

Exemplo - Cálculo de Notas de ICC em Python

```
prova1 = []
prova2 = []
prova3 = []
Media = []

Print("Informe o número de alunos:")
nr_alunos = int(input())

for i in range(0, nr_alunos):
    print("Informe as notas do aluno %d:", %i)
    prova1.append(float(input()))
    prova2.append(float(input()))
    prova3.append(float(input()))

for i in range(0, nr_alunos):
    media.append( ((prova1[i] + prova2[i] + prova3[i]) / 3) )

for i in range (0, nr_alunos):
    print("Media do aluno %d: %2.2f." % ( i+1, media[i]) )
```

UnB

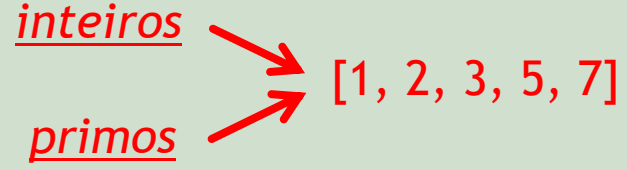
Listas – outras operações

- Uma lista em Python é um objeto. Por isso quando atribuímos uma lista a uma variável estamos apenas copiando a mesma referência do

Exemplo

```
>>>primos = [1,2,3,5,7]
>>>inteiros = primos
>>>primos
[1, 2, 3, 5, 7]
>>>inteiros
[1, 2, 3, 5, 7]
>>>inteiro[0]=10
>>>primos
[10, 2, 3, 5, 7]
>>>inteiros
[10, 2, 3, 5, 7]
>>>
```

Ou seja, *inteiros* virou apelido de *primos*! Quando modificamos *inteiros[0]*, estamos modificando *primos[0]*, uma vez que tanto primos como inteiros são referências para a mesma lista na memória



Listas – outras operações

- Mas então, como se cria uma outra cópia que seja independente da lista copiada?

Sintaxe para criar uma cópia independente de Lista em Python

```
<nome2> = <nome1> [:]
```

Exemplo

```
>>>primos = [1,2,3,5,7]
>>>inteiros = primos[:]
>>>inteiros[0]=10
>>>primos
[1, 2, 3, 5, 7]
>>>inteiros
[10, 2, 3, 5, 7]
>>>
```

Agora *inteiros* é uma nova cópia de *primos*

Listas – outras operações

- Obtendo o tamanho da lista

Exemplo

```
>>>primos = [1,2,3,5,7]
>>>len(primos)
5
>>>vazio = []
>>>len(vazio)
0
>>>
```

Listas de Listas

- Para representar uma matriz (vetor bidimensional) usamos lista de listas

Exemplo: matriz 3x3

```
>>>matriz =[[1,2,3],[4,5,6],[7,8,9]]
>>>print(matriz[0][0])
1
>>>print(matriz[1][1])
5
>>>print(matriz[2][2])
0
>>>matriz
[[1, 2, 3],[4, 5, 6],[7, 8, 9]]
>>>
```

Listas com elementos de tipos diferentes

- Assim como Python pode ter listas de listas, pode também ter listas de elementos com diferentes tipos.

Exemplo: Lista com tipos não homogêneos (estoque de uma loja)

```
item1 = [“Lápis”, 32, 1.50]
Item2 = [“Caneta”, 50, 2.30]
Item3 = [“Borracha”, 30, 2.00]
```

- Cada item da loja é uma lista com três elementos. O primeiro é do tipo string, o segundo é um inteiro e o terceiro é um float.

Listas com elementos de tipos diferentes

- O que me permite ter uma lista de listas chamada estoque.

