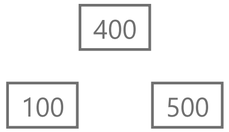
Sequência: 100, 500, 400, 300, 600, 450

Inserção: 100
- Raiz nula, é inserido como raiz

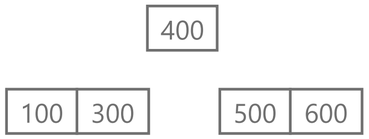
Inserção: 500
- Maior que a raiz esquerda, e tem vaga, adicionado a direita na raiz direita.

Inserção: 400
- Maior que a raiz esquerda e raiz direita, adicionado, estoura e balanceado, elemento mediano sobe.

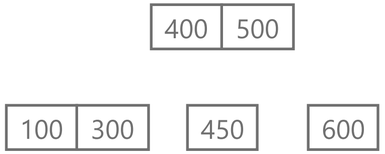


Inserção: 300
- Menor que a raiz, adicionado a raiz a esquerda;
- Maior que 100, adicionado a direita.

Inserção: 600
- Maior que a raiz esquerda e direita, adicionado a direita;
- Maior que 500, adicionado a direita.



Inserção: 600
- Maior que a raiz, estoura e sobe o elemento mediano.
- 450 passa a ser o filho do centro da raiz.



2) Impressão em ordem:

```
void showNumber(int n){
    if(n != 0) printf("%d\n", n);
}

void inOrder(Tree *root) {
    if (root != NULL) {
        inOrder(root->left);
        showNumber(root->left);
        inOrder(root->center);
        showNumber(root->right);
        inOrder(root->right);
    }
}
```

3) Busca:

```
int search(Arv23 *root, int x, int result){
    if(root != NULL){
        if(x == root->chaveEsq) result = 1;
        else if(x == root->chaveDir) result = 1;
        else if(x < root->chaveEsq) result = search(root->esq, x, result);
        else if (x > root->chaveEsq && x < root->chaveDir) result = search(root->centro, x, result);
        else result = search(root->dir, x, result);
    }
    return result;

}
```

4)

a) Nova Sequência: 500, 100, 400, 300, 600, 450

b) Se inserirmos o 200. Ela estourará o nó 100|300 e subirá para a raiz. Uma vez na raiz, o 200, estourará o nó da raiz e assim subirá pra cima, virando a raiz da árvore. O desenho da árvore ficará identica ao da árvore binária de busca.