Universidade Cruzeiro do Sul

Programação de Computadores

- LISTAS

Conceitos abordados nesta aula

- A proposta desta aula é apresentar as listas em Python e como podemos manipulá-las.

main.py

```
1  nomes = []
2  for i in range(5):
3      n = input("Digite um nome: ")
4      nomes.append(n)
5
6
7
8
9
10
11
12
13
```

Situação problema

Faça um programa em Python que leia os doze salários recebidos por um funcionário durante um ano, calcule e exiba na tela quanto ele receberá de 13º salário e 1/3 de férias. Para os cálculos, utilize as seguintes definições:

- O 13º salário deverá ser igual à média dos salários recebidos no ano.
- Para o cálculo de 1/3 de férias, faça a média dos salários * 1/3.

Obs.: **1-** Obrigatório utilizar alguma estrutura de repetição
2- Identificar o mês (Ex: Qual o salário recebido em Jan: R$)

Arrays: o que são?

Para entender o conceito de Listas, vamos começar com o conceito de arrays (vetores e matrizes).

- Arrays servem para armazenar várias informações utilizando apenas um identificador;
- Quando criamos uma variável, armazenamos apenas uma informação, ao passo que em um array podemos armazenar várias;
- Um array pode ter várias dimensões:

Variável

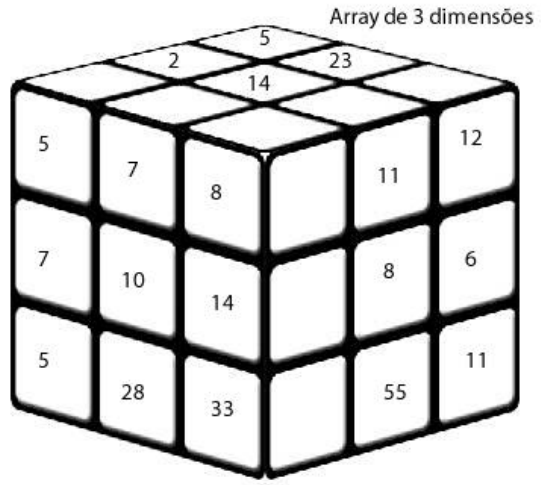
5

Array de 1 dimensão

5	7	8	10	14

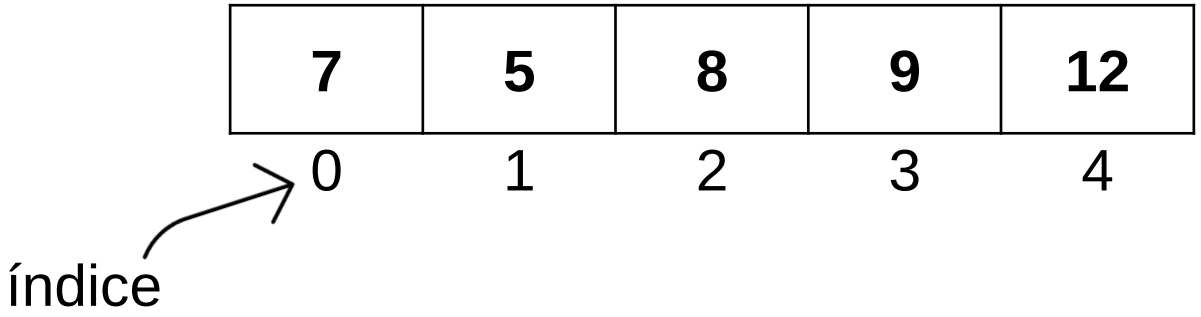
Array de 2 dimensões

5	7	8	10	14
7	11	23	2	5
5	33	28	12	14



Arrays: o que são?

- Arrays de uma dimensão também são chamados de vetores ou listas.
- Valores podem ser de tipos diferentes.
- **São dinâmico,** o que possibilita adicionarmos ou removermos valores uma vez que o array for criado.
- Acessamos cada posição usando o seu índice, que, nas linguagens de programação, **normalmente começa em 0 (zero).**



Listas: definição

- Coleção de valores referenciados por um identificador único.
- **Características:**
  - Acesso por meio de um índice inteiro.
  - Listas podem ser modificadas.
  - Pode-se incluir e remover itens de listas.

```
identificador = [dado1, dado2, ... , dadon]
notas = [8.0, 5.5, 9.3, 7.6, 3.1]
```

Listas em Python

- Uma lista em Python é uma estrutura que armazena vários dados, que **podem ser de um mesmo tipo ou não.**
- Listas são construções de linguagens de programação que servem para armazenar vários dados de forma simplificada.

```
lista1 = [10, 20, 30, 40]
lista2 = ["programação", "computadores", "python"]
lista3 = ["oi", 2.0, 2, 5, "exemplo"]
```

