UniAcademia

Disciplina

Estrutura de Dados

prof. Jacimar Tavares
jacimar.tavares@gmail.com

Módulo 02

Primeiros passos em estruturas de dados

Primeiros Passos

- Que tal começarmos relembrando algoritmos em Python?
  - Faça:

Elabore um algoritmo que dada a idade de um nadador classifica-o em uma das seguintes categorias:

infantil A = 5 - 7 anos

infantil B = 8-10 anos

juvenil A = 11-13 anos

juvenil B = 14-17 anos

adulto = maiores de 18 anos

Primeiros Passos

- Que tal começarmos relembrando algoritmos em Python?
  - Faça:

O cardápio de uma lancheria é o seguinte:

Especificação	Código	Preço
Cachorro quente	100	1,20
Bauru simples	101	1,30
Bauru com ovo	102	1,50
Hambúrger	103	1,20
Cheeseburguer	104	1,30
Refrigerante	105	1,00

Escrever um algoritmo que leia o código do item pedido, a quantidade e calcule o valor a ser pago por aquele lanche. Considere que a cada execução somente será calculado um item.

Primeiros Passos

- Que tal começarmos relembrando algoritmos em Python?
  - Faça:

> Faça um algoritmo que leia a idade de uma pessoa expressa em anos, meses e dias e mostre-a expressa apenas em dias.

Primeiras Implementações

- **Trabalhando com Arrays**
  - O array é uma coleção de objetos armazenados de forma contígua e cuja ordem de inserção e remoção não é definida previamente.
  - Considere o exemplo de estrutura ao lado. Vamos pensar na importância dos Arrays analisando-o.

```
if    (m == 1):
      print("Janeiro")
elif  (m == 2):
      print ("Fevereiro")
elif  (m == 3):
      print ("Março")
elif  (m == 4):
     print ("Abril")
elif  (m == 5):
     print ("Maio")
elif  (m == 6):
     print ("Junho")
elif  (m == 7):
     print ("Julho")
elif  (m == 8):
     print ("Agosto")
elif  (m == 9):
    print ("Setenbro")
elif  (m == 10) :
    print ("Outubro")
elif  (m == 11):
    print ("Novembro")
else (m == 12):
    print ("Dezembro")
```

Primeiras Implementações

- **Trabalhando com Arrays**
  - Link de documentação:
    https://docs.python.org/pt-br/3/library/array.html
  - Sintaxe básica:
    - sintaxe: <variável> = array(‘<tipo>, <valores>)

Primeiras Implementações

- **Trabalhando com Arrays**
  - Tabela informativa:

Type code	Tipo em C	Tipo em Python	Tamanho mínimo em bytes	Notas
`'b'`	signed char	int	1	
`'B'`	unsigned char	int	1	
`'u'`	Py_UNICODE	Caractere unicode	2	(1)
`'h'`	signed short	int	2	
`'H'`	unsigned short	int	2	
`'i'`	signed int	int	2	
`'I'`	unsigned int	int	2	
`'l'`	signed long	int	4	
`'L'`	unsigned long	int	4	
`'q'`	signed long long	int	8	
`'Q'`	unsigned long long	int	8	
`'f'`	float	float	4	
`'d'`	double	float	8	

Primeiras Implementações

- **Trabalhando com Arrays**
  - Exemplo

Exemplos de criação de vetores

```
from array import *      # carrega a biblioteca array e importa tudo
a=array('i',[10,20,30])  # cria um array de inteiros inicializado com os valores 10, 20 e 30
b=array('u','string')    # cria um vetor de characteres inicializado com "string"
```

