**Nome: Eduardo Lucas Lemes Januário CT3037568**

**Nome: Guilherme Batista de Souza CT3037274**

**Trabalho Avaliativo II - Parte Teórica**

1. Considere uma fila circular inicialmente vazia, com capacidade ilimitada (isto é, sem restrição de tamanho). Realize as seguintes operações na ordem apresentada:

enfileirar(10);

enfileirar(20);

enfileirar(30);

desenfileirar();

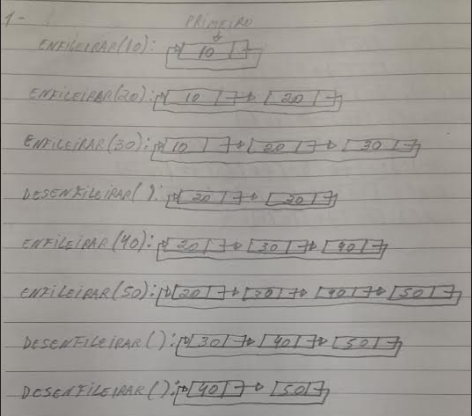
enfileirar(40);

enfileirar(50);

desenfileirar();

desenfileirar();

Para cada operação apresente o estado atual da fila.



2. Considere uma fila com prioridades, em que elementos com menor valor de prioridade são removidos antes (ou seja, prioridade 0 é mais alta que 1, e assim por diante). Em caso de empate de prioridade, o critério de desempate é a ordem de chegada (FIFO). Realize as seguintes operações:

inserir(5, prioridade=3);

inserir(8, prioridade=1);

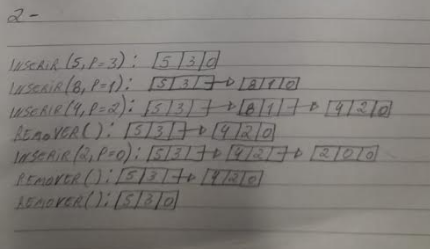
inserir(4, prioridade=2);

remover();

inserir(2, prioridade=0);

remover(); remover();

Para cada operação apresente o estado atual da fila de prioridade.



3. Considere uma pilha inicialmente vazia. Realize as seguintes operações na ordem dada:

empilhar(10);

empilhar(20);

empilhar(30);

desempilhar();

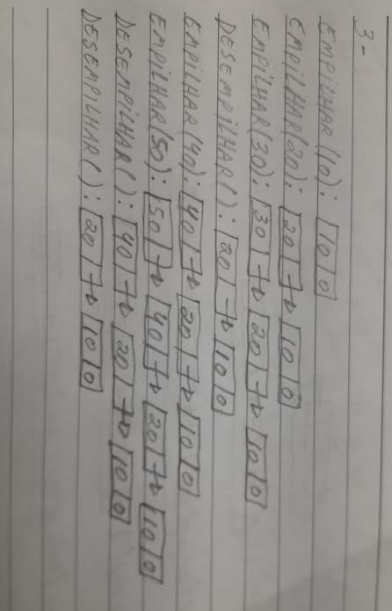
empilhar(40);

empilhar(50);

desempilhar();

desempilhar();

Para cada operação mostre o estado da pilha.



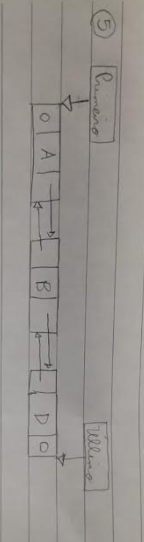
4. O que são Tipos Abstratos de Dados (TADs)? Explique a importância de se trabalhar com TADs na construção de algoritmos e estruturas de dados. Cite exemplos e relacione com pelo menos uma estrutura estudada na disciplina. Escreva uma breve dissertação (entre 10 e 15 linhas).

Os tipos abstratos de dados (TADs) são uma especificação de um conjunto de dados e operações que podem ser executados sobre esses dados. A importância de se trabalhar com TADs é o fato de ser crucial para a organização, eficiencia e manutenção de codigo, permitindo a abstração de informações e a criação de componentes reutilizaveis e flexiveis. Alguns exemplos de TADs são pilhas, fias e deques. Uma piha segue o principio LIFO (Last In, Fist Out), ou seja, o ultimo elemento a entrar é o primeiro a ser removido. A pilha pode ser implementada de várias maneiras, como array e lista ligada, mas isso não afeta seu conceito abstrato.

5. Considere uma lista duplamente encadeada, que inicialmente está vazia. Cada elemento da lista tem um conteúdo, além de ponteiros para o próximo elemento e para o anterior. Dois ponteiros indicam, respectivamente, o primeiro e o último elemento da lista. A tabela abaixo apresenta as operações ocorridas nesta lista durante um certo período de tempo.

| **Ordem** | **Operação** | **Conteúdo do Nó** | **Endereço do Nó** |
| --- | --- | --- | --- |
| 1 | Inserção no Inicio | “A” | #1111 |
| 2 | Inserção no Fim | “B” | #2222 |
| 3 | Inserção no Inicio | “C” | #3333 |
| 4 | Inserção no Fim | “D” | #4444 |
| 5 | Remoção no Inicio | – | – |
| 6 | Inserção no Inicio | “E” | #5555 |
| 7 | Inserção no Fim | “F” | #6666 |
| 8 | Remoção no Fim | – | – |
| 9 | Remoção no Inicio | – | – |

Preencha, na figura abaixo, os valores dos elementos da lista (conteúdos e ponteiros). Indique claramente onde estão o primeiro e o último elemento da lista.



6. Aplique o raciocínio da prova indutiva finita, conforme estudado em sala, para desenvolver um algoritmo recursivo que encontra o menor valor em um conjunto de dados (representado por uma lista ou vetor).

Escreva a ideia central da prova indutiva:

● Caso base;

(N = 1)

● Hipótese de indução;

N - 1

● Passo indutivo (redução).

N = K + 1, onde K representa o tamanho do vetor

Escreva o pseudocódigo do algoritmo recursivo que resolve o problema proposto.

Função MenorValor(vetor, n)

Se n = 1 então

retorne vetor[0]

Senão

menor ← MenorValor(vetor, n - 1)

Se vetor[n - 1] < menor então

retorne vetor[n - 1]

Senão

retorne menor

Justifique por que o algoritmo está correto com base no raciocínio da prova.

O algoritmo compara o último elemento (indice n - 1) com o menor valor dos anteriores, encontrado de forma recursiva.