**Spring Framework**

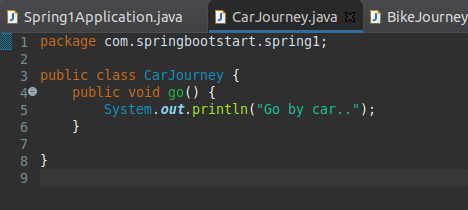
**Name: Akanksha Tyagi**

**Id: 4701**

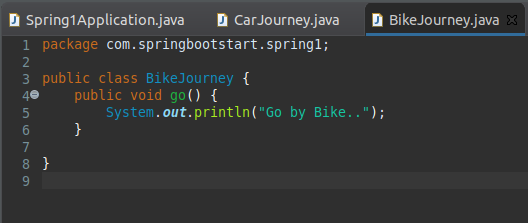
**Question 1:**

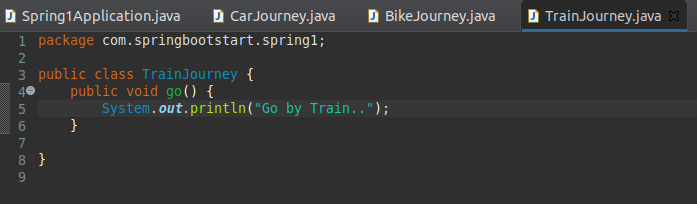
**Program to illustrate the tight coupling.**

carjourney.java

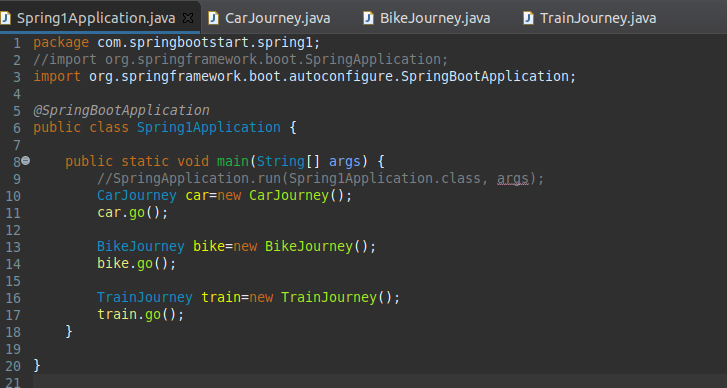


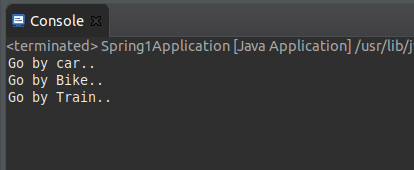
BikeJourney.java



TrainJourney.java

To run these classes, we need to create objects separately.

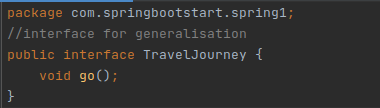




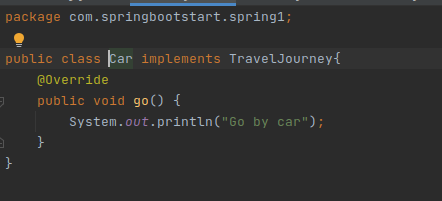
**Question 2:**

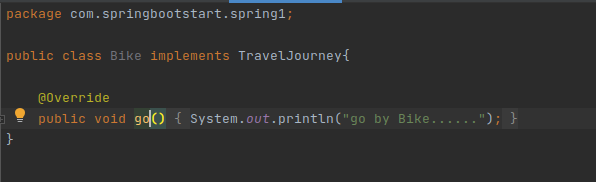
**Program to illustrate the loose coupling.**

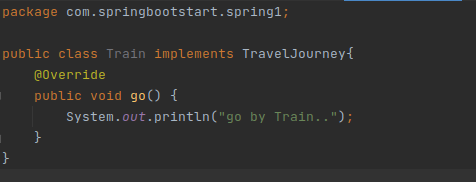
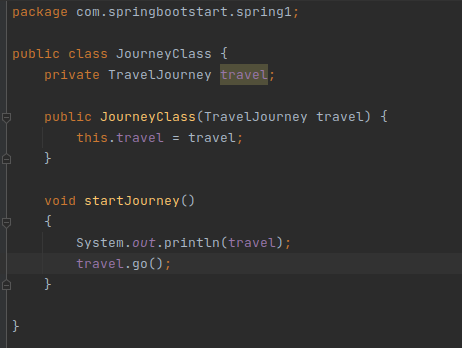
// interface

