Algoritmos e Lógica de Programação

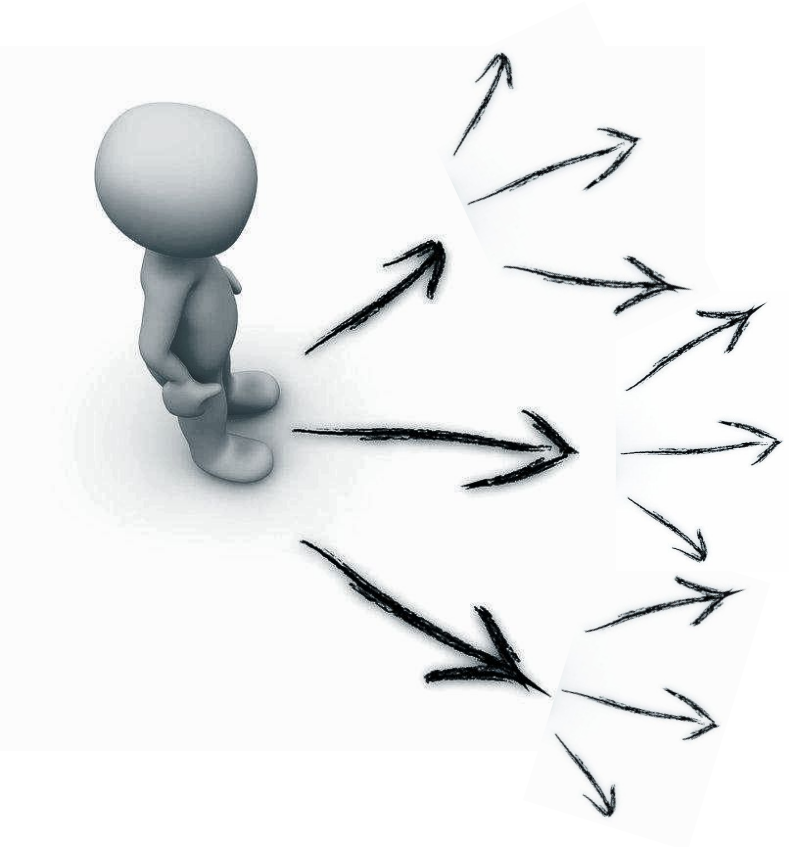
Prof. Flavius Gorgônio

Semana 5

Encadeamento de Estruturas de Controle de Decisão

Agenda

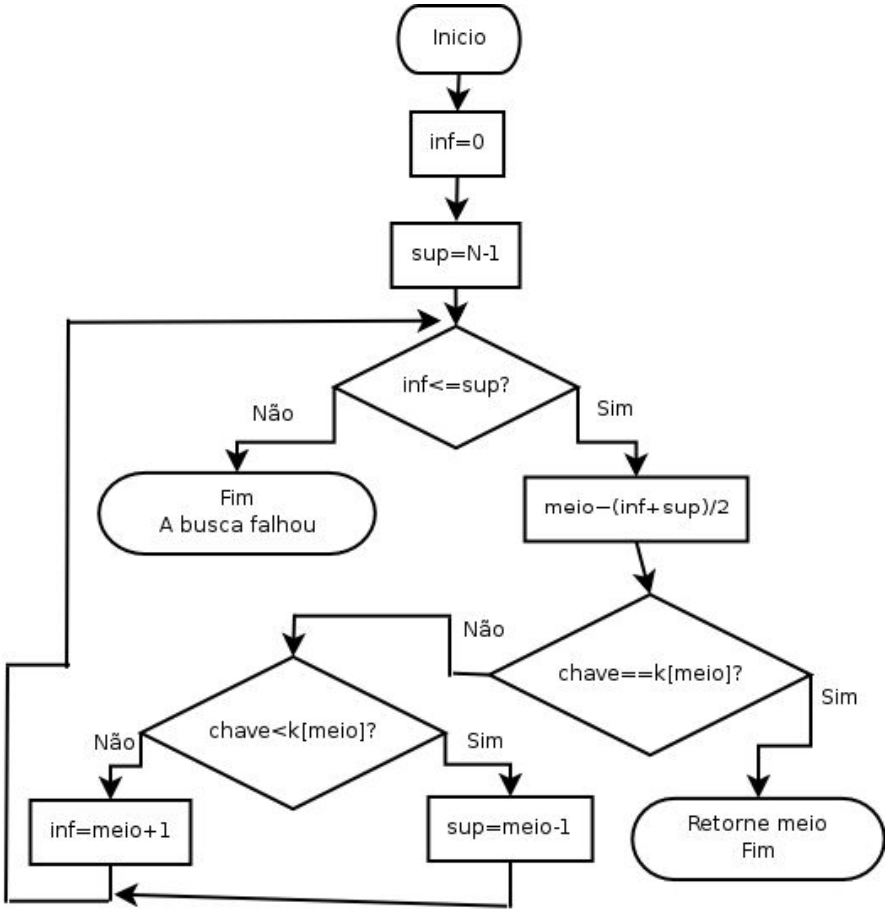
- Motivação
- Estruturas de decisão aninhadas
- Estruturas de seleção múltipla
- Utilização combinada de estruturas de decisão
- Aplicações e exemplos



