Lista 1 – Estrutura de Dados

Para uma melhor organização, cada questão de implementação deverá ser criada em um projeto separado, exemplo:

Projeto 1: questão 1.1	Projeto 6: questão 6
Projeto 2: questão 2.1	Projeto 7: questão 7.1
Projeto 3: questões 3.1 e 3.3	Projeto 8: questão 8
Projeto 4: questão 4	Projeto 9: questão 9
Projeto 5: questão 5	Projeto 10: questão 10

1 - *Considere um array de inteiros positivos de tamanho 100. Faça as seguintes questões:*

1.1 - *Sabendo que neste array é aceito a inserção de elementos iguais. Crie os seguintes métodos:*

a) Adicionar um elemento no início da lista;
b) Adicionar um elemento no final da lista;
c) Adicionar um elemento na n-ésima posição da lista;
d) Remover o primeiro elemento da lista;
e) Remover o último elemento da lista;
f) Remover o n-ésimo elemento da lista;
g) Imprimir os elementos da lista de forma iterativa;
h) Retornar o número de elementos da lista;

1.2 - *Qual a complexidade de cada método?*

1.3 - *Qual o maior problema quanto a fixação de tamanho do array? Como você resolveria esse problema?*

2 - Considere uma lista ***encadeada*** *de inteiros positivos com apenas um ponteiro denominado* ***inicio*** *que aponta para primeiro nó da lista (início da lista). Faça as seguintes questões:*

2.1 – *Sabendo que nesta lista é aceito a inserção de elementos iguais. Crie os seguintes métodos:*

a) Adicionar um elemento no início da lista;
b) Adicionar um elemento no final da lista;
c) Adicionar um elemento na n-ésima posição da lista;
d) Remover o primeiro elemento da lista;
e) Remover o último elemento da lista;
f) Remover o n-ésimo elemento da lista;
g) Imprimir os elementos da lista de forma iterativa;
h) Retornar o tamanho da lista;

2.2 - *Qual a complexidade de cada método? Em relação a questão 1, a complexidade dos métodos mudam? Qual/Quais? Por que?*

2.3 - *Desenhe e explique como funciona passo a passo a* ***inserção*** *e a* ***remoção*** *do n-ésimo elemento desta lista* ***encadeada****.*

3 - Considerando ainda que nesta lista ***encadeada*** *é aceito a inserção de elementos iguais,*

*modifique a lista colocando agora um ponteiro denominado **fim** que apontará para o final da lista.*

3.1 - *Refaça os métodos da questão 2.1 efetuando as modificações necessárias para que o ponteiro fim sempre aponte para o final da lista.*

3.2 - *Neste caso a complexidade de cada método muda? Qual/Quais? Por que?*

3.3 - *Crie um método que imprima os elementos da lista **encadeada** na ordem invertida. É possível fazer este método sem recursão? Por que?*

Exemplo

Elementos da lista: {1, 4, 2, 5, 1, 6}
Invertida: {6, 1, 5, 2, 4, 1}

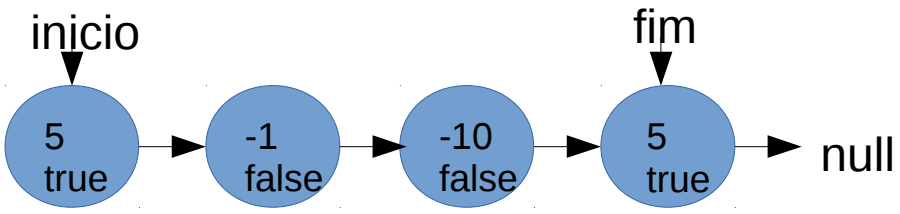
*4 - Considerando ainda que nesta lista **encadeada** é aceito a inserção de elementos iguais , faça as seguintes modificações na lista encadeada com ponteiro no **inicio** e no **fim**:*

(a) Troque os métodos (a), (b) e (c) da questão 3.1, por um único **método adicionar**, que adiciona os elementos da lista sempre de forma ordenada (não deve-se usar nenhum algoritmo de ordenação para ordená-la, apenas uma lógica para que os elementos sempre sejam adicionados de forma ordenada).

(b) Troque os métodos (d), (e) e (f) da questão 3.1, por um único **método remover**, que remove o n-ésimo elemento da lista mantendo-a sempre de forma ordenada (não deve-se usar nenhum algoritmo de ordenação para ordená-la, apenas uma lógica para que os elementos sempre sejam adicionados de forma ordenada).

(c) Considerando agora que a lista sempre estará ordenada, crie um método que recebe duas listas encadeadas ordenadas e retorna o valor verdadeiro se as duas listas são iguais; e falso se as duas listas forem diferentes.

