Atividade: Implementação de Pilha, Fila e Lista com Exemplos do Mundo Real

1. Pilha: Livros em uma Mesa de Estudo

Contexto: Imagine que você tem uma pilha de livros em sua mesa de estudo. Você só pode pegar ou colocar um livro no topo da pilha.

Tarefas:

1. Crie uma pilha para representar esses livros.
2. Empilhe um novo livro (representado por um título ou ID).
3. Desempilhe o livro do topo.
4. Recupere quantos livros estão na pilha.
5. Por fim, remova todos os livros da pilha (limpe sua mesa!).

2. Fila: Pessoas em uma Fila de Banco

Contexto: Imagine que várias pessoas estão esperando para serem atendidas em um banco. A pessoa que chega primeiro é atendida primeiro (FIFO).

Tarefas:

1. Crie uma fila para representar as pessoas.
2. Enfileire uma nova pessoa (representada por um nome ou ID).
3. Desenfileire a pessoa que está na frente da fila.
4. Recupere quantas pessoas estão esperando na fila.
5. Ao final do dia, quando o banco fechar, limpe a fila (todas as pessoas foram atendidas).

3. Lista: Lista de Reprodução de Músicas

Contexto: Imagine uma lista de reprodução em um aplicativo de música. Você pode adicionar ou remover músicas em qualquer posição, e também pode escolher uma música específica para tocar.

Tarefas:

1. Crie uma lista para representar sua lista de reprodução.
2. Insira uma música na posição 'x' de sua lista.
3. Remova a música na posição 'y'.
4. Recupere e reproduza a música na posição 'z'.
5. Adicione uma música no início da lista e outra no final.
6. Recupere quantas músicas você tem em sua lista de reprodução.
7. Depois de uma grande festa, decida começar do zero e limpe sua lista de reprodução.

Orientações:

1. Você pode escolher a linguagem de programação de sua preferência.
2. Recomendamos o uso de funções ou métodos para cada operação (por exemplo, **push()**, **pop()**, **enqueue()**, etc.).

3. Após implementar, tente correlacionar suas operações de código com as ações que você faria no exemplo do mundo real. Isso ajudará a consolidar sua compreensão.
4. Criar conforme os algoritmos em anexa.
5. Pode ser em C ***ou em*** Python.
6. Comentar todos as linhas dos códigos.

PADRÃO DE ENTREGA
7585ba7d
Padrão do nome do arquivo:
7585ba7d – Nome do aluno – Iniciais da disciplina

Pilha

```
INÍCIO

ESTRUTURA Nó:
    dado: INTEIRO
    próximo: PONTEIRO PARA Nó

ESTRUTURA Pilha:
    topo: PONTEIRO PARA Nó
    tamanho: INTEIRO

FUNÇÃO criarPilha() -> PONTEIRO PARA Pilha:
    pilha = NOVA Pilha
    pilha->topo = NULL
    pilha->tamanho = 0
    RETORNA pilha

FUNÇÃO empilhar(pilha, valor):
    novoNó = NOVO Nó
    novoNó->dado = valor
    novoNó->próximo = pilha->topo
    pilha->topo = novoNó
    pilha->tamanho = pilha->tamanho + 1

FUNÇÃO desempilhar(pilha) -> INTEIRO:
    SE pilha->topo É NULL:
        RETORNA -1
    FIMSE

    nóTemp = pilha->topo
    valor = nóTemp->dado
    pilha->topo = nóTemp->próximo
    LIBERA nóTemp
    pilha->tamanho = pilha->tamanho - 1
    RETORNA valor

FUNÇÃO tamanhoPilha(pilha) -> INTEIRO:
    RETORNA pilha->tamanho

FUNÇÃO destruirPilha(pilha):
    ENQUANTO pilha->topo NÃO É NULL:
        desempilhar(pilha)
    FIMENQUANTO
    LIBERA pilha

FIM
```

