Apresentação: Arvore Splay

Alunos: Mateus Miranda Guimarães; Michel Oliveira da Silva; Pedro Henrique Gomides Moraes

O que é:

A árvore splay é uma árvore binária de busca que faz autoajustes, o que significa que essa estrutura de árvore é ajustada dinamicamente com base nos elementos inseridos e pesquisados. Em outras palavras, a árvore se organiza automaticamente para que os elementos mais acessados se movam para a raiz ou próximos dela. As vantagens da árvore estão no fato de que pela natureza da árvore Splay, ela acaba reduzindo o tempo de busca dos elementos mais acessados, isso faz com que a árvore splay seja útil para as aplicações que precisam de dados rápidos e dinâmicos. No entanto, a árvore splay se mostra pouco eficiente quando tratamos de aplicações que terão garantia de “Pior cenário possível”, visto que ela se foca em deixar de fácil acesso aqueles dados mais acessados, o que faz com os que menos acessados acabem negligenciados.

Funcionamento da árvore Splay:

A árvore splay funciona de forma a otimizar o processo de pesquisa de um elemento x em seus nós. Durante o processo de pesquisa ela percorre todo o caminho até encontrar o elemento x, quando encontrado realiza o movimento de splay para que este seja levado para a raiz principal da árvore. Durante este processo podem ocorrer inúmeras rotações, para que o elemento x chegue até a raiz. As rotações podem ser classificadas como zig para a direita e zag para a esquerda, nas rotações simples o movimento de splay pode ser interpretado como zig se o elemento x estiver mais à esquerda na árvore, e zag se estiver mais à direita. Para as rotações duplas, para a direita caso o elemento x estiver mais à esquerda da árvore pode ser interpretado como zig zig, já se o elemento estiver mais à direita as rotações duplas a esquerda serão zag zag, e se durante o processo de rotação for necessário rotacionar para a direita e esquerda o splay será denominado zig zag e vice-versa.

Explicação do Código Splay:

A função Splay é responsável por rearranjar a árvore de acordo com as pesquisas do usuário, a cada pesquisa por item, a função é chamada e com isso pega o nó encontrado e o move para raiz realizando as rotações necessárias.
Funcionamento:

1° passo verificasse se o nó encontrado é a raiz da árvore, caso for retorna a própria raiz senão continua a função;

2° passo verificasse se o nó está na sub árvore a esquerda, caso estiver verifica em qual posição a esquerda está, com isso, se o valor a esquerda for maior que o valor pesquisado ele ira realizar o zig-zig usando uma função recursiva. Caso contrário ira realizar a função zig-zag da mesma forma.

3° passo verificasse se o nó está na sub árvore a direita, caso estiver verifica em qual posição a direita está, com isso, se o valor a direita for maior que o valor pesquisado ele ira realizar o zag-zig usando uma função recursiva. Caso contrário ira realizar a função zag-zag da mesma forma.

```c
tipoNoh *splay(tipoNoh *raiz, int valor)
{
    // Caso base
    if (raiz == NULL || raiz->item.valor == valor)
        return raiz;

    // O valor está na sub arvore esquerda
    if (raiz->item.valor > valor)
    {
        //O valor nao esta nessa sub arvore
        if (raiz->esquerda == NULL)
            return raiz;

        if (raiz->esquerda->item.valor > valor) // Zig-Zig (esquerda esquerda)
        {
            raiz->esquerda->esquerda = splay(raiz->esquerda->esquerda, valor);
            raiz = rotacaoDireita(raiz);
        }
        else if (raiz->esquerda->item.valor < valor) // Zig-Zag (esquerda direita)
        {

            raiz->esquerda->direita = splay(raiz->esquerda->direita, valor);

            if (raiz->esquerda->direita != NULL)
                raiz->esquerda = rotacaoEsquerda(raiz->esquerda);
        }

        //Condicional utilizando o operador ternario (é tipo um If e Else)
        return (raiz->esquerda == NULL)? raiz: rotacaoDireita(raiz);

    }
    else if (raiz->item.valor < valor) // Valor está na subárvore direita
    {
        //Nao esta nessa sub arvore
        if (raiz->direita == NULL)
            return raiz;

        // Zag-Zig (Direita esquerda)
        if (raiz->direita->item.valor > valor)
        {
            raiz->direita->esquerda = splay(raiz->direita->esquerda, valor);

            if (raiz->direita->esquerda != NULL)
                raiz->direita = rotacaoDireita(raiz->direita);
        }
        else if (raiz->direita->item.valor < valor)// Zag-Zag (direita direita)
        {
            raiz->direita->direita = splay(raiz->direita->direita, valor);
            raiz = rotacaoEsquerda(raiz);
        }

        //Condicional utilizando o operador ternario (é tipo um If e Else)
        return (raiz->direita == NULL)? raiz: rotacaoEsquerda(raiz);

    }
}
```