Armstrong's Axioms Format

Proof using Armstrong's axioms can only use the following rules: *given*, *reflexivity*, *augmentation* and *transitivity*. Proof using *extended* Armstrong's axioms can only use the following rules: *given*, *reflexivity*, *augmentation*, *transitivity*, *union* and *decomposition*. The format of each line of the proof is:

`FD [ Reason ]`

where `FD` is a functional dependency with the format as above (*e.g.*, `A -> B`) and `Reason` is as follows:

1. Given: `Given`
2. Reflexivity: `Reflexivity`
3. Augmentation: `Augmentation ( line ) with attr`
4. Transitivity: `Transitivity ( line ) and ( line )`
5. Union: `Union ( line , line , line , ⋯)`
6. Decomposition: `Decomposition of ( line )`
    - `line` is the line number according to your proof.
    - `attr` is an attribute (single letter uppercase) or a set of attributes (multi-letter uppercase).

Notes:
- To use *given*, the FD must be part of the assumption given in the question.
- To use *reflexivity*, the RHS of the FD must be a subset of the LHS of the FD.
- *augmentation* may add an already existing attribute.
- In *transitivity* (*i.e.*, if `a -> b` and `b -> c` then `a -> c`), the first line must be matched to `a -> b` and the second line must be matched to `b -> c`.
- *union* requires a minimum of 2 lines and has no upper limit on the maximum number of lines. However, it should not have duplicate line number.
- Attributes may be duplicated.
- If other rules are added in the question, the usage will be explained.

Example: *note that the proper alignment of arrows and/or reasons are not necessary*

Example of correct usage
```
A  -> B  [Given]
B  -> C  [Given]
AB -> AB [Reflexivity]
AB -> BB [Augmentation (1) with B]
A  -> C  [Transitivity (1) and (2)]
AB -> A  [Decomposition of (3)]
A  -> BC [Union (1,5)]
```

Example of wrong usage (*except for first two*)
```
A  -> B  [Given]
B  -> CC [Given] (note: duplicate is allowed)
A  -> AB [Reflexivity]
A  -> C  [Transitivity (2) and (1)]
     (note: wrong order for transitivity)
     (note: removal of duplicate is allowed)
A  -> B  [Union (1)]
```