Motivação

Alguns processos decisórios possuem fluxos computacionais mais complexos

- Fluxos computacionais encadeados
- Necessidade de estruturas de decisão aninhadas
- Utilização combinada de estruturas de decisão



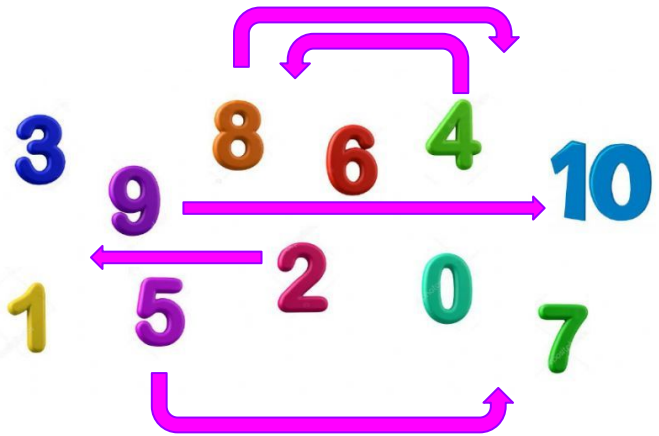
Problemas simples de ordenação de valores

Quem é o maior e o menor?

Escreva um programa em Python que leia três valores inteiros e determine o maior e o menor entre eles.

Exibindo valores em ordem crescente

Escreva um programa em Python que leia três valores inteiros e escreva-os em ordem crescente.



Valores em ordem crescente (versão 1)

```
print('Informe os valores a serem ordenados')
a = int(input('Valor 1: '))
b = int(input('Valor 2: '))
c = int(input('Valor 3: '))
if (a <= b) and (b <= c):
    print('Ordem crescente: %d < %d < %d'%(a,b,c))
if (a <= c) and (c <= b):
    print('Ordem crescente: %d < %d < %d'%(a,c,b))
if (b <= a) and (a <= c):
    print('Ordem crescente: %d < %d < %d'%(b,a,c))
if (b <= c) and (c <= a):
    print('Ordem crescente: %d < %d < %d'%(b,c,a))
if (c <= a) and (a <= b):
    print('Ordem crescente: %d < %d < %d'%(c,a,b))
if (c <= b) and (b <= a):
    print('Ordem crescente: %d < %d < %d'%(c,b,a))
```