Listas em Python

- Suponha que desejamos guardar notas de vários alunos.
- Com o conceito somente de variáveis, como faríamos para armazenar as notas de, por exemplo, 100 alunos?

```
nota1   = float(input("Digite a nota do aluno 001: "))
nota2   = float(input("Digite a nota do aluno 002: "))
nota31 = float(input("Digite a nota do aluno 003: "))
...
nota100 = float(input("Digite a nota do aluno 100: "))
```

Certamente, criar 100 variáveis distintas **não** seria uma boa solução.

Criação de listas em Python

Python

```
nome = [ ]
ou
nome = list()
ou
teste = [“mamão",10,1.5]
```

Todas as criações produzem uma lista dinâmica

Fique ligado!

Observe que os dados armazenados nas listas não precisam ser de mesmo tipo.

Nosso pequeno problema...

Faça um programa em Python que leia os doze salários recebidos por um funcionário durante um ano, calcule e exiba na tela quanto ele receberá de 13º salário e 1/3 de férias. Para os cálculos, utilize as seguintes definições:

- O 13º salário deverá ser igual à média dos salários recebidos no ano.
- Para o cálculo de 1/3 de férias, faça a média dos salários * 1/3.

Obs.: **1-** Obrigatório utilizar alguma estrutura de repetição
2- Identificar o mês (Ex: Qual o salário recebido em Jan: R$)

```
mes = ["Jan","Fev","Mar","Abr","Mai","Jun","Jul","Ago","Set","Out","Nov","Dez"]
for indiceMes in range(12):
    salario = float(input("Informe o salário do mês %s R$ " %mes[indiceMes]))
```

Lista: atribuição de valores

- A utilização de uma lista está associada a uma estrutura de repetição.
- Com isso podemos facilmente percorrer uma lista para **consultas** ou atualizações.

```
nomes = ["Marcos","Eduardo","Mônica","Felipe"]
for indiceNome in range(4):
    print(nomes[ indiceNome ])
```


```
nomes = ["Marcos","Eduardo","Mônica","Felipe"]
for i in range(4):
    print( i )
```

Exemplos de aplicação

1- Faça um programa em Python que calcule a média de um aluno a partir de cinco notas previamente armazenadas em uma lista.

- Utilize: notas = [6, 7, 6.5, 4.8, 8]

```
notas = [6,7,6.5,4.8,8]
soma = 0
for i in range(5):
    soma = soma + notas[ i ]
media = soma / 5
print("Média: %.2f" %media)
```

```
notas = [6,7,6.5,4.8,8]
soma = 0
for i in notas:
    soma += i
media = soma / 5
print("Média: %.2f" %media)
```

Lista: atribuição de valores

- A utilização de uma lista está associada a uma estrutura de repetição.
- Com isso podemos facilmente percorrer uma lista para consultas ou **atualizações**.

```
listaNomes = [ ]
for  i  in range(5):
    nome = input("Digite um nome: ")
    listaNomes.append( nome )
```

Adicionando itens a uma lista!!

Fique ligado!

O comando **append()** adiciona um valor (n) ao final da lista

Lista: atribuição de valores

- A utilização de uma lista está associada a uma estrutura de repetição.
- Com isso podemos facilmente percorrer uma lista para consultas ou **atualizações**.

```
listaNomes = [ ]
for  i  in range(5):
    nome = input("Digite um nome: ")
    listaNomes.insert( 1, nome )
```

Adicionando itens a uma lista!!

Fique ligado !

O comando **insert()** adiciona um valor(n) à posição i da lista

Lista: atribuição de valores

```
Digite um nome: 1
1
Digite um nome: 2
1
2
Digite um nome: 3
1
3
2
```

```
Digite um nome: 4
1
4
3
2
Digite um nome: 5
1
5
4
3
2
```

Lista: como utilizar

- Pode-se acessar uma determinada posição da lista utilizando-se um índice de valor inteiro.
- A sintaxe para acesso de uma determinada posição é: **identificador[posição]**

```
notas = [8.0, 5.5, 9.3, 7.6, 3.1]
print(notas[1])
```

Saída→ **5.5**

A primeira posição da lista tem índice 0

Sendo *n* o tamanho da lista, os índices válidos para ela vão de 0 até **n-1**.

- A primeira posição da lista tem índice 0.
- A última posição da lista tem índice **n-1**.

