Q1:

Output:

hye from message1

Non-dependency and only one method that does not need to be predicted will appear as it is

Q2:

Output:

hye from message1

hye from message2

Its appearance depends on the appearance of the getMassage1() method due to the presence of @Qualifier

Q3:

Output:

Case1:

hye from message1

hye from message3

hye from message2

@Qualifier was used 3 times, but the only time it was used was in getMassage2(@Qualifier("3") String data), so the reason for the appearance of this method depends on the appearance of getMassage3()

Case2:

hye from message3

hye from message2

hye from message1

Q4:

Output:

Case1:

hye from message1

hye from Main controller

hye from message3

hye from message2

The appearance of message1 depends on the appearance of the Main controller, so message1 will appear, then the Main controller, as well as message2, depend on the appearance of message3

Case2:

hye from message3

hye from message2

hye from message1

hye from Main controller

Q5:

Output:

hye from message3

hye from message2

hye from Main controller

hye from message1

Massage3 appeared because it is non-dependent, then Massage2 because it depends on Massage3, and MainController because it depends on Massage2, and the last message will be Massage1.

***Non-dependencies are always printed before dependencies, unless @Qualifier is placed based on the connection of the method to each other. The codes and values placed inside @Qualifier are printed just as symbols for the link. Number 1 does not mean that it is in the order of appearance. The first will be just predictions of the appearance. The first non-dependency method I expect will appear first. Then, if there is a link. It has another method that I put, and if I do not find it, I move to the other non-reliable method, and so on until I reach the reliable one.**