Valores em ordem crescente (versão 2)

```
print('Informe os valores a serem ordenados')
a = int(input('Valor 1: '))
b = int(input('Valor 2: '))
c = int(input('Valor 3: '))
if (a <= b) and (b <= c):
    print('Ordem crescente: %d < %d < %d'%(a,b,c))
elif (a <= c) and (c <= b):
    print('Ordem crescente: %d < %d < %d'%(a,c,b))
elif (b <= a) and (a <= c):
    print('Ordem crescente: %d < %d < %d'%(b,a,c))
elif (b <= c) and (c <= a):
    print('Ordem crescente: %d < %d < %d'%(b,c,a))
elif (c <= a) and (a <= b):
    print('Ordem crescente: %d < %d < %d'%(c,a,b))
elif (c <= b) and (b <= a):
    print('Ordem crescente: %d < %d < %d'%(c,b,a))
```

Valores em ordem crescente (versão 3)

```
print('Informe os valores a serem ordenados')
a = int(input('Valor 1: '))
b = int(input('Valor 2: '))
c = int(input('Valor 3: '))
if (a <= b) and (b <= c):
    print('Ordem crescente: %d < %d < %d'%(a,b,c))
elif (a <= c) and (c <= b):
    print('Ordem crescente: %d < %d < %d'%(a,c,b))
elif (b <= a) and (a <= c):
    print('Ordem crescente: %d < %d < %d'%(b,a,c))
elif (b <= c) and (c <= a):
    print('Ordem crescente: %d < %d < %d'%(b,c,a))
elif (c <= a) and (a <= b):
    print('Ordem crescente: %d < %d < %d'%(c,a,b))
else:
    print('Ordem crescente: %d < %d < %d'%(c,b,a))
```

Valores em ordem crescente (versão 4)

```
print('Informe os valores a serem ordenados')
a = int(input('Valor 1: '))
b = int(input('Valor 2: '))
c = int(input('Valor 3: '))
if (a <= b):       # a b c; a c b; c a b
    if (b <= c):    # a b c
          print('Ordem crescente: %d < %d < %d'%(a,b,c))
    elif (a <= c):    # a c b
          print('Ordem crescente: %d < %d < %d'%(a,c,b))
    else:          # c a b
          print('Ordem crescente: %d < %d < %d'%(c,a,b))
else:                    # b a c; b c a; c b a
    if (a <= c):    # b a c
          print('Ordem crescente: %d < %d < %d'%(b,a,c))
    elif (b <= c):    # b c a
          print('Ordem crescente: %d < %d < %d'%(b,c,a))
    else:          # c b a
          print('Ordem crescente: %d < %d < %d'%(c,b,a))
```

Vamos falar de parábolas...

Lista 5 - Questão 8

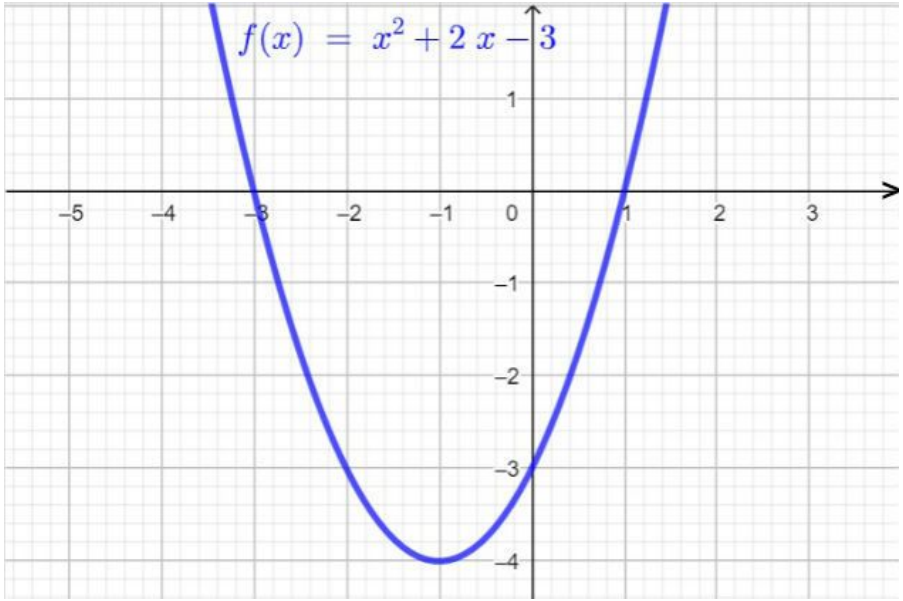
Escreva um programa em Python que leia os coeficientes de uma equação do 2º. grau e calcule as suas raízes reais, se for possível.

