Laboratório de Estrutura de Dados
Avaliação 3 (A3) - SEGUNDA CHAMADA
TAD e Listas

Semestre: 2022.2

Data e local de entrega: 05/11/2022

Valor: 8 pontos

Observações (LEIA COM ATENÇÃO)

- A atividade é **individual**.
- A **linguagem C** deverá ser **utilizada** para implementar a solução para os exercícios.
- **Qualquer** ambiente de desenvolvimento poderá ser **empregado** para desenvolver a atividade.
- A entrega **deverá** ser feita no SIGAA por meio de um **único** arquivo (compacte os arquivos em uma pasta). **Todos** os códigos-fonte (.c) e arquivos cabeçalhos (.h) deverão ser entregues.

EXERCÍCIOS

1. [2 pontos] **Defina uma função** na lista linear sequencial de inteiros que retorne a quantidade de números ímpares presentes na lista. Utilize o seguinte protótipo para a função:

   `int qtde_impares_lst(lista *l).`

   Para testar a função, **desenvolva um programa** para criar uma lista com capacidade 30. Insira na lista 25 valores "aleatórios" no intervalo [-10, 10]. Utilize a função `qtde_impares_lst` para verificar quantos valores ímpares estão presentes na lista. **Mostre** o conteúdo da lista e a quantidade de valores ímpares, conforme abaixo. Segue dois exemplos de execução do programa.

Exemplos de execução
`Lista: -7 3 -5 1 -3 2 8 3 -10 9 4 3 -4 5 -6 9 8 7 1 2 10 -9 -6 4 2` `Qtde de valores impares: 13`
`Lista: 3 5 -1 -3 3 2 7 -9 0 4 2 3 -4 -5 -6 1 3 2 4 -10 5 9 6 3 7` `Qtde de valores negativos: 15`

2. [2 pontos] **Defina uma função** na lista linear encadeada de inteiros (`lista_enc`) com o seguinte protótipo:

   `void oposto_lst(lista *l).`

   Esta função deve alterar o valor de cada célula da lista para o inteiro oposto, ou seja, 4 se torna -4, -9 se torna 9, 6 se torna -6 e, assim, sucessivamente.

   Para testar a função, **desenvolva um programa** para criar uma lista linear encadeada com capacidade 50. Insira na lista 20 valores "aleatórios" no intervalo [-12, 10]. Utilize a função `oposto_lst` para alterar os valores presentes na lista. Mostre o conteúdo da lista antes e após a alteração. Segue abaixo dois exemplos de execução do programa.

Exemplos de execução
`Lista: 4 -7 0 7 5 -9 -11 -14 -5 -11 3 -10 8 9 -3 -12 0 5 4 2` `Lista: -4 7 0 -7 -5 9 11 14 5 11 -3 10 -8 -9 3 12 0 -5 -4 -2`
`Lista: -8 -10 5 4 -12 -6 8 -5 6 5 -6 -11 -3 10 -8 -2 7 -1 -10 6` `Lista: 8 10 -5 -4 12 6 -8 5 -6 -5 6 11 3 -10 8 2 -7 1 10 -6`

3. [2 pontos] Crie uma estrutura de dados para representar uma **lista linear sequencial de pontos**. Use como base a lista linear sequencial de inteiros e adapte-a para armazenar pontos ao invés de inteiros. **Ajuste** as funções da lista para manipular pontos. **Altere** o arquivo de testes desenvolvido em aula para **testar a lista de pontos** criada.

4. [2 pontos] Defina uma função na lista sequencial circular (`lista_circ`) para dobrar o valor de todos os números pares presentes na lista.

   Para testar a nova função, **desenvolva um programa** para criar uma lista com capacidade 30. Insira na lista os valores de -10 a 10, ou seja, os valores -10, -9, -8, ..., 8, 9, 10. Remova 5 elementos do início da lista. Dobre o valor de todos os pares presentes na lista usando a função criada. Mostre o conteúdo da lista antes e depois de usar a função.

Exemplo de execução

```
Lista circular:
5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20       <-- índices
-5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10            <-- valores

Lista circular:
5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20       <-- índices
-5 -8 -3 -4 -1 0 1 4 3 8 5 12 7 16 9 20          <-- valores
```