Python Essentials – Curso da Cisco:

https://edube.org/learn/programming-essentials-in-python/python-essentials-part-1

Parte 1 : Básico

# Módulo 1 : Introdução ao Python e programação de computadores

## Python Essencial: Parte 1

### ~~Python Essencial (Introdução)~~

### ~~Programação~~

### ~~Python Uma ferramenta e não um réptil~~

### ~~Python2 x Python3~~

CPython 🡪 Referência do Python baseado na linguagem C, que foi sua origem

Cython 🡪 Transformar algumas funções Python em linguagem C para ganhar em performance

Jython 🡪 Transformar códigos em Python em Java para melhorar compatibilidade

PyPy 🡪 Python dentro do Python, escrito em RPython, linguagem semelhante e subconjunto do Python. Uma maneira efetiva de testar códigos em Python

## ~~Comece sua Jornada Python~~

### ~~Comece sua Jornada Python (introdução)~~

### ~~Download e instalação do Python~~

### ~~Começando seu trabalho com Python~~

### ~~Como escrever e executar seu primeiro programa (parte1)~~

### ~~Como escrever e executar seu primeiro programa (parte2)~~

### ~~Como estragar e concertar seu código (parte1)~~

### ~~Como estragar e concertar seu código (parte2)~~

### ~~Como estragar e concertar seu código (parte3)~~

## ~~– Interfaces do curso~~

### ~~Sandbox~~

### ~~Interface de Prática~~

### ~~Conclusão do Módulo 1~~

## ~~Teste do Módulo 1~~

~~12/11 – 100% de aproveitamento~~

# Módulo 2 : Tipos de dados, variáveis, operações básicas de entrada-saída, operadores básicos

## ~~Python Essencial (‘Olá Mundo’)~~

### ~~Seu Primeiro Programa~~

~~2.1.1.20 LAB: Formatting the output 🡪 Concluindo este item.~~

### ~~Literais Python~~

~~2.1.2.1 Python literals --- >~~

### ~~Operadores - ferramentas de manipulação de dados~~

### ~~Variáveis ​​- caixas em forma de dados~~

### ~~Um comentário sobre os comentários~~

### ~~Como falar com um computador~~

## ~~Teste do Módulo (Questionário)~~

## ~~Teste do Módulo (Teste)~~

Finalizamos o teste módulo 2 com 100% de aproveitamento. (01/12/20)

# Módulo 3 – Valores booleanos, condicionais, loops, listas e matrizes

## ~~Tomando decisões~~

~~Continuar na seção : 3.1.1.6~~

~~3.1.1.11 – Fazer este LAB LAB: Operadores de comparação e execução condicional~~

## Loops em Python | while

~~3.1.2.1 Loops in Python | while (Continua a partir daqui.)~~

~~3.1.2.4 Loops em Python | para (estou neste ponto)~~

~~3.1.2.8 Controle de loop em Python | quebrar e continuar~~

~~Para teste de condição se o valor da variável for 0 é o mesmo que falso.~~

~~Desta foram a variável pode ser usada diretamente na condição ao invés de != 0 :~~

~~Ex:~~

~~if counter: # No caso se conter for zero é considerado falso. Outro valor é Verdadeiro~~

~~print("The largest number is", largestNumber)~~

~~else:~~

~~print("You haven't entered any number.")~~

~~Exemplos:~~

~~for cont in range (1, 20, 20) --> Conta de de 1 a 19 saltando de 2 em 2.~~

~~break e continue --> break deixa o loop, continue retorna ao contador sem executar funções depois dele.~~

~~3.1.2.10 LAB: The continue statement - the Ugly Vowel Eater (Continuar daqui)~~

~~3.1.2.14 LAB: Essentials of the while loop (fazer este aqui)~~

## ~~Lógica e operações de bits em Python | Operadores bit a bit~~

~~3.1.3.1 Logic and bit operations in Python | and, or, not (começar este aqui)~~

## ~~Listas - Coleções de dados~~

3~~.1.4.1 Listas - coleções de dados 🡪 Começar aqui (21/12/20 -~~

## ~~Classificação de listas simples~~

~~3.1.5.1 Classificação de listas simples - o algoritmo de classificação de bolhas ( 22/12)~~

## ~~Operações em listas~~

3.1.6.1 Operações em listas

Resumos importantes:

1 – Na fatia vai de inicio até final-1 (o ultimo elemento do índice não entra no corte)

1. – Pode-se utilizar elementos negativos no índice, o que remete para o final da lista.

A varredura no entanto segue a ordem crescente na lista

3 – Se o inicio for maior que o final o retorno é uma lista vazia.

Uma fatia desta forma **cria uma nova lista (destino), pegando elementos da lista de origem - os elementos dos índices do início ao**end - 1 .

Dê uma olhada no snippet:

myList = [10, 8, 6, 4, 2]

newList = myList[1:3]

print(newList)

A newListlista terá end - start(3 - 1 = 2) elementos - aqueles com índices iguais a 1e 2(mas não 3).

A saída do snippet é: [8, 6]

myList[start:end]

Repetir:

* starté o índice do primeiro elemento **incluído na fatia** ;
* endé o índice do primeiro elemento **não incluído na fatia.**

É assim que **os índices negativos** funcionam com a fatia:

myList = [10, 8, 6, 4, 2]

newList = myList[1:-1]

print(newList)

A saída do trecho é: [8, 6, 4].

Se startespecificar um elemento mais distante do que o descrito por end(do ponto de vista inicial da lista), a fatia estará **vazia** :

myList = [10, 8, 6, 4, 2]

newList = myList[-1:1]

print(newList)

A saída do trecho é: [].

A delinstrução descrita anteriormente é capaz de **excluir mais do que apenas um elemento da lista de uma vez - ela também pode excluir fatias** :

myList = [10, 8, 6, 4, 2]

del myList[1:3]

print(myList)

Nota: neste caso, a fatia **não produz nenhuma lista nova** !

A saída do trecho é: [10, 4, 2].

## ~~Listas em aplicativos avançados | Array~~

3.1.7.1 Listas em aplicativos avançado (26/12 )

Preenchimento de uma lista com for e if em uma linha apenas

def Lista\_avancado():  
 numeros = [x for x in range (21)]  
 pares = [x for x in range (20) if x % 2 == 0]  
 impares = []  
 print (numeros,input('Numeros. Aguarda...'))  
 for x in numeros:  
 if x % 2 !=0: impares.append(x)  
 print (impares,input("Impares. Aguarda..."))  
 print(pares, input("Pares. Aguarda..."))

# Módulo 4

**4.1.2.1 Como as funções se comunicam com seu ambiente (Continuar daqui 28/12)**