Trabalho Prático 01 – AEDS 1 – Trios

Professora: Thais R. M. Braga Silva

Valor: 12 pontos

Data de Entrega: 21/12/2021 (até 08:00)

Forma de Entrega: PVANet Moodle (formato .zip ou .tar.gz)

Processos podem ser definidos como programas em execução. Em sistemas operacionais, cada processo pode ser visto como um módulo separado e carregável, composto por recursos tais como código de máquina executável, memória, descritores, atributos de segurança e contexto de estado do processador, por exemplo.

Os sistemas operacionais são responsáveis por gerenciar a utilização dos recursos de uma arquitetura computacional (processador, memória, dispositivos de entrada e saída, por exemplo) pelos processos que se encontram ativos (em execução). Para tanto, eles utilizam o conceito de lista linear. Os processos são colocados na lista a medida que fazem solicitação por recursos computacionais. O sistema operacional gerencia a lista, adicionando processos quando ocorrem novas solicitações e removendo-os quando o acesso é liberado.

Neste trabalho, você implementará um Tipo Abstrato de Dados (TAD) Lista Linear, para que um sistema operacional faça gerência de processos ativos. Em seguida, você implementará um programa de teste, que usa o TAD para criar uma lista de processos, realizando inserções e remoções desta lista. Por fim, uma avaliação simples de desempenho deverá ser realizada de forma que possamos observar a eficiência do TAD considerando diferentes tamanho de lista.

TIPO ABSTRATO DE DADOS PROCESSO

Para começar, será necessário que você implemente o TAD Processo. Cada processo deverá ser representado por PID (identificador do processo, gerado aleatoriamente), Horário da criação do processo, Prioridade (valor de 1 a 5, gerado aleatoriamente). As operações disponibilizadas devem ser aquelas que permitem acessar os dados do processo para leitura (operação Get) e alteração (operação Set). Exemplo: GetPID() e SetPID(int PID).

TIPO ABSTRATO DE DADOS LISTA DE PROCESSOS – CURSORES [Ziviani, Nivio, 2007 – Capítulo 2, Exercício 12, Página 108]

A finalidade deste TAD é gerenciar uma área interna de memória de forma que o maior e o menor elemento possam ser removidos com custo constante. A estrutura de dados que deve ser utilizada é uma lista linear duplamente encadeada implementada por cursores. Os cursores são números inteiros que representam posições em um arranjo e são utilizados para simular os apontadores da implementação tradicional das listas lineares duplamente encadeadas. A utilização de cursores evita a alocação e a liberação dinâmica de itens de memória, sendo mais eficiente em aplicações muito dinâmicas em que o número máximo de itens é conhecido.

Este TAD possui as seguintes operações:

a) Criar uma área de memória interna vazia. Aloca um vetor de células de tamanho N. Vamos utilizar alocação dinâmica deste vetor, devido aos altos valores de N que adotaremos;

b) Obter o número de células ocupadas na área de memória. Retorna o número de posições ocupadas no vetor;

c) Inserir um item de dado na área interna de memória, mantendo os itens ordenados;

d) Retirar o primeiro item da área de memória;

e) Retirar o último item da área de memória;

f) Imprimir o conteúdo da área de memória. Somente imprime o conteúdo das células ocupadas.

A retirada de um item de uma lista duplamente encadeada implementada por cursores pode ser realizada a um custo constante, desde que se conheça previamente o endereço do item na lista. Ao manter a lista ordenada, os elementos de menor e maior chave estão na primeira e última posição, respectivamente.

Os itens de dados (no caso deste trabalho, processos) da lista linear duplamente encadeada implementada por cursores são armazenados em um vetor do tipo *Celula*. Cada entrada do vetor contém uma estrutura (uma célula) que armazena um item de dado (um processo), um cursor que aponta para o item que sucede aquela entrada (*prox*) e um cursor que aponta para o item que antecede aquela entrada (*ant*). No TAD Lista de Processos, além do vetor, são representados dois cursores, *primeiro* e *último,* que apontam para o primeiro e o último item da lista, respectivamente. Para facilitar o controle de quando a lista se encontra cheia ou vazia, utilize o campo *numCelOcupadas*, que indica quantas células da lista estão ocupadas.

Os itens armazenados no TAD Lista de Processos precisam ser mantidos ordenados. Você pode utilizar qualquer algoritmo de ordenação que desejar. Para fins de comparação entre os processos, de forma a ordená-los, utilize o PID como chave de ordenação. Isto significa que, na lista, os processos aparecerão por ordem de PID.