Fila

```
INÍCIO

ESTRUTURA Nó:
    dado: INTEIRO
    próximo: PONTEIRO PARA Nó

ESTRUTURA Fila:
    início: PONTEIRO PARA Nó
    fim: PONTEIRO PARA Nó
    tamanho: INTEIRO

FUNÇÃO criarFila() -> PONTEIRO PARA Fila:
    fila = NOVA Fila
    fila->início = NULL
    fila->fim = NULL
    fila->tamanho = 0
    RETORNA fila

FUNÇÃO enfileirar(fila, valor):
    novoNó = NOVO Nó
    novoNó->dado = valor
    novoNó->próximo = NULL

    SE fila->fim É NULL:
        fila->início = fila->fim = novoNó
    SENÃO:
        fila->fim->próximo = novoNó
        fila->fim = novoNó
    FIMSE

    fila->tamanho = fila->tamanho + 1

FUNÇÃO desenfileirar(fila) -> INTEIRO:
    SE fila->início É NULL:
        RETORNA -1
    FIMSE

    nóTemp = fila->início
    valor = nóTemp->dado
    fila->início = nóTemp->próximo

    SE fila->início É NULL:
        fila->fim = NULL
    FIMSE

    LIBERA nóTemp
    fila->tamanho = fila->tamanho - 1
    RETORNA valor
```

```
FUNÇÃO tamanhoFila(fila) -> INTEIRO:
    RETORNA fila->tamanho

FUNÇÃO destruirFila(fila):
    ENQUANTO fila->início NÃO É NULL:
        desenfileirar(fila)
    FIMENQUANTO
    LIBERA fila

FIM
```

Lista

```
INÍCIO

ESTRUTURA Nó:
    dado: INTEIRO
    próximo: PONTEIRO PARA Nó

ESTRUTURA Lista:
    início: PONTEIRO PARA Nó
    tamanho: INTEIRO

FUNÇÃO criarLista() -> PONTEIRO PARA Lista:
    lista = NOVA Lista
    lista->início = NULL
    lista->tamanho = 0
    RETORNA lista

FUNÇÃO inserirNaPosição(lista, valor, posição):
    novoNó = NOVO Nó
    novoNó->dado = valor

    SE posição < 0 OU posição > lista->tamanho:
        RETORNA "Posição inválida"
    FIMSE

    SE posição É 0:
        novoNó->próximo = lista->início
        lista->início = novoNó
    SENÃO:
        nóTemp = lista->início
        PARA i = 1 ATÉ posição - 1:
            nóTemp = nóTemp->próximo
        FIMPARA
        novoNó->próximo = nóTemp->próximo
        nóTemp->próximo = novoNó
    FIMSE
```

```
    lista->tamanho = lista->tamanho + 1

FUNÇÃO removerNaPosição(lista, posição) -> INTEIRO:
    SE posição < 0 OU posição > lista->tamanho - 1:
        RETORNA -1
    FIMSE

    nóTemp = lista->início
    SE posição É 0:
        lista->início = nóTemp->próximo
    SENÃO:
        PARA i = 1 ATÉ posição - 1:
            nóTemp = nóTemp->próximo
        FIMPARA
        nóARemover = nóTemp->próximo
        nóTemp->próximo = nóARemover->próximo
        valor = nóARemover->dado
        LIBERA nóARemover
    FIMSE

    lista->tamanho = lista->tamanho - 1
    RETORNA valor

FUNÇÃO recuperarElemento(lista, posição) -> INTEIRO:
    SE posição < 0 OU posição > lista->tamanho - 1:
        RETORNA -1
    FIMSE

    nóTemp = lista->início
    PARA i = 0 ATÉ posição:
        nóTemp = nóTemp->próximo
    FIMPARA

    RETORNA nóTemp->dado

FUNÇÃO inserirNoInicio(lista, valor):
    inserirNaPosição(lista, valor, 0)

FUNÇÃO inserirNoFinal(lista, valor):
    inserirNaPosição(lista, valor, lista->tamanho)

FUNÇÃO removerNoInicio(lista):
    removerNaPosição(lista, 0)

FUNÇÃO removerNoFinal(lista):
    removerNaPosição(lista, lista->tamanho - 1)
```

```
FUNÇÃO tamanhoLista(lista) -> INTEIRO:
    RETORNA lista->tamanho

FUNÇÃO destruirLista(lista):
    ENQUANTO lista->início NÃO É NULL:
        removerNoInicio(lista)
    FIMENQUANTO
    LIBERA lista

FIM
```