UNIVERSIDADE FEDERAL DE MATO GROSSO

CURSO: Ciência da computação
DISCIPLINA: Estrutura de dados
PERÍODO: 2022/2
PROFESSOR: Ivairton Monteiro Santos

Nome: Herrison Batista de Meneses Junior e Gabriel Gomes Marchesan
Matrícula: 202111722014, 202111722011

Relatório

Relatório referente ao
trabalho 4 sobre a análise de
lista estática e dinâmica

Requisitado pelo professor
Ivairton Monteiro Santos

Conclusões solicitadas no relatório

Lista estática com capacidade de 10.000.

Na lista estática foram implementadas 7 funções no total. A primeira função é para inicializar a lista(inicializaLista) que tem como parâmetro a própria lista e o tamanho a ser alocado, que no caso é 10.000, ela aloca memória para o vetor e inicializa a variável de controle da quantidade de elementos na lista.

A função listaCheia tem como parâmetro o ponteiro da lista, verificando se o número de Elementos da função (inicializaLista) é menor que o tamanho da lista.
A função listaVazia tem como parâmetro o ponteiro da lista, verificando se o número de elementos é maior que 0, assim concluindo se está vazia.

A função insereFim verifica se a lista está cheia através da função listaCheia , se for Positivo ela retorna falso, se não ela coloca o valor no fim da lista, tem como parâmetro a lista e o elemento a ser inserido no fim da lista.

A função insereInicio verifica se a lista está cheia utilizando a função listaCheia, ela utiliza uma variável de controle para fazer um loop que coloca todos os elementos da lista para trás e em seguida insere no inicio.

A função inserePos verifica se a lista está cheia utilizando a função listaCheia, ela faz um teste verificando se a posição solicitada é maior ou igual a zero, se está no tamanho disponível, e se aquela posição está livre, em caso positivo ela utiliza uma variável de controle para fazer um loop que empurra os elementos para inserir na posição solicitada, em caso negativo ela coloca o valor na posição solicitada, tem como parâmetro a lista, o valor e a posição.

A função imprimeLista tem como parâmetro lista, ela varre o vetor e imprime a Quantidade de elementos definida e imprime todas as posições dos vetores.

Listas encadeadas

Na lista com encadeamento simples foram implementadas 5 funções. A primeira função é inicializarLista que tem como parâmetro lista, o ponteiro apontado para o inicio recebe NULL e tendo seu tamanho definido como 0.

Na função insereInicioLista, tem como parâmetro a lista e o número a ser inserido no Inicio, ela cria um novo nó e verifica se ele foi criado com êxito, em caso positivo o valor do novo nó passa a ser o número inserido no parâmetro, e novo nó próximo recebe o começo da lista, o começo da lista recebe novo nó e aumenta a lista.

Na função insereFimLista, que tem como parâmetro o número a ser inserido e a lista, cria um novo nó, um auxiliar e verifica se teve êxito, em caso positivo, o valor do novo nó passa a ser o valor inserido no parâmetro da função, e novo nó próximo recebe NULL já que é o ultimo elemento da lista. Depois verifica se o inicio da lista está em NULL, que no caso é uma verificação se está sendo o primeiro elemento e o ultimo, em caso positivo o começo da lista passa a apontar para o novo nó, em caso negativo, utilizamos o auxiliar para apontar para o inicio da lista, e enquanto o auxiliar próximo existir ele irá apontar para novo e o auxiliar para o auxiliar próximo e no fim aumenta

o tamanho da lista.

Na função inserePosicaoLista, que tem como parâmetro a lista, o valor e a posição, ela cria o novo nó e verifica se ele foi criado, logo após o novo nó recebe como valor o número inserido no parâmetro da função, em seguida ela verifica se o inicio da lista está em NULL, caso der positivo o novo nó próximo aponta para NULL e lista inicio aponta para o novo nó. Em caso contrario utiliza o auxiliar para apontar para o inicio da lista e faz um loop utilizando o valor de auxiliar em comparação com a posição solicitada, enquanto não for a posição e enquanto existir auxiliar próximo, o auxiliar vai receber auxiliar próximo, novo nó próximo vai receber auxiliar próximo e auxiliar próximo recebe novo nó.

A função imprimeLista tem como parâmetro lista, ela varre o vetor e imprime a quantidade de elementos definida pelo usuário e imprime todas as posições dos vetores.

Na lista com encadeamento duplo foram utilizadas a mesma lógica em relação a com encadeamento simples, com a exceção do uso do auxiliar, ao envez de utilizar uma variável que assumiria o papel de auxiliar, é criado mais um ponteiro para cada nó, então cada nó possui dois ponteiros, um que aponta para o próximo nó e o outro para o nó anterior.

Conclusão final

A lista estática é definitivamente mais eficaz que a dinâmica em questão de velocidade, porém tem um tamanho máximo predefinido.

A lista encadeada simples é mais eficaz em relação ao consumo da memoria e também é menos complexa em comparação a duplamente encadeada, porém só da para percorrer por um lado já que os nós apontam somente para o próximo nó.

A lista duplamente encadeada pode ser percorrida pelos dois lados já que você pode começar pelo inicio ou pelo fim e percorrer nó por nó, podendo assim escolher o caminho mais eficiente, porém possui um consumo de memoria relativamente alto.

Pelo que consegui ver na internet, um jeito de medir a memoria utilizada é através de uma extensão chamada valgrind, porém só está disponível em Linux. No Windows achei algumas opções, utilizei um programa chamado Dr memory que deu como Resultado:

Estática : em torno 4419 bytes
Encadeada: em torno 4559 bytes
Duplamente encadeada: em torno 4857 bytes

Tabela final

Estrutura	Qtd Ins Inicio	Qtd Ins Fim	Qtd Ins Posição	Menor tempo	Maior tempo	Tempo médio
Lista estática com capacidade de 10.000	3.333	3.334	3.333	26ms	31ms	28,7ms
Lista com encadeamento simples	3.333	3.334	3.333	30,1ms	56,5ms	43,5ms
Lista dinâmica duplamente encadeada	3.333	3.334	3.333	51ms	57ms	52,4ms