



# Algoritmos e Programação de Computadores

## Execução Repetitiva: Uso de Variáveis Indicadora e Contadora

Ref.: material original (1o S., T. KLMN). por **Profa. Sandra Avila**, Instituto de Computação (IC/  
Unicamp)  
MC102-Z, 28 Agosto, 2018

# Agenda

---

- Uso de *loops*/laços para resolver problemas.
  - Variável indicadora
  - Variável contadora
- Uso de tais variáveis para obter maior eficiência na codificação
- Sugestão: uso de ferramenta de auxílio para análise de código (*debug*):
  - Exemplo: IDE (*Integrated Development Environment*), Thonny (Univ. of Tartu, Estonia)

# Comandos para execução repetitiva

- Instruções e/ou comandos para exec. repetitiva em Python.
- alguns exemplos de sua utilização (*loops/laços*).

```
while condicao:  
    comando(s)
```

```
for variável in lista:  
    comando(s)
```

Variável Indicadora

# Variável Indicadora

- Um uso comum de laços (*loops*) é a verificação se um determinado objeto, ou conjunto de objetos, satisfaz uma propriedade ou não (i.e., para que uma condição particular seja satisfeita).
- Um *padrão* que pode ser útil na resolução deste tipo de problema é o uso de uma **variável indicadora**.
- *padrão*: uso de uma construção específica na linguagem Python
  - Ex. > atribuição de um valor de verdade a uma variável de tipo *bool*

# Variável Indicadora

- Um padrão a ser utilizado na resolução de problemas cuja característica é a execução repetitiva, via uma **variável indicadora**.
  - Assumimos que o objeto satisfaz a propriedade (**indicadora = True**)
- padrão: uso de uma construção específica na linguagem Python
  - Ex1.. Se Z é uma variável do tipo *bool*, atribui-se `Z= True`
  - Ex.2. `<expression> Z=A or B` (sendo A e B vars. de tipo bool);

# Variável Indicadora

- Exemplo de técnica utilizada na resolução de problemas via o uso de uma **variável indicadora** :
  - Assumimos que o objeto satisfaz a propriedade (**indicadora = True**).
  - Por meio de um laço (*loop*), verifica-se se o objeto realmente satisfaz a dita propriedade.
  - Se em alguma iteração descobrirmos que o objeto não satisfaz a propriedade, então modifica-se o valor da var. indicadora, ex. **indicadora = False**.

# Variável Indicadora

- Uso da **variável indicadora** fora e dentro de um *loop*/laço
  - Inicialmente assume-se que o objeto satisfaz uma certa propriedade, ex. (**indicadora = True**).
  - Por meio de um *loop*/laço verifica-se se o objeto satisfaz a propriedade.
- Exemplo de uso em trecho de programa (pseudo-código)

Inicialização: Atribuição da var. Indicadora (ex. **Z=True**)

Loop: while <expression>

    If <condição satisfeita>

        Modificar variavel indicadora (ex. **Z=False**)



# Exemplo: Número Primo

- Problema: Determinar se um número  $n$  é primo ou não.

# Exemplo: Número Primo

- Problema: Determinar se um número  $n$  é primo ou não.
- Um número é primo se seus únicos divisores são 1 e ele mesmo.
- Dado um número  $n$  como detectar se este é ou não primo?
  - Leia o número  $n$ .
  - Teste se nenhum dos números entre 2 e  $(n - 1)$  satisfaz a condição de ser **divisor** de  $n$ .

# Exemplo: Número Primo

- Problema: Determinar se um número  $n$  é primo ou não.
- Um número é primo se seus únicos divisores são 1 e ele mesmo.
- Dado um número  $n$  como detectar se este é ou não primo?
  - Leia o número  $n$ .
  - Teste se nenhum dos números entre 2 e  $(n - 1)$  divide  $n$ .
- Lembre-se que o operador % retorna o resto da divisão.
- Portanto  $(a \% b)$  é zero se e somente se  $b$  divide  $a$ .

# Exemplo: Número Primo

- Dado um número  $n$  como detectar se este é ou não primo?
  - Leia o número  $n$ .
  - Teste se nenhum dos números entre 2 e  $(n - 1)$  divide  $n$ .