O programa deve exibir uma mensagem informando quantas raízes reais a equação possui e as raízes calculadas.


Bhaskara Akari (1114-1185)

$$\Delta = b^2 - 4.a.c$$

$$x = \frac{-b \pm \sqrt{\Delta}}{2.a}$$

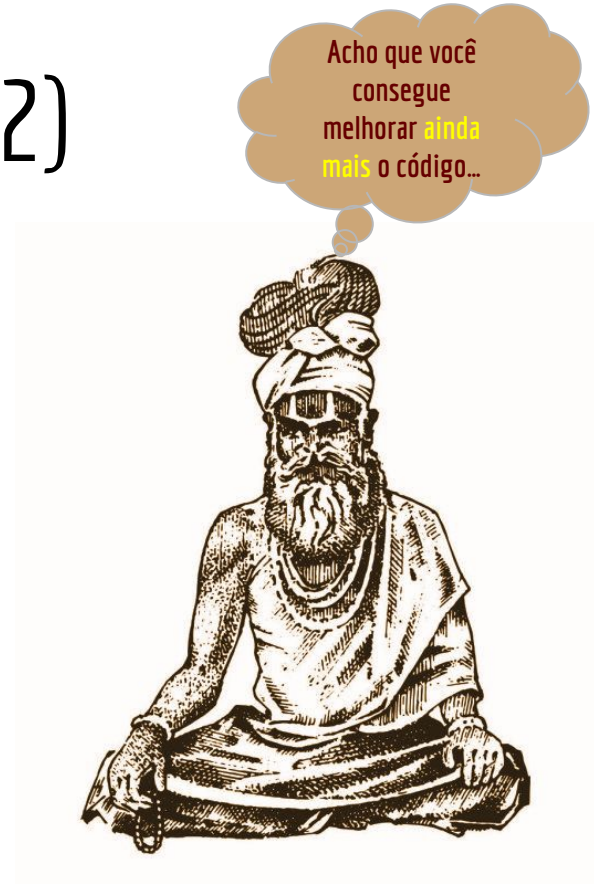


Raízes de equação 2º grau (versão 1)

```
from math import sqrt
print('Informe os coeficientes a, b e c de uma equação do 2o. grau')
a = int(input('a: '))
b = int(input('b: '))
c = int(input('c: '))
delta = b**2 - (4 * a * c)
x1 = (-b + sqrt(delta)) / (2 * a)
x2 = (-b - sqrt(delta)) / (2 * a)
print('As raízes da equação %dx² + %dx + %d = 0 são:'%(a,b,c))
print('x1 = %.1f'%x1)
print('x2 = %.1f'%x2)
```

