Tutoria ED2 - Semana 1

Matheus Peixoto Ribeiro Vieira - 22.1.4104

1 - FIFO: Utiliza uma fila padrão, assim, a primeira página a ser carregada também será a primeira a ser excluída, quando todos os espaços forem preenchidos. Todavia há a desvantagem de quando a primeira página se torna a mais requisitada, pois ela será removida e, de tempos em tempos, ela deverá ser recuperada novamente.

LRU: A política de remoção do mais recentemente utilizada implementa uma pilha onde sempre coloca a página mais utilizada no final, e quando se torna necessário remover alguma, a primeira da fila será excluída. A sua desvantagem é quando uma página é muito utilizada, fazendo com que tenha que ir até o fim da fila para buscar a mesma

LFU: Implementa um contador para saber quantas vezes uma página foi utilizada e remover a que tiver o menor valor. A desvantagem é que no momento em que uma página é carregada na memória principal ela se torna a mais suscetível a ser removida na próxima remoção.

2 - Com o arquivo tendo suas chaves ordenadas, o programa irá ler as informações de forma sequencial e dividi-las em blocos chamados de páginas e cada uma terá a sua quantidade máxima de itens que consegue comportar.

Dessa forma, para localizar um dado, será verificado a chave da página e a desejada. Caso a chave desejada seja maior do que a vista na primeira página, passa para a próxima, até que ela seja encontrada ou que seja identificado uma maior.

Neste último caso, o item estará na página anterior, pois a chave desejada é menor. Assim ocorrerá uma pesquisa dentro da página, podendo ser implementado uma pesquisa sequencial dentro dela ou outros métodos de pesquisa em memória principal.

3 - Além do item, torna-se necessário adicionar no arquivo externo os ponteiros para os próximos itens filhos, um à esquerda e outro à direita.

Um dos métodos para organizar os nós é simplesmente salvar todos em forma sequencial, somente preocupando-se em salvar sequencialmente as informações.

Outra maneira de salvar os dados é a partir do agrupamentos dos nós filhos com o nó pai, agrupando como se estivessem juntos na paginação.

4 - Para construir uma árvore B, é necessário definir um número arbitrário M que será o responsável por ditar o tamanho de todos os nós.

Com o M definido, a página raiz terá um tamanho que varia de 1 até 2m, enquanto as demais páginas, ou nós, irão variar com o tamanho m até 2m e terão pelo menos m+1 apontadores até 2m+1 apontadores para os filhos.

Ademais, é necessário que todos os itens dentro de um nó tenham as suas chaves ordenadas, seguindo a mesma ordem para todos os filhos.

5 -

C/C++

```c
#define M 4
typedef struct {
    int chave;
}Registro;

typedef struct No * nos;
typedef struct{
    nos filhos [2*M + 1];
    Registro items [2*M];
}No;

void iniciarArvore(No * arvore){
    arvore = NULL;
}
```

```
Registro * procura(No * arvore, int chavePesquisa){
    if (arvore == NULL) return NULL;
    int i;
        for  (i  =  0;  i  <  2*M  &&  arvore->items[i].chave  <
chavePesquisa; i++)
        if(arvore->items[i].chave == chavePesquisa)
            return &(arvore->items[0]);

    return procura(&(arvore->items[i]), chavePesquisa);

}
```