# Exemplo: Número Primo

- Dado um número  $n$  como detectar se este é ou não primo?
  - Leia o número  $n$ .
  - Faça a variável **indicadora = True**, assumindo que é primo.
  - Teste se nenhum dos números entre 2 e  $(n - 1)$  divide  $n$ .

# Exemplo: Número Primo

- Dado um número  $n$  como detectar se este é ou não primo?
  - Leia o número  $n$ .
  - Faça a variável **indicadora = True**, assumindo que é primo.
  - Teste se nenhum dos números entre 2 e  $(n - 1)$  divide  $n$ .
  - Se o resto da divisão for igual a zero então faça **indicadora = False**. Com isto descobrimos que não é primo.

# Exemplo: Número Primo


```
n = int(input("Digite um número inteiro positivo: "))

numero = 2
primo = True # primo é a variável indicadora

while (numero <= n-1) and (primo):
    if (n % numero == 0): # se n é divisível por numero
        primo = False
    numero = numero + 1

if (primo):
    print("É primo.")
else:
    print("Não é primo.")
```

# Exemplo: Número Primo



```
n = int(input("Digite um número inteiro positivo: "))

numero = 2
primo = True # primo é a variável indicadora

while (numero <= n-1) and (primo):
    if (n % numero == 0): # se n é divisível por numero
        primo = False
    numero = numero + 1

if (primo):
    print("É primo.")
else:
    print("Não é primo.")
```



# Exemplo: Número Primo (com break)

```
n = int(input("Digite um número inteiro positivo: "))

numero = 2
primo = True # primo é a variável indicadora

while (numero <= n-1):
    if (n % numero == 0): # se n é divisível por numero
        primo = False
        break
    numero = numero + 1

if (primo):
    print("É primo.")
else:
    print("Não é primo.")
```

# Exemplo: Números em Ordem Crescente

- Problema: Fazer um programa que lê  $n$  números inteiros do teclado, e no final informa se os números lidos estão ou não em ordem crescente.
- Usaremos uma variável indicadora na resolução deste problema.

# Exemplo: Números em Ordem Crescente

- Um laço (*loop*) principal será responsável pela leitura dos números.
- Vamos usar duas variáveis, uma que guarda o número lido na iteração atual, e uma que guarda o número lido na iteração anterior.
- Os números estarão ordenados se a condição (*anterior*  $\leq$  *atual*) for válida (*Verdadeira*) durante a leitura de todos os números.

```
n = int(input("Digite um número: "))
anterior = int(input())

i = 1 # leu um número
ordenado = True # ordenado é a variável indicadora

while (i < n) and (ordenado):
    atual = int(input())
    i = i + 1 # leu mais um número
    if (atual < anterior):
        ordenado = False
        anterior = atual

if (ordenado):
    print("Sequência está ordenada.")
else:
    print("Sequência não está ordenada.")
```

# Variável Contadora

# Variável Contadora

- Considere ainda o uso de laços para a verificação se um determinado objeto, ou conjunto de objetos, satisfaz uma propriedade ou não.
- Um outro padrão (recurso de programação, expressão Python) que pode ser útil é o uso de uma **variável contadora**.

# Variável Contadora

- Considere ainda o uso de laços (*loops*) para a verificar se um determinado objeto, ou conjunto de objetos, satisfaz uma propriedade (ou condição) ou não.
- Um outro padrão que pode ser útil é o uso de uma **variável contadora**.
  - Esperamos que um objeto satisfaça  $x$  vezes uma sub-propriedade. Usamos um laço (*loop*) e uma variável que **conta** o número de vezes que o objeto tem a sub-propriedade satisfeita.

# Variável Contadora

- Considere ainda o uso de laços para a verificação se um determinado objeto, ou conjunto de objetos, satisfaz uma propriedade ou não.
- Um outro padrão que pode ser útil é o uso de uma **variável contadora**.
  - Esperamos que um objeto satisfaça  $x$  vezes uma sub-propriedade. Usamos um laço e uma variável que **conta** o número de vezes que o objeto tem a sub-propriedade satisfeita.
  - Ao terminar o laço (*loop*), se a variável contadora for igual a  $x$  então o objeto satisfaz a propriedade.