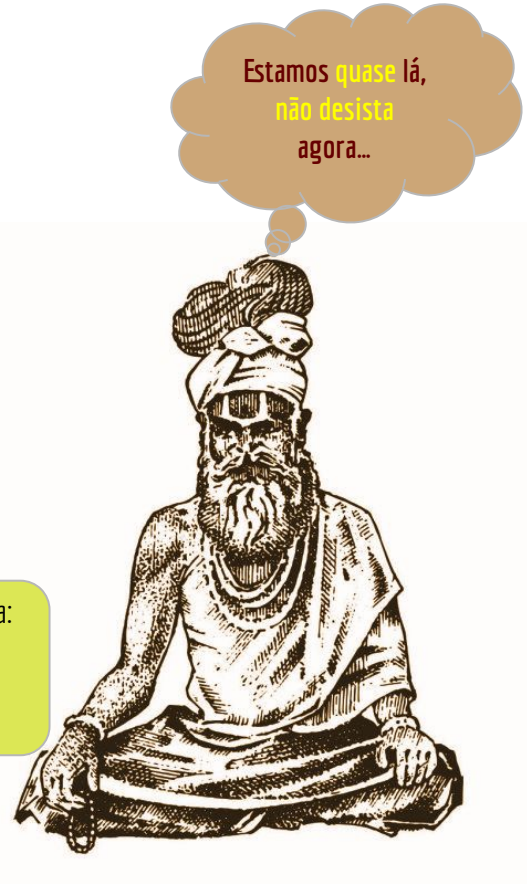
Raízes de equação 2º grau (versão 2)

```
from math import sqrt
print('Informe os coeficientes a, b e c de uma equação do 2o. grau')
a = int(input('a: '))
b = int(input('b: '))
c = int(input('c: '))
delta = b**2 - (4 * a * c)
if delta < 0:
    print('Valor de delta: %d'%delta)
    print('Dessa forma, não existem raízes reais que satisfaçam a equação')
else:
    x1 = (-b + sqrt(delta)) / (2 * a)
    x2 = (-b - sqrt(delta)) / (2 * a)
    print('As raízes da equação %dx² + %dx + %d = 0 são:'%(a,b,c))
    print('x1 = %.1f'%x1)
    print('x2 = %.1f'%x2)
print('Fim do programa')
```



Raízes de equação 2º grau (versão 3)

```
from math import sqrt
print('Informe os coeficientes a, b e c de uma equação do 2o. grau')
a = int(input('a: '))
b = int(input('b: '))
c = int(input('c: '))
delta = b**2 - (4 * a * c)
if delta < 0:
    print('Valor de delta: %d'%delta)
    print('Dessa forma, não existem raízes reais que satisfaçam a equação')
elif delta == 0:
    x = (-b) / (2 * a)
    print('A raiz da equação %dx² + %dx + %d = 0 é:'%(a,b,c))
    print('x = %.1f'%x)
else:
    x1 = (-b + sqrt(delta)) / (2 * a)
    x2 = (-b - sqrt(delta)) / (2 * a)
    print('As raízes da equação %dx² + %dx + %d = 0 são:'%(a,b,c))
    print('x1 = %.1f'%x1)
    print('x2 = %.1f'%x2)
print('Fim do programa')
```



Raízes de equação 2º grau (versão 4)

E agora ???

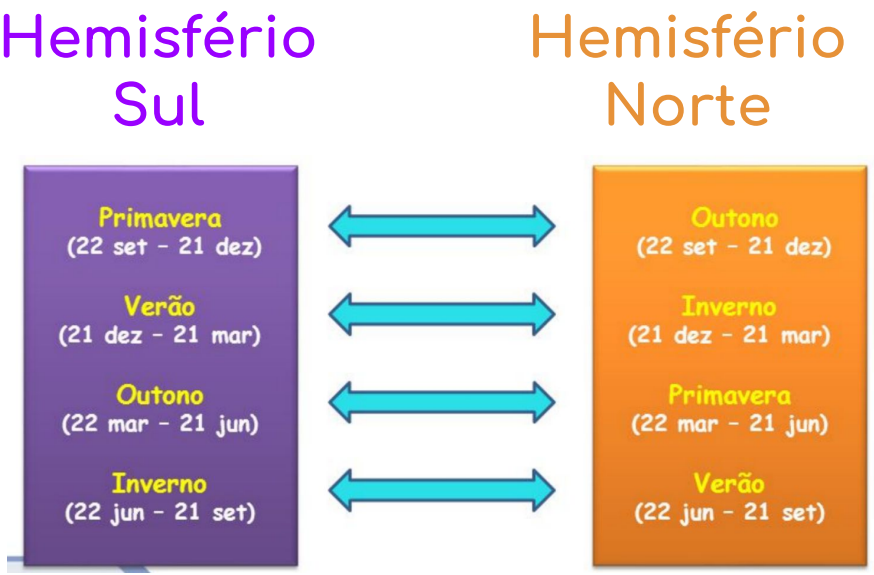
Mudaram as estações, nada mudou... ♪🎵♪🎵♩♩

