**FACULDADE DE COMPUTAÇÃO E INFORMÁTICA**

**CIÊNCIA DA COMPUTAÇÃO E SISTEMAS DE INFORMAÇÃO**

**Linguagem de Programação I**

**AULA 01: Revisão de Estruturas de Controle de Fluxo**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | http://images.flatworldknowledge.com/ketchen/ketchen-fig05_x001.jpg | Nosso **objetivo** nesta aula é fazer uma revisão das estruturas lógicas de controle de fluxo:   * Escrever programas envolvendo estruturas condicionais simples e compostas; * Escrever programas envolvendo estruturas condicionais encadeadas com **if-else** e com a cláusula **elif**; * Escrever programas envolvendo de **laço definido** realizado pelo **for** do Python; * Exercitar escrever programas envolvendo de **laço indefinido** realizado pelo **while** do Python; * Identificar quando se usa uma laço definido e laço indefinido e aplicar adequadamente a melhor estrutura. | | [http://media.wiley.com/product_data/coverImage300/46/EHEP0020/EHEP002046.jpg](http://www.wiley.com/WileyCDA/WileyTitle/productCd-EHEP002046.html) | A referência para esta aula está na seção **4.3.1 (Laço for)** do **Capítulo 4 (Lists)** do livro:  DIERBACH, C. *Introduction to Computer Science Using Python: A Computational Problem Solving Focus.* 1st Edition, New York: Wiley, 2012 | |
|  |

**Introdução**

* Vimos três formas de controlar o fluxo de um programa: estrutura de controle sequencial, estrutura de controle de seleção ou condicional as estruturas de repetição, também conhecidos por loop (laço) e iterativo.

|  |  |  |
| --- | --- | --- |
| **Estrutura Sequencial** | **Estrutura de Seleção** | **Estrutura de Repetição, Laço ou Iterativo** |
|  |  |  |

* A **estrutura de controle sequencial**, é uma forma implícita de controle, em que as instruções são executadas na ordem em que são escritas.
* A estrutura lógica que permite que o fluxo de execução de um algoritmo possa sofrer desvios é conhecida como **estrutura condicional ou de seleção**.
* Sintaxe da estrutura simples:

**if *condição*:**

***instrução(ões)\_verdadeiro***

* Sintaxe da estrutura composta:

**if *condição*:**

***instrução(ões)\_verdadeiro***

**else:**

***instrução(ões)\_falso***

* A linguagem Python nos fornece duas formas de estruturas com múltiplos possibilidades:
  + Estrutura com múltiplas condições aninhadas ou encadeadas **if-else**
* Sintaxe:

**if *condição*:**

***bloco instruções***

***else:***

***if condição:***

***bloco instruções***

***else:***

***if condição:***

***bloco instruções***

***...***

* + Estrutura com uma única condição **if** e uso de múltiplas cláusulas **elif**
* Sintaxe:

**if <condição 1>:**

**<bloco de código if>**

**elif <condição 2>:**

**<bloco de código elif>**

**else:**

**<bloco de código else>**

* Exemplo: Trecho de código usando **if-else** e **if-elif-else.**

|  |  |
| --- | --- |
|  |  |

**EXERCÍCIO TUTORIADO 1**

1. Escreva u programa que leia três números inteiros e encontra o menor deles, usando **if-else** e **if-elif-else**

* As estruturas de repetição permitem que uma ação seja executada **mais de uma vez** sem que tenhamos que executar novamente o programa.
* Uma estrutura de **controle iterativo, estrutura de repetição ou simplesmente laço** é uma estrutura de controle de fluxo formada por um conjunto de instruções que são executadas um **número determinado de** **vezes** ou até que uma determinada **condição** se torne verdadeira ou falsa.
* Vimos que a linguagem Python classifica as estruturas de controle iterativo em:
  + **loop definido** ou **loop for** quando se sabe previamente o número de vezes que um bloco de instruções será executado.

