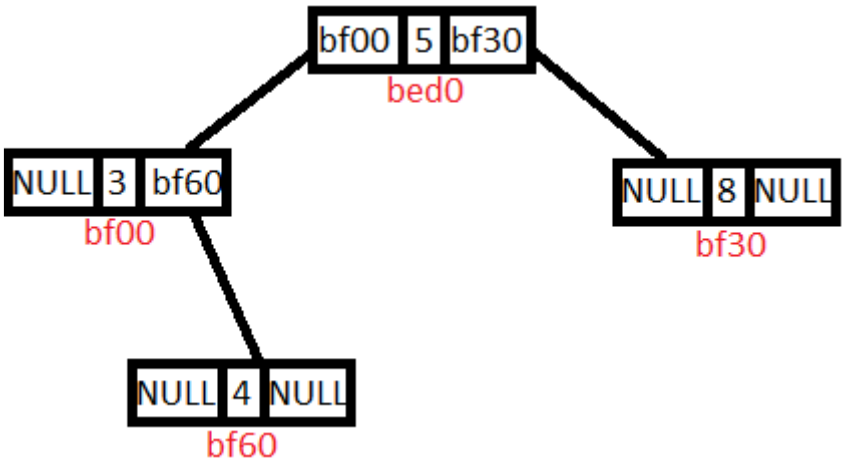
Atividade avaliativa – Estrutura de Dados II

Matheus de Oliveira Cortez Cervantes

1)- Apresente os empilhamentos produzidos nas funções recursivas (Pré-ordem, em-ordem e pós-ordem);



a) Pré-ordem:

```
void Arvore::preordem (noArvore *raiz) {
1   if(raiz != NULL) {
2       cout << raiz->dado << " ";
3       preordem(raiz->esquerda);
4       preordem(raiz->direita);
    }
}
```

4	4°Desempilhamento
4	2°Desempilhamento
4	1°Desempilhamento
4	3°Desempilhamento

b) Em-ordem:

```
void Arvore::emordem (noArvore *raiz) {
1   if(raiz != NULL) {
2       emordem(raiz->esquerda);
3       cout << raiz->dado << " ";
4       emordem(raiz->direita);
    }
}
```

3,4	4°Desempilhamento
3,4	2°Desempilhamento
3,4	1°Desempilhamento
3,4	3°Desempilhamento

c) Pós-ordem:

```
void Arvore::posordem (noArvore *raiz) {
1   if(raiz != NULL) {
2       posordem(raiz->esquerda);
3       posordem(raiz->direita);
4       cout << raiz->dado << " ";
    }
}
```

bf30	3	6° Desempilhamento
	4	7° Desempilhamento
bf60	3	2° Desempilhamento
	4	3° Desempilhamento
bf00	3	1° Desempilhamento
	4	4° Desempilhamento
bed0	3	5° Desempilhamento
	4	8° Desempilhamento

2)- Implemento um método recursivo que apresente o maior elemento de uma árvore binária.

```
void Arvore:: maiorelemento (noArvore *raiz) {
    if (raiz->direita == NULL)
        cout << raiz->dado;
    else
        maiorelemento(raiz->direita);
}
```

3)- Implemente um método recursivo que some todos os elementos de uma árvore binária.

```
int Arvore::somaelementos (noArvore *raiz) {
    static int soma=0;
    if(raiz != NULL) {
        somaelementos(raiz->esquerda);
        somaelementos(raiz->direita);
        soma = (soma + raiz->dado);
    }
    return soma;
}
```