Lista 5 - Questão 10

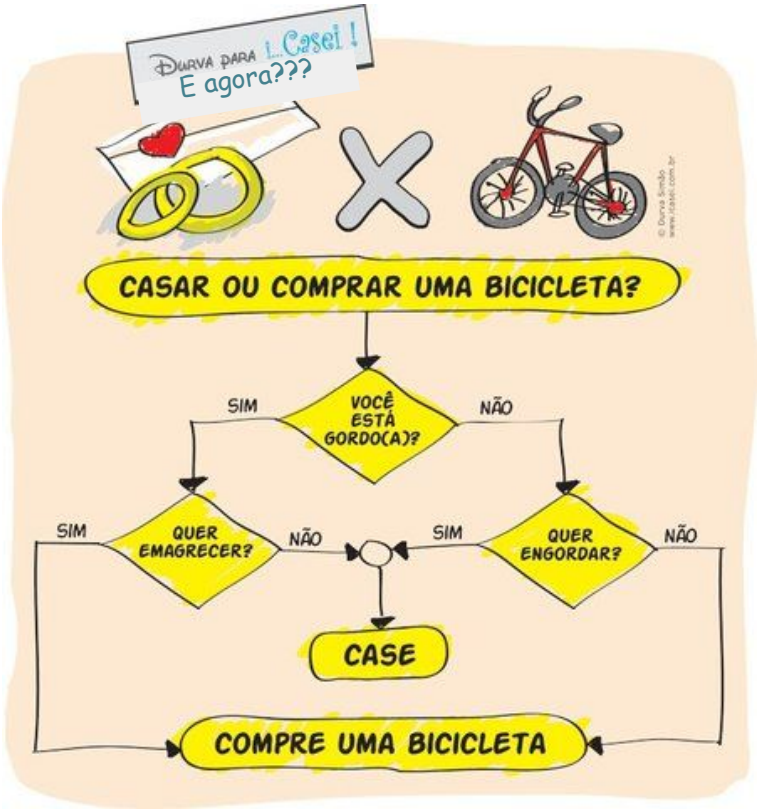
Escreva um programa em Python que leia dia e mês de uma determinada data e a localização (hemisfério) do usuário e determine qual a estação do ano correspondente à data.

Essas datas são fixas?

O que ocorre nos anos bissextos?



Casar ou comprar uma bicicleta?



Escreva um programa em linguagem Python que simule o fluxograma ao lado, auxiliando um(a) noivo(a) indeciso(a) a tomar a decisão correta em relação ao seu futuro matrimônio

Utilize variáveis para armazenar respostas para as perguntas que serão feitas ao candidato a nubente

Utilize estruturas de decisão para simular o fluxo decisório

Jogando dados contra o computador

Podemos simular dados randômicos com o gerador de números (pseudo) aleatórios do Python

A biblioteca random possui uma função chamada randint(a, b) que gera números inteiros (pseudo) aleatórios dentro de um intervalo [a, b]

E aí?

Vamos fazer um joguinho?

```
import random

jog = random.randint(1,6)
comp = random.randint(1,6)
print("Jogador   : ", jog)
print("Computador: ", comp)
if jog > comp:
  print("Jogador Ganhou!")
elif comp > jog:
  print("Computador Ganhou!")
else:
  print("Empate!")
print("Fim do Jogo!")
```

Dois novos joguinhos com dados

Lançando os Dados

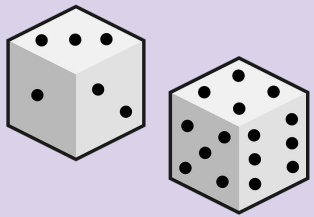
Escreva um programa em Python que simule uma disputa de dados entre o usuário e o computador. Cada jogador deve lançar dois dados e os pontos dos dados devem ser somados

O programa deve gerar números pseudo-aleatórios para representar os dados do jogador e do computador, exibindo os valores obtidos e identificando quem ganhou a partida. Utilize uma representação gráfica para apresentar o resultado

Sete ou Onze

Escreva um programa em Python que simule um jogo onde dois dados devem ser lançados simultaneamente

O jogador vence se a soma dos pontos dos dois dados for 7 ou 11, caso contrário vence o computador.