Primeiras Implementações

- **Trabalhando com Arrays**
  - Uma forma mais elegante, está em definirmos um vetor de meses do tipo String, onde
    - os índices do vetor representam o mês na forma inteira.
      - Por exemplo:
      - no vetor months, para representarmos o mês de janeiro escreveríamos months[1], months[2] para representarmos o mês de fevereiro e assim sucessivamente.

```
if    (m == 1):
      print("Janeiro")
elif  (m == 2):
      print ("Fevereiro")
elif  (m == 3):
      print ("Março")
elif  (m == 4):
     print ("Abril")
elif  (m == 5):
     print ("Maio")
elif  (m == 6):
     print ("Junho")
elif  (m == 7):
     print ("Julho")
elif  (m == 8):
     print ("Agosto")
elif  (m == 9):
    print ("Setenbro")
elif  (m == 10) :
    print ("Outubro")
elif  (m == 11):
    print ("Novembro")
else (m == 12):
    print ("Dezembro")
```

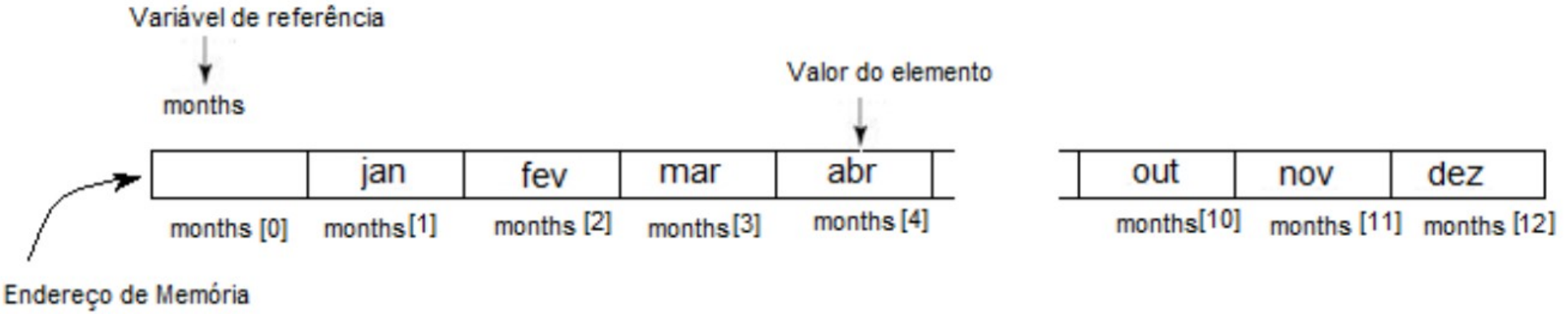
Primeiras Implementações

- **Trabalhando com Arrays**
  - Uma forma mais elegante, está em definirmos um vetor de meses do tipo String, onde
    - os índices do vetor representam o mês na forma inteira.
      - Por exemplo:
      - no vetor months, para representarmos o mês de janeiro escreveríamos months[1], months[2] para representarmos o mês de fevereiro e assim sucessivamente.

months = [“”, "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"]

