There are the following JAVA classes implementing the multi-frontal direct solver algorithm for finite difference problem:

- *A.java*
- *A1.java*
- *A2.java*
- *AN.java*
- *Aroot.java*
- *BS.java*
- *E2.java*
- *Eroot.java*
- *Executor.java*
- *Main.java*
- *P1.java*
- *P2.java*
- *P3.java*
- *Production.java*
- *Vertex.java*

Let us focus on the classes *A2.java, E2.java, Aroot.java,Eroot.java,BS.java*.

The class *A2* merges two matrices from son nodes into one matrix at parent node. This is illustrated in Figure below

**Please implement the production *A2***

Hint:

The merging of right hand sides is already implemented there:

```
1   class A2 extends Production {
2     A2(Vertex Vert,CyclicBarrier Barrier){
3       super(Vert,Barrier);
4     }
5   Vertex apply(Vertex T) {
```

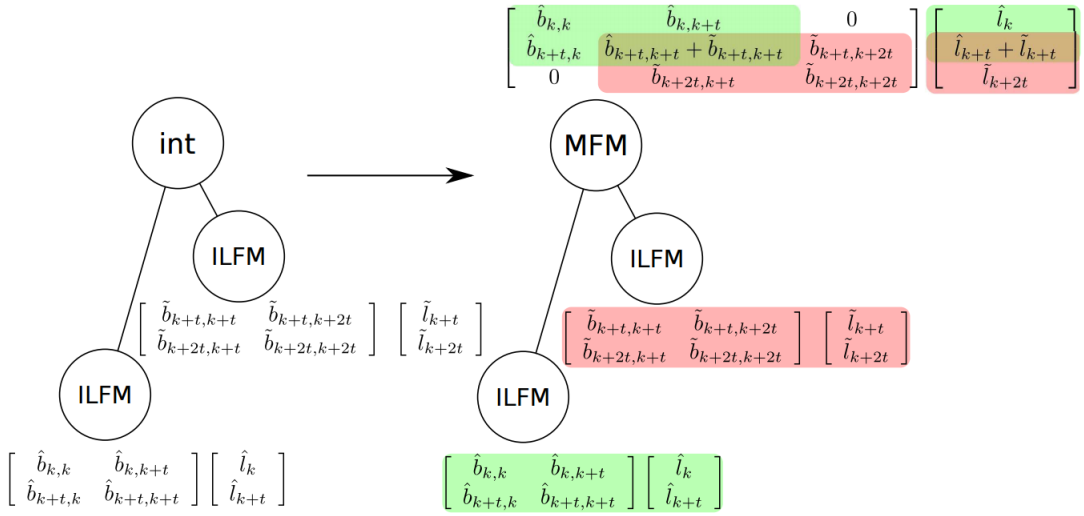


Figure 1: Production *A2* merging two matrices at parent level.

```
6       System.out.println(''A2'');
8       T.m_b[0] = T.m_left.m_b[0];
7       T.m_b[1] = T.m_left.m_b[2] + T.m_right.m_b[1];
9       T.m_b[2] = T.m_right.m_b[2];
11      ...
11      return T;
12      }
13    }
```

For more convenient implementation of the forward elimination, we replace the first and second row and column in the matrix:

**Please reimplement the production *A2* including exchange of rows and columns**

Hint:

The merging of right hand sides including the exchange is already implemented there:

```
1    class A2 extends Production {
2      A2(Vertex Vert,CyclicBarrier Barrier){
3        super(Vert,Barrier);
4      }
```

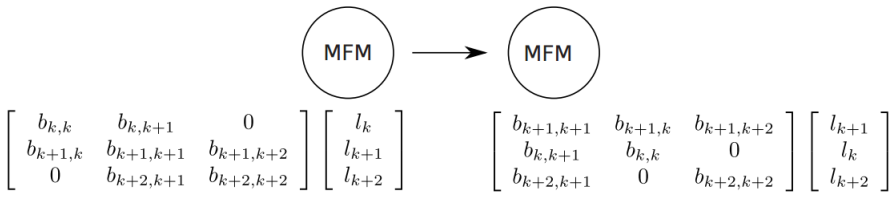


Figure 2: Production *A2* merging two matrices at parent level.

```
5   Vertex apply(Vertex T) {
6     System.out.println(''A2'');
7     T.m_b[0] = T.m_left.m_b[2] + T.m_right.m_b[1];
8     T.m_b[1] = T.m_left.m_b[1];
9     T.m_b[2] = T.m_right.m_b[2];
11    ...
11    return T;
12    }
13  }
```