Outras ideias de jogos com dados

Kiriri: Jogo de dados para crianças grandes

- O computador lança dois dados e informa aos 2 jogadores qual o valor da soma dos pontos dos dados
- Cada jogador tentará adivinhar qual a combinação dos dados que deu origem aquela soma (em qualquer ordem)
- Os palpites devem ser alternados em cada rodada, um ou outro inicia o jogo
- O segundo jogador deve, necessariamente, sugerir um palpite diferente do primeiro jogador
- O jogador que acertar o palpite, vence o jogo
- Se ninguém acertar, o computador vence

Pedra, papel e tesoura

Escreva um programa em Python que simule o jogo PEDRA, PAPEL e TESOURA, a ser disputado entre um jogador humano e o computador

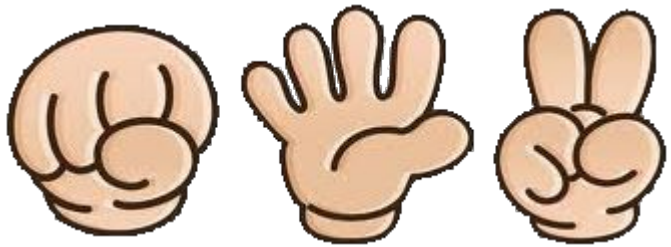
O jogador humano deverá escolher entre uma das três opções e a escolha do computador deverá ser feita de forma aleatória

O programa deverá realizar o julgamento e definir quem venceu o jogo, lembrando que PEDRA vence TESOURA, TESOURA vence PAPEL e PAPEL vence PEDRA. Considere a possibilidade de haver empate

JO

KEN

PÔ



Desafio da Semana: Validando datas

Escreva um programa em Python que leia uma data, composta por dia, mês e ano (cada valor informado separadamente) e verifique se a entrada corresponde a uma data válida

Verifique se o valor informado para o ano é maior ou igual a zero, se o valor informado para o mês está compreendido entre 1 e 12 e se o dia existe naquele mês

Considere, ainda, se o ano é ou não bissexto, lembrando que para um ano ser considerado bissexto, ele deve ser divisível por 4 e, ao mesmo tempo, não ser divisível por 100, a menos que seja divisível por 400

Sugestões de bibliografia

Básica

- MENEZES, Nilo Ney Coutinho. Introdução à programação com Python: algoritmos e lógica de programação para iniciantes. 2a ed. São Paulo: Novatec, 2010. 328p. ISBN: 9788575224083
- FARRELL, Joyce. Lógica e design de programação: introdução. São Paulo: Cengage Learning, 2010. xiv, 416p. ISBN: 9788522107575.
- SOUZA, Marco; GOMES, Marcelo; SOARES, Márcio; CONCILIO, Ricardo. Algoritmos e Lógica de Programação. 2a ed. São Paulo: Cengage Learning, 2013. 240 p. ISBN: 9788522111299

Complementar

- ASCENCIO, Ana Fernanda; CAMPOS, Edilene Aparecida. Fundamentos da programação de computadores: Algoritmos, Pascal, C/C++(Padrão Ansi) e Java. 3a ed. São Paulo: Pearson Prentice Hall. 569 p. ISBN: 9788564574168
- ZELLE, John M. Python programming: an introduction to computer science. 2nd ed. Sherwood, Or.: Franklin, Beedle & Associates, c2010. xiv, 514 p. ISBN: 9781590282410.
- LUTZ, Mark; ASCHER, David. Learning Python. 4nd ed. Sebastopol, CA: O Reilly, c2009. xliv, 1162 p. ISBN: 978059615064.

Copyright