Exemplos de aplicação

2- Vamos criar um programa em Python que solicite ao usuário o nomes de 5 pessoas e armazene em uma lista.

Em seguida o programa deve solicitar ao usuário um número de 0 a 4, correspondendo ao índice, e o programa deverá mostrar nome armazenado nesse índice.

```
Digite um nome: sss
Digite um nome: rrr
Digite um nome: fff
Digite um nome: ggg
Digite um nome: hhh
['sss', 'rrr', 'fff', 'ggg','hhh']
Digite um número: 4
hhh
```

Lista: outras funções

- A função **pop()** remove e retorna o item da posição

```
# Lista de animais
animais = ['gato', 'cão', 'cavalos', 'gato', 'tigre']


animais.pop(3)
>>> 'gato'


print(animais)
>>>['gato', 'cão', 'cavalos', 'tigre']


animais.pop()
>>> 'tigre'
```

Lista: outras funções

- A função **len()** retorna o tamanho da lista (nº de elementos):

```
nomes = [ "Maria", "José", "Joao" ]
print( len(nomes) )
```

Retorna a quantidade de elementos da lista

- É muito comum usar a função **len** junto com o laço for para percorrer todas as posições de uma lista:

```
notas = [ 8.5, 1.2, 9.0, 0.5, 3.1 ]
for i = range( len(notas) ):
        print( notas[ i ] )
```

Exemplos de aplicação

3- Faça um programa em Python que calcule e mostre a média de uma quantidade indeterminada de números inteiros digitados pelo usuário. Para sair o usuário deverá digitar 0. Use lista e exiba no final os números digitados.

Exemplos de aplicação

4- Faça um programa que leia n notas, mostre as notas e a média.

Exemplos de aplicação

4- Faça um programa que leia n notas, mostre as notas e a média.

Lista: outras funções

- A função **remove(x)** remove o primeiro item encontrado na lista cujo valor é igual a x.

```
nomes = [ "Maria", "José", "Joao" ]
nomes.remove("José")
print( nomes )
```

Lista: outras funções

- A função **enumerate** gera um tupla(*) em que o primeiro valor é o índice e o segundo é o elemento da lista sendo enumerada.

```
nomes = [ "Maria", "José", "Joao" ]
for x,e in enumerate(nomes):
    print( "[%d] - %s"%(x + 1, e) )
```

```
[1] - Maria
[2] - José
[3] - Joao
```

Exemplos de aplicação

5- Vamos fazer um programa em Python que controle a utilização de 5 salas do cinema CINEMARKO. O programa deverá ter as seguintes funcionalidades:

- Uma lista deverá armazenar os lugares vagos por sala: lugaresVagos = [10, 5, 6, 8, 0], respectivamente para as sala 1, 2, 3, 4 e 5.
- O usuário deverá digitar o número da sala e a quantidade de ingressos que deseja comprar, ou zero para encerrar o programa.
- O programa deverá verificar se a venda é possível antes de concretizá-la, informando quando não há lugares disponíveis para venda.
- Caso a compra seja efetivada, atualizar o número de lugares livres e exibir na tela.

Resumo dos métodos

Método	Parâmetros	Descrição
append	item	Acrescenta um novo item no final da lista
insert	posição, item	Insere um novo item na posição dada
pop	nenhum	Remove e retorna o último item
pop	posição	Remove e retorna o item da posição
sort	nenhum	Ordena a lista
reverse	nenhum	Ordena a lista em ordem reversa
index	item	Retorna a posição da primeira ocorrência do item
count	item	Retorna o número de ocorrências do item
remove	item	Remove a primeira ocorrência do item
enumerate	nenhum	Exibe o índice da lista sendo enumerada

Exemplos de aplicação

6- Faça um programa em Python que leia o nome e duas notas de n alunos e calcule a média. O usuário deverá digitar o número do aluno e o programa exibirá a média e o resultado, sabendo que o critério para aprovação é média igual ou maior que 6.0.

Exemplos de aplicação

7- Vamos criar um programa em Python que solicite ao usuário o nome de 5 pessoas, armazene em uma lista e exiba os nomes digitados e o tamanho da lista. Em seguida o programa deve solicitar ao usuário um nome, e o programa deverá remover o nome armazenado na lista, exibir os nomes digitados e o tamanho da lista.

Material Complementar

https://www.devmedia.com.br/como-trabalhar-com-listas-em-python/37460

http://devfuria.com.br/python/listas/

Alguma dúvida????

Créditos

Esta aula teve por base o material produzido e cedido gentilmente pelos **Professores Marco Antonio, Alcides, Lédon, Amilton e Cristiane**.

Grato à todos