**1ª Lista Avaliativa**

**O Problema de Josephus.**

     O Problema de Josephus é um bom exemplo de emprego de lista circular. A seguir, descreve-se o problema de Josephus (Tenenbaum, 1989).

     Há um grupo de soldados circundado por uma força inimiga esmagadora. Não há esperanças de vitória sem a chegada de reforços, mas existe apenas um cavalo disponível para escapar. Os soldados entram num acordo para determinar qual deles deverá escapar e trazer ajuda.

* Eles formam um círculo e um número n é sorteado num chapéu.
* Um de seus nomes é sorteado também.
* Começando pelo soldado cujo nome foi sorteado, eles começam a contar ao longo do círculo em sentido horário.
* Quando a contagem alcança n, esse soldado é retirado e a contagem reinicia com o soldado seguinte.
* O processo continua de maneira que, toda vez que n é alcançado, outro soldado sai do círculo.
* Todo soldado retirado do círculo não entra mais na contagem.
* O último soldado que restar deverá montar no cavalo e escapar.

    Considerando um número n, a ordenação dos soldados no círculo e o soldado a partir do qual começa a contagem, o problema é determinar a sequência na qual os soldados são eliminados do círculo e o soldado que escapará.

     A entrada do programa é o número *n* e uma lista de nomes, que será o sequenciamento do círculo em sentido horário, começando pelo soldado a partir do qual a contagem deve ser iniciada

**2ª Lista Avaliativa**

1. Que é um Tipo Abstrato de Dados (TAD) e qual a característica fundamental na sua utilização?

1. Faça a especificação de um sistema de controle de reservas de um clube que aluga quadras poliesportivas usando TAD.

1. O que significa alocação sequencial de memória para um conjunto de elementos?

1. O que significa alocação estática de memória para um conjunto de elementos?

1. Qual a diferença entre alocação sequencial e alocação encadeada?

1. Quais as vantagens de se utilizar alocação encadeada para um conjunto de elementos? Quais as possíveis desvantagens?

1. Projete uma estrutura do tipo lista sequencial ordenada de 20 espaços para abrigar os inimigos ativos de um jogo de computador em ordem crescente de dificuldade (com relação ao número de pontos de vida). Deve ser possível adicionar ou remover um inimigo da lista à medida que este nasce ou é morto pelo jogador, através das funções INSERIR, REMOVER e LISTAR.

1. Desenvolva uma função (com parâmetros) para testar se uma lista F1 tem mais elementos do que uma lista F2 (não se esqueça de mexer nas listas apenas através de seus operadores primitivos).

1. Implemente uma lista em um vetor circular, sem armazenar o número total de elementos (sugestão: nunca deixe que o indicador “fim” alcance o indicador “início”).

1. Projete uma estrutura de lista encadeada para abrigar os IDs dos usuários ativos em uma certa aplicação. A estrutura de um nó da lista é a seguinte: Node( int userId; Node pProx;) Ao final do projeto, seu programa deve rodar as seguintes funções:

* Escreva a função inserir, que cria e insere um nó na primeira posição da lista encadeada. Para simplificar o projeto, não se preocupe em casos de inserção de um ID repetido.
* Escreva a função printLista, que mostra na tela todos os IDs dos usuários ativos num dado momento.
* Escreva a função remover, que exclui um certo ID de usuário da lista e mostra na tela o ID removido.

**3ª Lista Avaliativa**

Desenvolva através de uma lista duplamente encadeada as funções que representem um player (dispositivo MP3 para reprodução de músicas)

***Player***é um reprodutor multimídia que nos permite controlar a reprodução de um áudio ou vídeo. Um *player* pode apresentar diversas funcionalidades, vamos conhecer as principais:

|  |  |  |
| --- | --- | --- |
| **Opção** | **Função** | **Descrição** |
| T | **Play (Tocar)** | |  |  | | --- | --- | |  | Reproduzir o conteúdo selecionado. | |
| S | **Stop (Sair)** | |  |  | | --- | --- | |  | Parar completamente a reprodução. | |
| P | **Pause (Pausar)** | |  |  | | --- | --- | |  | Congela a reprodução temporariamente. Se em seguida acionar a função Play, a reprodução começa Do último ponto em que foi Acionada a pausa. | |
| A | **Forward (Avançar) Ir para a música seguinte** | |  |  | | --- | --- | |  | Esse comando nos possibilita avançar a próxima música da lista, mas nunca depois da ultima | |
| V | **Rewind (Voltar)**  **Ir para a música anterior** | |  |  | | --- | --- | |  | Esse comando nos possibilita voltar a música anterior da lista, mas nunca antes da primeira | |

Bom trabalho!!