5 - *Considerando ainda que nesta lista **encadeada** é aceito a inserção de elementos iguais, modifique a lista **encadeada** para armazenar inteiros (positivos e negativos), e um atributo booleano que é verdadeiro se o número inteiro armazenado é positivo; e falso se ele é negativo.*



(a) Refaça os métodos da questão 3.1 efetuando as modificações necessárias.
(b) Neste caso a complexidade de cada método muda? Qual/Quais? Por que?
(c) Crie um método que busca um elemento na lista pelo seu valor inteiro e não pela sua posição, retornando o atributo booleano deste elemento.
(d) Qual a complexidade deste método? Por que?
(e) Crie um método que remova todos os elementos negativos desta desta lista.

*6 - Considere agora que na lista **encadeada não** serão aceitos números inteiros repetidos.*

(a) Refaça os métodos da questão 3.1 efetuando as modificações necessárias.
(b) Neste caso a complexidade de cada método muda? Qual/Quais? Por que?

7.1 – Dada *uma lista **duplamente encadeada** com ponteiros **inicio** e **fim**. Considere que esta lista contenha um atributo **ContaBancaria** que tem um nome (dono da conta), número da conta, senha (apenas dígitos inteiros) e um saldo.*

*(*a) Refaça os métodos da questão 3.1 efetuando as modificações necessárias.
(b) Neste caso a complexidade de cada método muda? Qual/Quais? Por que?

7.2 - *Desenhe e explique como funciona passo a passo a **inserção** e a **remoção** do n-ésimo elemento desta lista **duplamente encadeada**.*

8 - *Considere o seguinte programa para simular o gerenciamento de contas de um banco (uma lista **duplamente encadeada** de contas bancárias).*

(a) Crie um método denominado **cadastrarConta** que recebe um nome, um número e uma senha, cria o objeto conta com saldo igual a zero; e adiciona de **forma ordenada** em uma lista **duplamente encadeada** que simula as contas de um banco (você deverá modificar a **lista duplamente encadeada**, criada na questão 7.1(a), para que tenha apenas um método adicionar ordenado).

(b) Crie um método denominado **visualizar** que não possui parâmetros; e deve apresentar o número e saldo de todas as contas cadastradas até o momento.

(c) Crie um método denominado **removerConta** que recebe o número de uma conta bancária; e remove a conta bancária da **lista duplamente encadeada mantendo-a ordenada** (você deverá modificar as **lista duplamente encadeada**, criada na questão 7.11(a), para que tenha apenas um método remover ordenado).

(d) Crie um método denominado **depositar** que recebe o número de uma conta bancária, uma senha e um valor; e credita o valor informado ao saldo da conta bancária. Para isso você deve pesquisar a conta bancária pelo número na lista duplamente encadeada que simula a lista de contas bancárias.

(e) Crie um método denominado **sacar** que recebe o número de uma conta bancária, uma senha e um valor; e debita o valor informado ao saldo da conta bancária. Para isso você deve pesquisar a conta bancária pelo número na lista duplamente encadeada que simula a lista de contas bancárias.

(f) Crie um método denominado **saldo** que recebe o número de uma conta bancária e uma senha; e informa o valor do saldo da conta bancária. Para isso você deve pesquisar a conta bancária pelo número na lista duplamente encadeada que simula a lista de contas bancárias.

9 – **Crie** *um lista **duplamente encadeada circular**, ou seja, só existirá um ponteiro que apontará para o "início" da lista circular, onde o último elemento desta lista deverá apontar para o início.*

*(*a) Refaça os métodos da questão 1.1 efetuando as modificações necessárias.
(b) Neste caso a complexidade de cada método muda? Qual/Quais? Por que?

10 – *(DESAFIO) – Maratona de Programação: faça a questão abaixo utilizando um lista encadeada ou um lista duplamente encadeada.*

Juliano é fã do programa de auditório Apagando e Ganhando, um programa no qual os

participantes são selecionados através de um sorteio e recebem prêmios em dinheiro por participarem. No programa, o apresentador escreve um número de N dígitos em uma lousa. O participante então deve apagar exatamente D dígitos do número que está na lousa; o número formado pelos dígitos que restaram é então o prêmio do participante.

Juliano finalmente foi selecionado para participar do programa, e pediu que você escrevesse um programa que, dados o número que o apresentador escreveu na lousa, e quantos dígitos Juliano tem que apagar, determina o valor do maior prêmio que Juliano pode ganhar.

Entrada

A entrada contém vários casos de teste. A primeira linha de cada caso de teste contém dois inteiros N e D $(1 \leq D < N \leq 10^5)$, indicando a quantidade de dígitos do número que o apresentador escreveu na lousa e quantos dígitos devem ser apagados. A linha seguinte contém o número escrito pelo apresentador, que não contém zeros à esquerda.

O final da entrada é indicado por uma linha que contém apenas dois zeros, separados por um espaço em branco.

Os dados devem ser lidos da entrada padrão.

Saída

Para cada caso de teste da entrada seu programa deve imprimir uma única linha na saída, contendo o maior prêmio que Juliano pode ganhar.

O resultado de seu programa deve ser escrito na saída padrão.

Exemplo de entrada	Exemplo de saída
4 2 3759 6 3 123123 7 4 1000000 0 0	79 323 100