Aula Prática: Filas e Pilhas

Questão 01

Implemente o TAD **Lista** que implementa as funções a seguir. Dica: adapte uma lista implementada em aulas anteriores:

```
/* inicia uma lista */
void Lista_Inicia(Lista *pLista);

/* retorna 1 se a lista for vazia e 0 caso contrario */
int Lista_EhVazia(Lista *pLista);

/* insere o elemento x na posição p da lista */
void Lista_Insere(Lista *pLista, int p, Item x);

/* retira o elemento da posição p da lista e o retorna pelo parâmetro pX
 * a função retorna 0 caso não haja elemento na posição p da lista e 1 caso contrario */
int Lista_Remove(Lista *pLista, int p, Item *pX);

/* retorna o tamanho da lista */
int Lista_Tamanho(Lista *pLista);
```

Utilizando o TAD **Lista**, implemente um TAD **Pilha** que contenha os seguintes métodos:

```
/* inicia uma pilha */
void Pilha_Inicia(Pilha *pPilha);

/* retorna 1 se a pilha for vazia e 0 caso contrario */
int Pilha_EhVazia(Pilha *pPilha);

/* insere o elemento x no topo da pilha */
void Pilha_Push(Pilha *pPilha, Item x);

/* retira o elemento do topo da pilha e o retorna pelo parametro pX
 * a função retorna 0 caso não haja nenhum elemento na pilha e 1 caso contrario */
int Pilha_Pop(Pilha *pPilha, Item *pX);

/* retorna o tamanho da pilha */
int Pilha_Tamanho(Pilha *pPilha);
```

Exemplo de implementação (função Pilha_EhVazia):

```
int Pilha_EhVazia(Pilha *pPilha) {
    return Lista_EhVazia(&pPilha->lista);
}
```

Em seguida, escreva uma calculadora com operações pós-fixadas (como em uma HP). A notação pós-fixada consiste em primeiro inserir os operandos para apenas depois inserir os operadores.

Exemplo: **1 2 - 4 5 + ***
Os valores são empilhados, e sempre que um operador é inserido elas são processadas. O resultado será o seguinte: **-1 4 5 + ***
-1 9 *
-9

Cada linha da entrada contém um número ou um operador. Seu programa deve terminar quando o operador '=' for entrado pelo usuário. Neste momento, seu programa deve imprimir o resultado. Caso não haja resultado (entrada inconsistente), seu programa deve imprimir o texto "Entrada inconsistente." na saída. Seguem exemplos de entrada/saída:

Exemplo de Entrada	**Exemplo de Saída**
1 2 - 4 5 + * =	-9

Exemplo de Entrada	**Exemplo de Saída**
1 2 3 + =	Entrada inconsistente.

Exemplo de Entrada	**Exemplo de Saída**
1 =	1

Exemplo de Entrada	**Exemplo de Saída**
* =	Entrada inconsistente.

Entrega no Moodle:
Certifique-se de que seu programa compila e executa na linha de comando antes de efetuar a entrega. Quando o resultado for correto, entregue um arquivo .ZIP com seu nome, sobrenome e o número da questão (exemplo: tulio-toffolo-01.zip). O .ZIP deve conter a TAD e o programa principal (main.c).

Questão 02

Utilize o TAD **Lista** mencionado no exercício anterior para implementar um TAD **Fila** com as seguintes funções:

```
/* inicia uma fila */
void Fila_Inicia(Fila *pFila);

/* retorna 1 se a fila for vazia e 0 caso contrário */
int Fila_EhVazia(Fila *pFila);

/* insere o elemento x na fila */
void Fila_Enfileira(Fila *pFila, Item x);

/* retira o elemento da frente da fila e o retorna pelo parâmetro pX
 * a função retorna 0 caso não haja elemento na fila e 1 caso contrário */
int Fila_Desenfileira(Fila *pFila, Item *pX);

/* retorna o tamanho da fila */
int Fila_Tamanho(Fila *pFila);
```

Exemplo de implementação (função Fila_Tamanho):

```
int Fila_Tamanho(Fila *pFila) {
    return Lista_Tamanho(&pFila->lista);
}
```

Sabe-se que o TAD **Lista** implementado anteriormente é uma lista encadeada**.** Com esta informação, responda: a implementação de uma fila usando as funções disponíveis na TAD **Lista** acima é eficiente? Justifique sua resposta.

Entrega no Moodle:
Entregue um arquivo .ZIP com seu nome, sobrenome e o número da questão (exemplo: tulio-toffolo-02.zip). O .ZIP deve conter a TAD e um PDF respondendo à questão discursiva.