Somente o que foi apresentado até agora não é suficiente para implementar o TAD Lista de Processos. Isto porque, para incluir um novo item de dado na lista, é necessário haver células disponíveis a fim de que a inserção seja realizada. Assim, para gerenciar a lista de células disponíveis em determinado instante, basta incluir um cursor na representação da estrutura de dados (ou seja, na estrutura do próprio TAD Lista de Processos), o qual irá apontar para a primeira célula disponível. Tal cursor foi denominado *celulasDisp*. As demais células disponíveis podem ser encontradas através do encadeamento entre os cursores das mesmas, isto é, cada célula disponível, possui em seu campo *prox* o endereço da próxima célula disponível no vetor. A última célula disponível possui o valor -1 no seu campo *prox*. Da mesma forma, cada célula ocupada possui o endereço da próxima célula ocupada em seu campo *prox*. A última célula ocupada possui o valor -1 em seu campo *prox*. O valor -1 também deve ser utilizado no campo *ant* da primeira célula ocupada e de todas as células disponíveis.

Dessa forma, antes de incluir um novo item de dado, remove-se a primeira célula da lista de disponíveis (apontada por *celulasDisp*) e a insere ordenadamente na lista linear duplamente encadeada que armazena os itens de dados do TAD Lista de Processos. Já ao remover um item de dados, a célula que continha tal item deve ser inserida na lista de disponíveis. Para que a

inserção e remoção da lista de disponíveis seja realizada a um custo constante, elas devem ser realizadas na posição apontada por *celulasDisp*.

SISTEMA DE TESTE

Para testar a implementação dos seu TAD, você deverá implementar um programa principal que simula a gerência de uma lista de processos feita pelo sistema operacional. Para tanto, este programa deverá possuir as seguintes funcionalidades:

- Inicialização da lista de processos;

- Inserção de novos processos na lista;

- Remoção de processos da lista (primeiro);

- Impressão da lista (na saída padrão);

O programa deverá permitir a criação interativa de uma lista, bem como o carregamento de uma nova lista a partir de um arquivo (ver detalhes sobre essa última opção abaixo). O tamanho da lista é, portanto, definido pelo usuário. Em seguida, deverá permitir que o usuário indique o número de inserções ou remoções que deseja realizar sobre a lista (ou lê essas informações do arquivo), efetuando tais operações. Por fim, o seu sistema de testes deverá medir qual o tempo gasto para a realização de cada teste.

ARQUIVO DE ENTRADA

O arquivo de entrada deverá possuir o seguinte formato:

N *// Tamanho do vetor para cursores*

NLO *// Número de linhas de operações especificadas abaixo*

Op Qt *// Op = 0 para inserção e 1 para remoção início*

// Qt = quantidade de vezes que a operação será realizada

ARQUIVO DE SAÍDA

O arquivo de saída deverá ser uma tabela constando o número do teste e o tempo total de execução encontrado para sua realização.

NUM_TESTE <tempo> *// Número do teste, valor para TAD cursores*

TESTES

Teste (01) Usar TAD cursores, com vetor de tamanho 100.000 e fazer 100.000 inserções;

Teste (02) Usar TAD cursores, com vetor de tamanho 100.000 e fazer 100.000 remoções;

Testes (03) e (04) – Mesmo que testes (01) e (02), porém para 200.000

Testes (05) e (06) – Mesmo que testes (01) e (02), porém para 300.000

Testes (07) e (08) – Mesmo que testes (01) e (02), porém para 400.000

Testes (09) e (10) – Mesmo que testes (01) e (02), porém para 500.000

Testes (11) e (12) – Mesmo que testes (01) e (02), porém para 600.000

Em particular, atente para:

- O programa deve ser organizado em módulos, conforme estudado em sala de aula. O módulo do programa principal deve estar separado dos módulos que compõem os TADs.
- O programa deve estar bem indentado e comentado
- Caso apareçam números fixos no código, estes devem ser definidos como constantes
- **Trabalhos copiados serão penalizados.**

ATENÇÃO: Soluções que não correspondam à implementação de Tipos Abstratos de Dados serão duramente penalizadas por não atenderem à especificação.

O que deve ser entregue:

- Todo código fonte produzido, incluindo os arquivos de cabeçalho
- Uma pequena porém completa documentação, descrevendo o objetivo do trabalho, o projeto do sistema implementado, as principais decisões de projeto, os módulos desenvolvidos, os métodos implementados e a conclusão. O formato para a entrega da documentação é pdf
- Fazer um zip ou tar.gz de todos os arquivos, nomeá-lo com o nome e número de matrícula do trio, bem como número do TP, e submetê-lo apenas uma vez pelo PVANet Moodle (NÃO UTILIZAR COMPACTAÇÃO RAR!).

Como será a avaliação:

- Entrevista
- Avaliação do professor