Exemplo: Lista com tipos não homogêneos (estoque de uma loja)

```
>>>item1 = ["Lápis", 32, 1.50]
>>>item2 = ["Caneta", 50, 2.30]
>>>item3 = ["Borracha", 30, 2.00]
>>>
>>>estoque = [item1, item2, item3]
>>>
>>>for e in estoque:
>>>    print("Item: %s" % e[0])
>>>    print("Quantidade: %d" % e[1])
>>>    print("Preço: R$ %5.2f" % e[2])
Item: Lápis
Quantidade: 32
Preço: R$  1.50
Item: Caneta
Quantidade: 50
Preço: R$  2.30
Item: Borracha
Quantidade: 30
Preço: R$  2.00
>>>
```

Dicionários

- É uma estrutura de dados similar às listas, mas com propriedades de acesso diferentes.
- É composto por um conjunto de chaves e valores.
- O dicionário basicamente relaciona uma chave a um valor específico

Dicionários

- Por exemplo, podemos criar tabelas com dicionários:

Sintaxe para declarar um Dicionário em Python

```
<nome> = { chave1 : elemento1, . . .]
```

Exemplo

```
estoque = { "Lápis": 32,
            "Caneta": 50,
            "Borracha": 30}
```

Dicionários

- Mas a lista com o estoque da loja tinha a quantidade e o preço do item...
- Sem problema, o valor pode ser uma lista:

Exemplo

```
estoque = { "Lápis": [32, 1.50],
            "Caneta": [50, 2.30],
            "Borracha": [30, 2.00]}
```

Dicionários

- Dicionários usam suas chaves como índice e não índices numéricos que correspondem à sua posição em uma lista
- Ao atribuir valor a uma chave, duas coisas podem ocorrer:
  - Se a chave já existe:
    - O valor associado é atualizado para o novo valor
  - Senão
    - Nova chave será acrescentada ao dicionário com o seu respectivo valor

Dicionários

Exemplo: Dicionário (estoque de uma loja)

```
>>>estoque = {"Lápis": [32, 1.50],
>>>           "Caneta": [50, 2.30],
>>>           "Borracha": [30, 2.00]}
>>>
>>>print(estoque[“Caneta"])
[32, 1.50]
>>>estoque["Apontador"]=[6, 5.20]
>>>print(estoque)
{'Lápis': [32, 1.5], 'Caneta': [50, 2.3], 'Borracha': [30,
2.0], 'Apontador': [6, 5.2]}
>>>
```

Dicionários

- Ao acessar uma chave inexistente, ocorre um erro

Exemplo: Dicionário (estoque de uma loja)

```
>>>estoque = {"Lápis": [32, 1.50],
>>>           "Caneta": [50, 2.30],
>>>           "Borracha": [30, 2.00]}
>>>
>>>print(estoque[“Apontador"])
Traceback (most recent call last):
  File "<pyshell#26>", line 1, in <module>
    print(a["Apontador"])
KeyError: 'Apontador'
>>>
```

Dicionários

- Para ver se uma chave existe use o operador in

Exemplo: Dicionário (estoque de uma loja)

```
>>>estoque = {"Lápis": [32, 1.50],
>>>            "Caneta": [50, 2.30],
>>>            "Borracha": [30, 2.00]}
>>>
>>>print("Apontador" in estoque)
False
>>>print("Lápis" in estoque)
True
>>>
```

Dicionários

- Para deletar uma chave use a instrução del

Exemplo: Dicionário (estoque de uma loja)

```
>>>estoque = {"Lápis": [32, 1.50],
>>>           "Caneta": [50, 2.30],
>>>           "Borracha": [30, 2.00]}
>>>
>>>del estoque("Caneta")
>>>print(estoque)
{'Lápis': [32, 1.5], 'Borracha': [30, 2.0]}
>>>
```

Considerações finais

- Listas e Dicionários em Python são estruturas de dados muito poderosas
- Existem vários operdaores e instruções que podem ser usadas com estas estrtuturas
- Consulte os tutoriais de Python para aprender mais . . .

Listas – Exercícios

1. Escreva um algoritmo que leia duas listas de 25 inteiros cada (um vetor e depois o outro), crie uma nova lista que seja o resultado da intercalação dos elementos dos outros dois vetores. Ao final, mostre o conteúdo do novo vetor.

Listas – Exercícios

2. Escreva um algoritmo que leia duas listas bidimensionais 3x3 de inteiros cada. Crie uma nova lista 3x3 que seja o resultado da multiplicação das duas listas bidimensionais (matrizes 3x3). Ao final, mostre o conteúdo da nova lista.