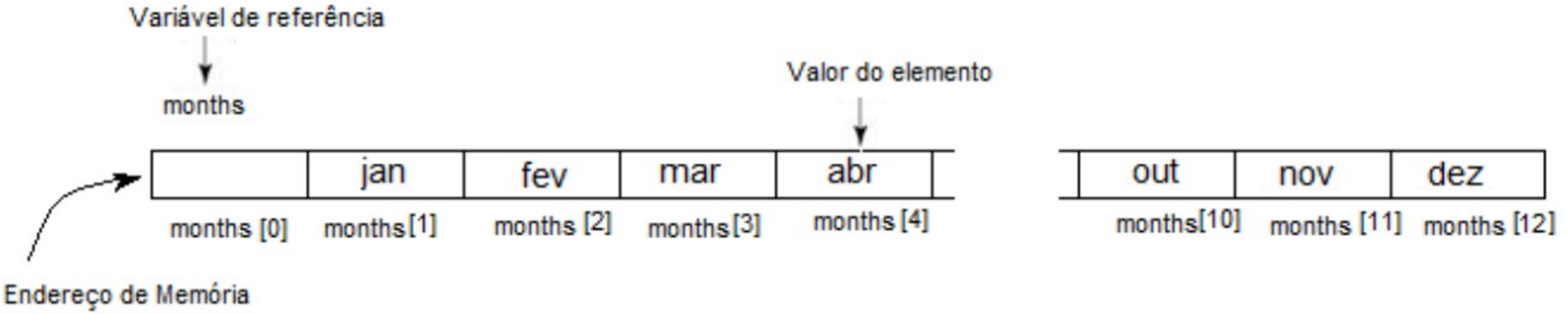
Primeiras Implementações

- Trabalhando com Arrays



Primeiras Implementações

- Trabalhando com Arrays



Primeiras Implementações

- **Trabalhando com Arrays**

```
números[0]
v = vogais[3]
```

Obtendo o índice de um determinado elemento em um vetor

```
alfabeto = ['a','e','i','o','u']
print(alfabeto.index('e'))
1
```

Primeiras Implementações

- Trabalhando com Arrays

```
len(números)
len(letras)
```

```
# retorna o tamanho do vetor números
# retorna o tamanho do vetor letras
```

Primeiras Implementações

- **Trabalhando com Arrays**
  - Percorrendo arrays

```
for s in suit_names:
    print(s)
```

Primeiras Implementações

- **Trabalhando com Arrays**
  - Arrays não tipados

```
a[0]=10
a[0]= 'Ana'
```

Atividades

- **Trabalhando com Arrays**
  - Atividade:
    - Defina um vetor de notas de um aluno. Carregue-o com dados e apresente o somatório das notas obtidas no semestre.

Primeiras Implementações

- **Trabalhando com Matriz**
  - Como em Python não se tem uma representação do tipo matriz, o que se faz é uma coleção de estruturas do tipo list() para fazer esta representação.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mn} \end{bmatrix}$$

Primeiras Implementações

- **Trabalhando com Matriz**
  - Como em Python não se tem uma representação do tipo matriz, o que se faz é uma coleção de estruturas do tipo list() para fazer esta representação.

```
# inicialização de uma matriz 3 x 2 de números inteiros
A = [ [1,2],[3,4], [5,6]]
# inicialização de uma matriz 5 x 1 do tipo char
A = [ ['a], ['e'], ['i'], ['o'], ['u']]
```

Primeiras Implementações

- **Trabalhando com Matriz**
  - Quando criamos uma matriz por exemplo (4x2) do tipo inteiro, podemos escrever inicializados com zero

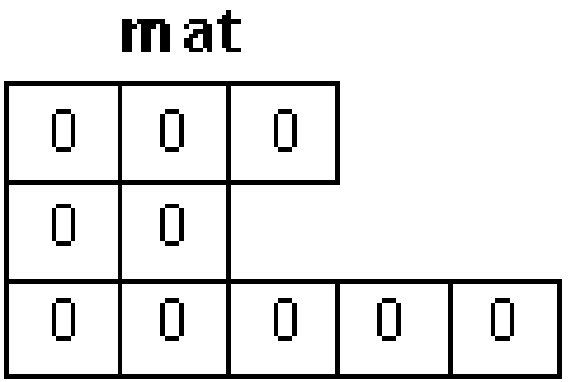
```
>>> # matriz de 4 linhas e 2 colunas com elementos nulos
>>> a = [ [0] * 2] * 4
>>> a
[[0, 0], [0, 0], [0, 0], [0, 0]]
```

```
def display(mat):
    s = ''
    for i in range(len(mat)):
        for j in range(len(A[i])):
            s += str(mat[i][j]) + ', '
        s +='\n'
    return s
```

Primeiras Implementações

- **Trabalhando com Matriz**
  - Com quantidade de colunas diferentes

```
mat = [[0] * 3, [0] * 2, [0] * 5 ]
#  alocamos 3 colunas para a linha 0
#  alocamos 2 colunas para a linha 1
#  alocamos 5 colunas para a linha 2
```



Atividades

- **Trabalhando com Matrizes**
  - Atividade:
    - Crie uma matriz com as notas obtidas no semestre. Ao final, imprima o somatório de todos os valores. Matriz deve ser 3,3.