UNIVERSIDADE FEDERAL DE JUIZ DE FORA – INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

2ª avaliação de Estrutura de Dados e Laboratório de Programação II – 12/12/2022

Nome: _____ **Matrícula:** _____

Ler atentamente, por favor, as instruções abaixo:

- Fazer o download do arquivo **Avaliacao2EDLab2-2022-2.zip** do *site* do curso e descompactá-lo em sua máquina. Esse arquivo contém todos os códigos para o desenvolvimento da prova.
- A resposta de cada questão deve, **obrigatoriamente**, estar entre cada par de marcadores (//Qi, //-Qi). Dessa forma, a questão 1 está entre //Q1 e //-Q1, a questão 2 entre //Q2 e //-Q2 e assim por diante. Não remover, em hipótese alguma, estes marcadores de início e fim de questões da sua prova. Caso sua solução tenha mais de uma função ou operação, elas devem estar entre esses marcadores, obrigatoriamente.
- Colocar no arquivo `main.cpp` seu nome completo e número de matrícula.
- A prova é **individual** e **sem qualquer tipo de consulta**.
- Há apenas um projeto do `Code::Blocks` usado na prova.
- Antes de sair do laboratório, enviar ao servidor – usando a janela de *upload* – cada arquivo de código que contém as respostas das questões da sua prova. Aguarde um momento para ver as suas respostas de cada questão da prova.
- **O desenvolvimento e envio do código são de inteira responsabilidade do aluno!**
- Para o envio da prova, usar o servidor: http://172.18.40.119:8080/edlab2ufjf/.

---

1) Implementar a operação `ListaCont* ListaCont::moveParesInvertidos();` que cria dinamicamente uma nova lista e move para esta nova lista todos os elementos pares da lista interna de modo que fiquem em ordem invertida na nova lista. Ao final, a lista interna deve ficar apenas com os elementos ímpares em sua ordem original e a operação deve retornar a lista criada dinamicamente.

   Exemplo: Considere uma lista inicial com os valores `[2, 3, 4, 5, 6, 7, 8]`. Após a execução da operação `moveParesInvertidos()`, a lista interna deve ficar com `[3, 5, 7]` e a operação deve retornar uma nova lista com `[8, 6, 4, 2]`.

   [25]

---

2) Implementar a operação `void ListaEncad::insereDepoisMenor(int val);` que insere o valor `val` depois do nó com menor valor de uma lista simplesmente encadeada com descritor. A operação deve percorrer a lista uma única vez. Em caso de mais de um valor menor, considerar a primeira ocorrência. Imprima uma mensagem de erro caso não seja possível inserir o nó. [25]

   Exemplo: Considere uma lista com os valores `lEnc=[3, 2, 5, 12, 7, 9]` e `val = 10`. O valor `10` deve ser inserido depois do valor `2` e a lista resultante é `lEnc=[3, 2, 10, 5, 12, 7, 9]`.

---

3) Implementar a operação `void ListaDupla::anexar(ListaDupla *lde)` para, dado um ponteiro para uma LDE `lde`, anexar a `lde` com a lista intrínseca, nesta ordem. Alterar apenas os ponteiros necessários para a anexação. Não pode haver movimentações de valores dos nós das listas e nem pode criar uma nova LDE e adicionar os nós das listas nessa nova lista. A lista `lde` deve se tornar vazia no fim da operação. [25]
   Exemplo: Considere a lista intrínseca inicial `L=[1,2,3,4,5]` e `lde=[3,8,15]`. Após a execução de `L.anexar(lde)`, obtém-se: `L`→`[3,8,15,1,2,3,4,5]` e `lde`→`[ ]`.

---

4) Implemente a função `FilaEncad* intercala(FilaEncad *f1, FilaEncad *f2)` no programa principal, que recebe duas filas preenchidas como parâmetros e as intercala utilizando funções do TAD `FilaEncad` e retorna o resultado em uma terceira fila. Observe que a função deve funcionar para intercalar duas filas que podem conter tamanhos diversos. Assuma que no programa principal a chamada ficaria assim: `f3 = intercala(f1, f2);`. [25]
   Exemplo: considerando uma fila `f1` contendo os valores : `[1 2 3 4]` e a outra fila `f2` contendo os valores: `[5 6 7 8 9 10]`. A fila intercalada `f3` deve conter: `[1 5 2 6 3 7 4 8 9 10]`.