**for <variavel> in range(inicial, final, passo):**

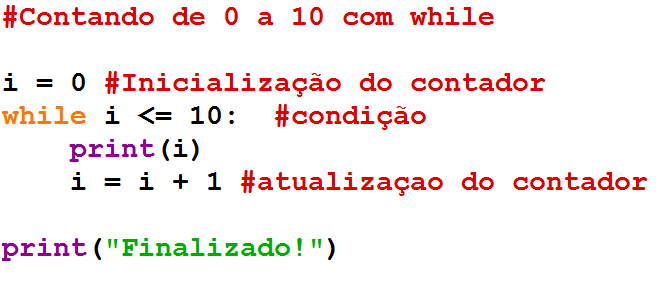
**<bloco de instruções>**

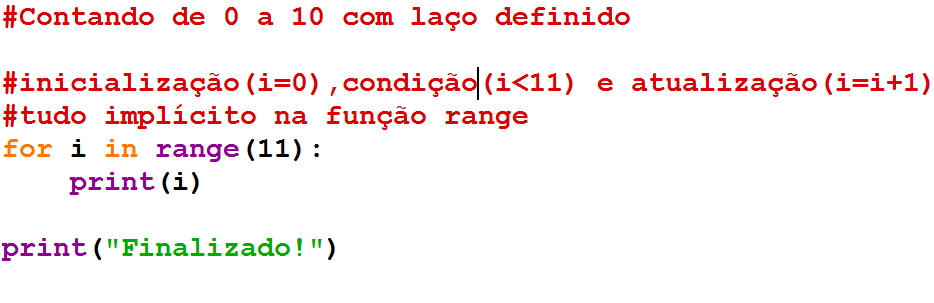
* + **Loop indefinido** quando o bloco de instruções será executado até que uma condição se torne verdadeira ou falsa.

**while <condição>:**

**<bloco de instruções ou bloco verdade>**

* Exemplo: Exibir os inteiros de 0 a 10, usando loop indefinido e loop definido.

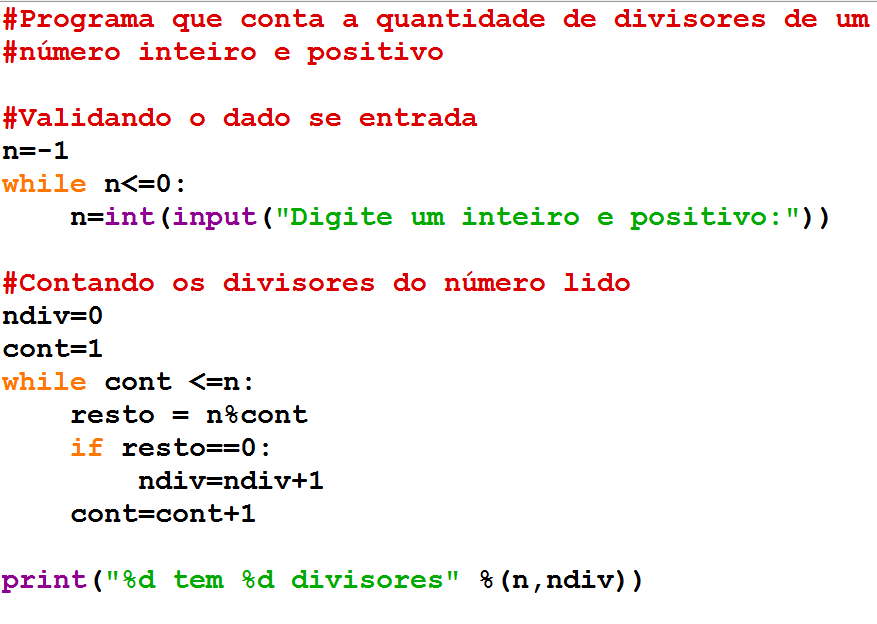




**EXERCÍCIO TUTORIADO 2**

Considere o programa que conta a quantidade de divisores de um número inteiro positivo, usando o laço indefinido.

Reescrever usando um laço definido. O que faz mais sentido?



**EXERCÍCIO TUTORIADO 3**

Escreva um programa que verifica se um número é primo, usando a lógica de contar divisores do exercício anterior e usando a estrutura mais adequada.

**EXERCÍCIO COM DISCUSSÃO EM DUPLAS**

Dados n números inteiros positivos, escreva um programa para calcular a soma dos que são primos.

**EXERCÍCIOS**

1. Faça um programa que calcule e apresente o fatorial de um número inteiro e natural fornecido pelo usuário.

Exemplo: 5! = 5 x 4 x 3 x 2 x 1=120.

Por definição 0! = 1.

1. Dado um número inteiro positivo, escreva um programa para calcular a **soma de seus dígitos**.
2. A série de Fibonacci é formada pela sequência 1, 1, 2, 3, 5, 8, 13, 21, 34, ... Escreva um programa que apresente a série de Fibonacci até o *n-ésimo* termo (n > 0).
3. Escreva um programa que leia uma sequência de números inteiros, encontre e imprima o maior e o menor número. A entrada de um número negativo indica que sequência terminou.
4. Escreva um algoritmo que calcule o MMC (mínimo múltiplo comum) entre dois números naturais.
5. Escreva um algoritmo que calcule o MDC (máximo divisor comum) entre dois números naturais.
6. Escreva um algoritmo que, dados dois números inteiros positivos m e n, determina e escreve, entre todos os pares de números inteiro (x,y) tais que x ≤ m e y ≤ n, um par para o qual o valor da expressão **xy – x2 + y** seja máximo e calcula e escreve também esse máximo.