```
/*
 * 3) Implemente os procedimentos de rotação a direita e rotação a esquerda considerando
 * o contexto das árvores AVL. Assuma que é dado na entrada o ponteiro de ponteiro do nó a ser rotacionado.
 */


int FB(TNoAVL *p)
{
   if(!p) return 0;
   return p->fb;
}

 //Assumindo que o ponteiro de ponteiro é de uma AVL
int rotacaoDir(TNoAVL **pp)
{

   //rotação:
   TnoAVL **esq = &((*pp)->esq);
   TNoAVL *dir = *(*esq)->dir;
   //pai vira filo direito de seu filho esquerdo
   (*esq)->dir = *pp;
   //filho direito do antigo filho agora é o filho esquerdo de do antigo filho
   (*pp)->esq = dir;
   //filho se tornou o novo pai
   *pp = *esq;

   //Balanceando
   /*
    * FB do novo filho direito (ex raiz passado para a função) da nova raiz da sub árvore (ex filho esquerdo da antiga raiz)
    * O filho esquerdo da nova raiz não é alterado na rotação direita
    */
   (*pp)->dir->fb = FB((*pp)->dir->esq->fb) + FB((*pp)->dir->dir->fb);

   //atualizando FB da nova raiz
   (*pp)->fb = FB((*pp)->esq->fb) + FB((*pp)->dir->fb);

   //retorna o novo fator de balancemanto da nova raiz
   return (*pp)->fb;
}

//Assumindo que o ponteiro de ponteiro é de uma AVL
int rotacaoEsq(TNoAVL **pp)
{

   //rotação:
   TnoAVL **dir = &((*pp)->dir);
   TNoAVL *esq = *(*dir)->esq;
```

```
    //pai vira filo esquerdo de seu filho direiro
    (*dir)->esq = *pp;
    //filho esquerdo do antigo filho agora é o filho direito de antigo pai
    (*pp)->dir = esq;
    //filho se tornou o novo pai
    *pp = *dir;

    //Balanceando
    /*
     * FB do novo filho esquerdo (ex raiz passado para a função) da nova raiz da sub árvore (ex filho direito da antiga raiz)
     * O filho direito da nova raiz não é alterado na rotação esquerda
     */
    (*pp)->esq->fb = FB((*pp)->esq->esq->fb) + FB((*pp)->esq->dir->fb);

    //atualizando FB da nova raiz
    (*pp)->fb = FB((*pp)->esq->fb) + FB((*pp)->dir->fb);

    //retorna o novo fator de balancemanto da nova raiz
    return (*pp)->fb;
}

/*
 * 4) À semelhança da questão 3, implemente cada um dos procedimentos de rotação dupla da AVL e da árvore splay.
 */

//Serão utilizadas as funções: FB(), retacaoEsq() e rotacaoDir() implementados no exercício 3) dessa mesma lista

int rotaçãoEsqDir(TNoAVL **pp)
{
    //realizamos a primeira rotação e atualizamos o FB do pai do nó dado como parâmetro para a rotação
    (*pp)->fb = FB(rotacaoEsq(&((*pp)->esq))) + FB((*pp)->dir);
    return rotaçãoDir(pp);
}
int rotaçãoDirEsq(TNoAVL **pp)
{
    //realizamos a primeira rotação e atualizamos o FB do pai do nó dado como parâmetro para a rotação
    (*pp)->fb = FB(rotacaoDir(&((*pp)->dir))) + FB((*pp)->esq);
    return rotaçãoEsq(pp);
}
```