Universidade Federal de Viçosa
Campus Rio Paranaíba
Sistemas de Informação
Prof. Alan Diêgo Aurélio Carneiro

Linguagem Python (Introdução)

Objetivos

Praticar os conceitos de descrição de problemas em algoritmos e transcrevê-los na linguagem de programação Python. Além disso, conhecer o ambiente de programação em linux e entender o seu funcionamento.

Dica do bem: Google é seu amigo.

Exercícios

Desenvolva seus programas em linguagem Python. Codifique-os no editor de texto padrão do linuxe teste sua execução via terminal. Tente resolver os exercícios criando uma **descrição narrativa** (em comentário dentro do código).

Os exercícios estão organizados de uma outra forma para representar uma **entrada** e **saída** de informação para cada programa. A coluna **entrada** representa os valores que o usuário digitaria ao executar o programa, e na coluna **saída** a representação da tela esperada ao executar o programa.

1 - Resolvendo com o professor(a)!

1.1 – Armazene e retorne a lista de entrada ordenada.

Exemplos:

Entrada

A entrada é composta por duas linhas: a primeira linha contém um inteiro que indica a quantidade de elementos da segunda linha; a segunda linha contém uma lista de elementos a ser ordenada.

Saída

Sua função deve imprimir a lista dada em ordem crescente.

Como mostra abaixo:

Entrada	Saída
10 8 7 6 1 2 3 9 10 4 5	1 2 3 4 5 6 7 8 9 10

1.2 – Faça uma função que dada uma frase, retorne o número de palavras da frase. Considere que a frase pode ter espaços no início e no final.

1.3 – Faça uma função que calcule a soma dos n primeiros termos da sequência de Fibonacci, onde n é dado como entrada. A Sequência de Fibonacci consiste em uma sucessão de números, tais que, definindo os dois primeiros números da sequência como 0 e 1, os números seguintes serão obtidos por meio da soma dos seus dois antecessores. Portanto, os números são:
0,1,1,2,3,5,8,13,21,34,55,89,144,233,377,610,987,1597,2584,4181, ...
A sequência de Fibonacci é dada pela seguinte definição:

$$F(n) = \begin{cases} 0, \text{se } n = 0 \\ 1, \text{se } n = 1 \\ F(n-1) + F(n-2), \text{outros casos} \end{cases}$$

2 - Roteiro

2.1 – Escreva uma função que tenha dois parâmetros, uma string e um caractere, e retorne apenas o trecho da string situado entre a primeira ocorrência do caractere até o final da string. Por exemplo, se a entrada for ”abcabc” e ”a”, a saída deve ser ”bcabc”.

2.2 – Faça uma função que dadas duas listas L1 e L2 de tamanho 3, gera uma lista L3 que é formada intercalando os elementos de L1 e L2.
Exemplo: L1 = [1, 3, 5] e L2 = [2, 4, 6] gera L3 = [1, 2, 3, 4, 5, 6].

2.3 – Faça uma função que dada uma lista de números, retorna o maior elemento da lista.

2.4 – Faça uma função que dada uma matriz de inteiros de tamanho qualquer e um número inteiro, conta e retorna quantas vezes aquele número aparece na matriz.

2.5 – Insertion sort, ou ordenação por inserção, é um simples algoritmo de ordenação, eficiente quando aplicado a um pequeno número de elementos. Em termos gerais, ele percorre um vetor de elementos da esquerda para a direita e à medida que avança vai deixando os elementos mais à esquerda ordenados. O algoritmo de inserção funciona da mesma maneira com que muitas pessoas ordenam cartas em um jogo de baralho (fonte: wikipedia)
Não é um método muito eficiente porém sua implementação é simples: supondo que queremos ordenar uma lista de números em ordem crescente. Inicialmente verificamos se o segundo elemento da lista é menor que o primeiro. Caso seja, trocamos eles de posição. Como à esquerda do segundo elemento da lista, só temos

o primeiro elemento, nada mais precisa ser feito.
Passamos então para o terceiro elemento da lista. Temos que compará-lo com o segundo elemento e trocá-los de posição caso o terceiro seja menor. Caso tal troca seja feita, devemos comparar o novo valor que está na segunda posição com o valor da primeira, trocando eles de lugar quando for necessário. Se o terceiro elemento não for trocado com o segundo, isso indica que toda a parte inicial da lista (do primeiro ao terceiro elementos) já está ordenada, e devemos portanto passar para o quarto elemento.
Este processo vai se repetindo até atingirmos o último elemento da lista.
Para entender melhor a idéia, assista aos vídeos:
https://www.youtube.com/watch?v=ROalU379l3U
https://youtu.be/-Z00it6Nkz8
Implemente uma função ordenainsercao que recebe uma lista e a ordena em ordem crescente, utilizando o método de ordenação por inserção.

2.6 – Faça uma função que, dada uma matriz qualquer, gera e retorna a matriz transposta.

2.7 – Faça uma função que receba como parâmetro uma palavra e retorne esta mesma palavra traduzida para a lingua do P. Uma palavra traduzida para a lingua do P quando, após cada vogal da palvra original, é inserida a sequencia de letras p mais a vogal original. Por exemplo:
exemplo → epexepemplopo
entao → epentapaopo
caderno → capadepernopo

2.8 Faça uma função que dado um número, verifique se este número é primo ou não.