

## Lista de Exercícios N° 1

Esta lista deve ser desenvolvida individualmente, seguindo as especificações contidas no arquivo 00\_ProcListas.pdf disponível na pasta de arquivos do canal Geral da equipe na plataforma Teams. A entrega ao professor deverá ocorrer **até o início da aula do dia 19 de outubro de 2022**.

- 1) Faça um programa que implementa uma lista linear duplamente encadeada contendo os dados de cidades informadas pelo usuário. Deverão ser armazenados o nome da cidade (uma palavra de até 30 caracteres úteis), sua população (um inteiro), sua área territorial, PIB e IDH (todos do tipo float). Encerrar a entrada de dados quando o usuário informar “FIM” para o nome do município e imprimir a lista tanto a partir do início como a partir do final. Após a impressão do conteúdo original da lista, o programa deverá executar um *looping* onde, a cada ciclo, um valor de IDH é informado e todas as ocorrências de cidades na lista que possuem IDH igual ou superior a esse valor devem ser excluídas. Após a exclusão, imprimir a lista novamente, tanto a partir do início como a partir do final. Se não for encontrado nenhum nó com valor de IDH igual ou superior ao informado, emitir mensagem apropriada e não imprimir a lista. Encerrar o programa quando o usuário informar um IDH negativo. Organize o programa em arquivo cliente, de interface e de implementação.
- 2) Faça uma versão do programa que monta a lista de cidades no exercício anterior de maneira que a lista seja implementada por contiguidade. Mantenha o programa cliente igual ao original, modificando apenas os arquivos de interface e de implementação da lista.
- 3) Implemente um programa que constrói duas listas encadeadas de valores inteiros, encerrando a entrada de dados de cada lista assim que o valor -999 for informado. Após a entrada de dados, imprimir o conteúdo das duas listas. Em seguida o programa deverá excluir da segunda lista todos os elementos cujo valor exista na primeira lista. Exibir novamente o conteúdo das duas listas após o processamento.

Situação inicial:

|          |       |   |    |    |    |    |     |
|----------|-------|---|----|----|----|----|-----|
| Lista 1: | 2     | 9 | 23 | 40 |    |    |     |
| Lista 2: | -1000 | 0 | 9  | 12 | 23 | 25 | -20 |

Situação final:

|          |       |   |    |    |     |
|----------|-------|---|----|----|-----|
| Lista 1: | 2     | 9 | 23 | 40 |     |
| Lista 2: | -1000 | 0 | 12 | 25 | -20 |

- 4) Fazer um programa em C que recebe uma *string* de até 20 caracteres contendo uma expressão aritmética digitada pelo usuário. Essa expressão poderá conter 3 tipos de delimitadores de escopo e precedência: parênteses, colchetes e chaves. O programa deverá verificar se os delimitadores estão corretamente balanceados ou não, utilizando uma pilha para essa finalidade.

Exemplos de expressões válidas:

- $(A+B) / C$
- $A + (B / C)$
- $[a-2] * (5+c)$
- $\{a * [c+d * (4*x)]\}$
- $(A)$

Exemplos de expressões inválidas:

- $) A+B (/ C$
- $A+B / C)$
- $[a-2] * (5+c]$
- $\{a * [c+d * (4*x)]\}$
- $(A$

**Observações:**

- a) Considere que não há qualquer tipo de hierarquia entre os delimitadores, ou seja, o usuário pode, a qualquer momento, iniciar uma expressão com qualquer um dos 3 tipos permitidos.

### Lista de Exercícios N° 1

- b) Usar subrotinas para empilhar e desempilhar os elementos.
  - c) Use a pilha para nela acrescentar (apenas) os delimitadores de abertura, ou seja, '(', '[', '{' encontrados na *string*. Ao encontrar um delimitador de fechamento, ou seja, ')', ']', '}', retire o elemento do topo da pilha e verifique se ele é compatível com o delimitador que foi encontrado na cadeia. Se for, prossiga o processamento, se não for, encerre emitindo mensagem de erro. Se, ao final, a pilha estiver vazia, concluímos que a expressão contida na *string* é válida, pois está balanceada, e o programa deverá emitir uma mensagem informando isso.
- 5) Implemente um programa que recebe do usuário dois conjuntos de palavras e armazena cada conjunto como uma lista encadeada. A entrada de dados de cada conjunto deverá ser encerrada assim que o valor "\*" for informado. Após a entrada de dados, imprimir o conteúdo das duas listas. Em seguida o programa deverá exibir os valores correspondentes às operações de União, Intersecção e Diferença dos dois conjuntos informados, conforme ilustrado no exemplo a seguir:

Conjunto 1:

{ bala goiaba tomate viola banana }

Conjunto 2:

{ cidade laranja banana goiaba carro cachorro }

União dos conjuntos

{ bala goiaba tomate viola banana cidade laranja carro cachorro }

Intersecção dos conjuntos

{ goiaba banana }

Diferença Conjunto1 – Conjunto2

{ bala tomate viola }

Diferença Conjunto2 – Conjunto1

{ cidade laranja carro cachorro }