# Exemplo: Número Primo

- Problema: Determinar se um número  $n$  é primo ou não.
- Um número  $n$  é primo se nenhum número de 2 até  $(n - 1)$  dividi-lo.
  - Podemos usar uma variável que **conta** quantos números são divisores de  $n$ .
  - Se o número de divisores for 0, então  $n$  é primo.

# Exemplo: Número Primo

```
n = int(input("Digite um número inteiro positivo: "))

numero = 2
divisores = 0 # divisores é a variável contadora

while (numero <= n-1):
    if (n % numero == 0): # se n é divisível por numero
        divisores = divisores + 1
    numero = numero + 1

if (divisores == 0):
    print("É primo.")
else:
    print("Não é primo.")
```


# Exemplo: Número Primo

```
n = int(input("Digite um número inteiro positivo: "))

numero = 2
divisores = 0 # divisores é a variável contadora

while (numero <= n-1) and (divisores == 0):
    if (n % numero == 0): # se n é divisível por numero
        divisores = divisores + 1
        numero = numero + 1

if (divisores == 0):
    print("É primo.")
else:
    print("Não é primo.")
```



É melhor terminar o laço assim que descobrirmos algum divisor de  $n$ .

# Exemplo: Número Primo

```
n = int(input("Digite um número inteiro positivo: "))

numero = 2
divisores = 0 # divisores é a variável contadora

while (numero <= n-1) and (divisores == 0):
    if (n % numero == 0): # se n é divisível por numero
        divisores = divisores + 1
    numero = numero + 1

if (divisores == 0):
    print("É primo.")
else:
    print("Não é primo.")
```



Basta testarmos até  $n/2$ . Por que?

# Exemplo: Números em Ordem Crescente

- Problema: Fazer um programa que lê  $n$  números inteiros do teclado, e no final informa se os números lidos estão ou não em ordem crescente.
- Vamos refazer o programa com uma variável contadora.

```
n = int(input("Digite um número: "))
anterior = int(input())

i = 1 # leu um número
ordenado = True # ordenado é a variável indicadora

while (i < n) and (ordenado):
    atual = int(input())
    i = i + 1 # leu mais um número
    if (atual < anterior):
        ordenado = False
    anterior = atual

if (ordenado):
    print("Sequência está ordenada.")
else:
    print("Sequência não está ordenada.")
```

```
n = int(input("Digite um número: "))
anterior = int(input())

i = 1 # leu um número
ordenado = 0 # ordenado é a variável contadora

while (i < n) and (ordenado == 0):
    atual = int(input())
    i = i + 1 # leu mais um número
    if (atual < anterior):
        ordenado = ordenado + 1
    anterior = atual

if (ordenado == 0):
    print("Sequência está ordenada.")
else:
    print("Sequência não está ordenada.")
```

# Resumo

- O uso de variáveis **acumuladora**, **indicadora** e **contadora** são úteis em várias situações.
- Mas não existem fórmulas para a criação de soluções para problemas.
- Em outros problemas, o uso destes padrões pode aparecer em conjunto, ou nem mesmo aparecer como parte da solução.



# Exercício: Número Adjacente

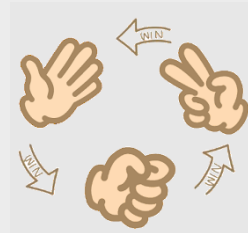
- Fazer um programa que lê  $n$  números inteiros do teclado, e no final informa se os números lidos tem dois dígitos adjacentes iguais.

Exemplos:

Para  $n = 5$  números inteiros e 21212, a resposta é não.

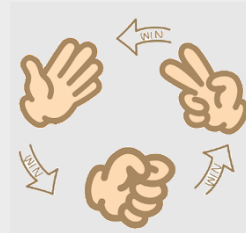
Para  $n = 5$  números inteiros e 21221, a resposta é sim.

