INF101 – Introdução à Programação II

Roteiro de Prática: 23 de agosto de 2018

Introdução

O objetivo principal desta prática é familiarizar o estudante com a utilização da estrutura de dados *lista* diretamente fornecida pela linguagem Python. Para ilustrar o uso de listas, vamos considerar o problema de ordenar uma lista. Sabemos que uma ordenação pode ser *crescente* (ascendente) ou *decrescente* (descendente). Para fixar a ideia, nesta aula, vamos considerar ordem crescente. Já fica, de antemão, o convite para que o aluno faça, como exercício depois, a ordem decrescente. Existem vários algoritmos clássicos de ordenação. Nesta aula, vamos considerar um dos mais simples, denominado *seleção direta*. Infelizmente, ele não é eficiente para ser usado em produção, no mundo real. A não ser que a lista seja pequena, de tamanho, digamos, menor que 1000 elementos. Depois aprenderemos métodos de ordenação mais eficientes que poderão ser implementados sem problema em *software* de produção.

Algoritmo de ordenação por seleção direta

O algoritmo consiste em três passos básicos. Supondo uma lista L de qualquer tipo ordenável totalmente, temos:

1. Determine o índice `j` do menor elemento considerando toda a lista inicialmente, isto é, do índice 0 até o último índice que todos sabemos ser o tamanho da lista menos um.
2. Troque de posição o elemento de índice `j` com o primeiro.
3. Repita os dois primeiros passos considerando agora a parte da lista que vai do índice 1 até o último e troque o elemento mínimo agora de índice `j` com o segundo. Faça isto até que reste apenas um elemento na parte ainda não ordenada da lista.

Refinamento do algoritmo

Para h de 0 até `len(L) - 1` (exclusive), faça:
```
   j = minimo(L, h)
   troque L[j] e L[h]
```
Fim_para

Instruções

1. Abra o IDLE e crie um novo arquivo fonte denominado `p03.py`. Não se esqueça de salvá-lo de tempos em tempos, porque pode ocorrer falha de energia elétrica durante a aula prática.
2. Digite os comentários obrigatórios (nome, matrícula, data e uma breve descrição sobre o que o programa faz).
3. Estruture seu programa em três funções: `main()`, `minimo()` e `ordena()`.
4. Para hoje, a função `main()` pode ser bem simples: crie uma lista com alguns elementos (desordenados) para teste. Uma sugestão: 36, 18, 43, 9, 18, 25, 14. Chame a função `ordena` passando a lista como parâmetro. E, finalmente, imprima a lista que agora deve estar ordenada crescentemente.
5. Implemente a função `minimo` com dois parâmetros: uma lista e um índice a partir do qual o mínimo será verificado. A função deve retornar o índice do elemento mínimo obtido da parte da lista considerada no momento.
6. Implemente a função `ordena` usando a algoritmo de ordenação dado acima. A função deve ter um parâmetro contendo a lista a ser ordenada. A função `ordena` não precisa retornar

nada, pois, se o parâmetro for uma lista, ele será passado por referência, e, no final da execução da função, o próprio parâmetro conterá a lista ordenada.

7. Não se esqueça de chamar a função `main()` no final de seu código para desencadear todo o processo de execução.
8. Por segurança, teste seu programa com, pelo menos, mais outra lista. Sugestão: 43, 36, 25, 18, 18, 14, 9.
9. Se seu programa entrar em *laço infinito,* digite CTRL-C na janela do *Shell*, para interromper a execução do programa. Após, conserte-o.

☞ Não se esqueça de preencher o cabeçalho do código fonte com seus dados, a data de hoje e uma breve descrição do programa.

Após certificar-se de que seu programa esteja correto, envie o arquivo do programa fonte (`p03.py`) através do sistema de entrega do LBI.