# Exercício: Pedra, Papel e Tesoura



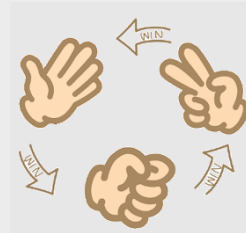
- Vamos continuar o programa “Pedra, Papel e Tesoura”.
  - O jogador só pode digitar 0 (pedra), 1 (papel) ou 2 (tesoura). Imprima a mensagem “Opção inválida” se não for nenhuma dessas opções.
  - Vamos jogar novamente? Se “Sim”, recomece o jogo. Se “Não”, encerre o jogo.

# Exercício: Pedra, Papel e Tesoura



- Vamos continuar o programa “Pedra, Papel e Tesoura”.
  - O jogador só pode digitar 0 (pedra), 1 (papel) ou 2 (tesoura). Imprima a mensagem “Opção inválida” se não for nenhuma dessas opções.
  - Vamos jogar novamente? Se “Sim”, recomece o jogo. Se “Não”, encerre o jogo.

# Exercício: Pedra, Papel e Tesoura

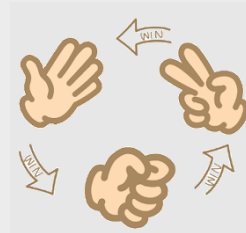


```
jogador1 = int(input("Jogador1, digite 0 p/pedra, 1 p/papel ou 2/tesoura: "))
jogador2 = int(input("Jogador2, digite 0 p/pedra, 1 p/papel ou 2/tesoura: "))

pedra = 0
papel = 1
tesoura = 2

if (jogador1 == jogador2):
    print("Empate! Ninguém ganhou.")
elif (jogador1 - jogador2 == -2) or (jogador1 - jogador2 == 1):
    print("Jogador 1 ganhou.")
else:
    print("Jogador 2 ganhou.")
```

# Exercício: Pedra, Papel e Tesoura

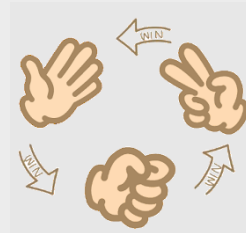


```
jogador1 = int(input("Jogador1, digite 0 p/pedra, 1 p/papel ou 2/tesoura: "))
jogador2 = int(input("Jogador2, digite 0 p/pedra, 1 p/papel ou 2/tesoura: "))

pedra = 0
papel = 1
tesoura = 2

if (0 <= jogador1 <= 2) and (0 <= jogador2 <= 2):
    if (jogador1 == jogador2):
        print("Empate! Ninguém ganhou.")
    elif (jogador1 - jogador2 == -2) or (jogador1 - jogador2 == 1):
        print("Jogador 1 ganhou.")
    else:
        print("Jogador 2 ganhou.")
else:
    print("Opção inválida.")
```

# Exercício: Pedra, Papel e Tesoura



- Vamos continuar o programa “Pedra, Papel e Tesoura”.
  - O jogador só pode digitar 0 (pedra), 1 (papel) ou 2 (tesoura). Imprima a mensagem “Opção inválida” se não for nenhuma dessas opções.
  - Vamos jogar novamente? Se “Sim”, recomece o jogo. Se “Não”, encerre o jogo.

```
pedra = 0
papel = 1
tesoura = 2

jogar_novamente = "Sim"

while (jogar_novamente == "Sim"):

    jogador1 = int(input("Jogador1, digite 0 p/pedra, 1 p/papel ou 2/tesoura: "))
    jogador2 = int(input("Jogador2, digite 0 p/pedra, 1 p/papel ou 2/tesoura: "))

    if (0 <= jogador1 <= 2) and (0 <= jogador2 <= 2):
        if (jogador1 == jogador2):
            print("Empate! Ninguém ganhou.")
        elif (jogador1 - jogador2 == -2) or (jogador1 - jogador2 == 1):
            print("Jogador 1 ganhou.")
        else:
            print("Jogador 2 ganhou.")
    else:
        print("Opção inválida.")

    jogar_novamente = input("Você quer tentar novamente? Digite Sim ou Não")

print("Até a próxima!")
```

# Mais Exercícios =)

- <https://wiki.python.org.br/EstruturaDeRepeticao>: 51 exercícios \o/
- Curso de Python:
  - <https://www.codecademy.com/learn/learn-python>

## **Python IDE Ref.**

- Thonny (Python IDE tool, Univ. de Tartu, Estonia)
  - <https://thonny.org/>