Travel.java 

//classes implementing interface

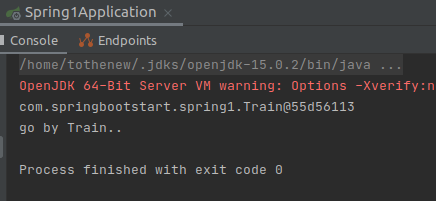
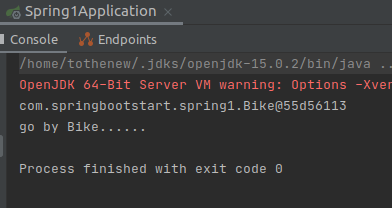
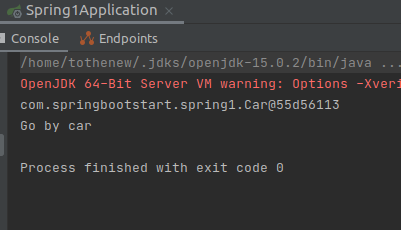
Car.java 

Bike.java 

Train.java JourneyClass.java 

Main class 

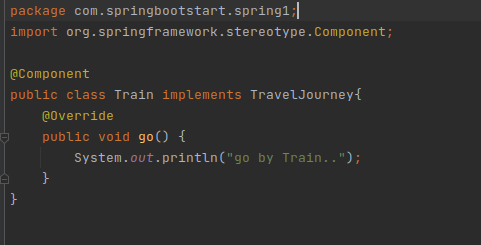
Outputs:

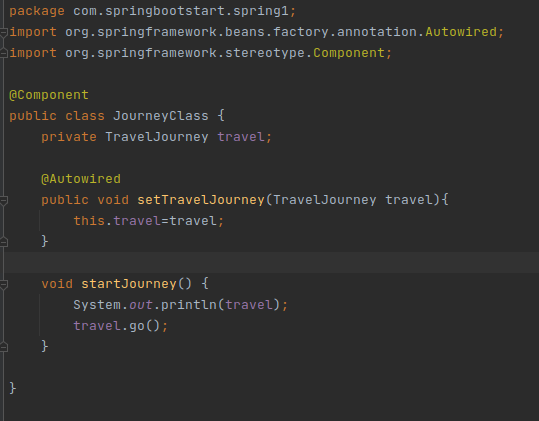


**Question 3:**

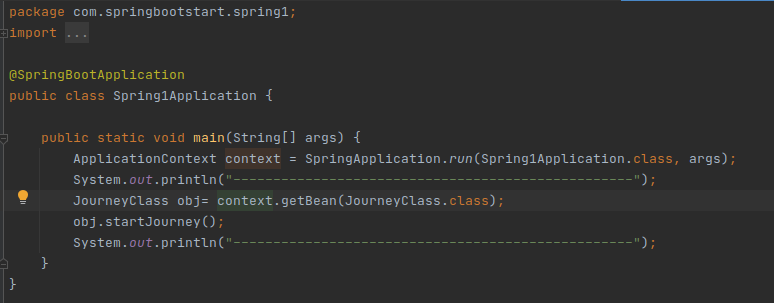
**Use @Compenent and @Autowired annotations to in Loosely Coupled code for dependency management.**

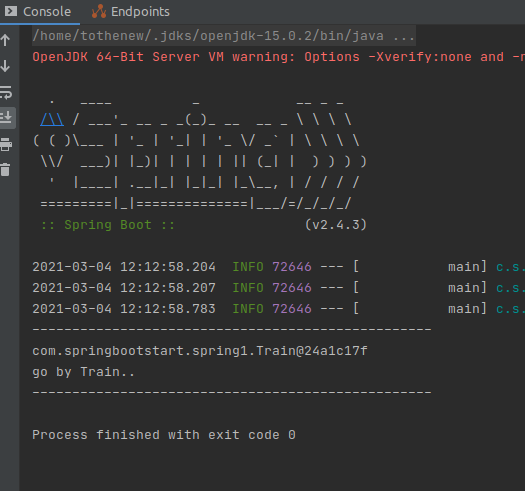
Adding @Component to Train class

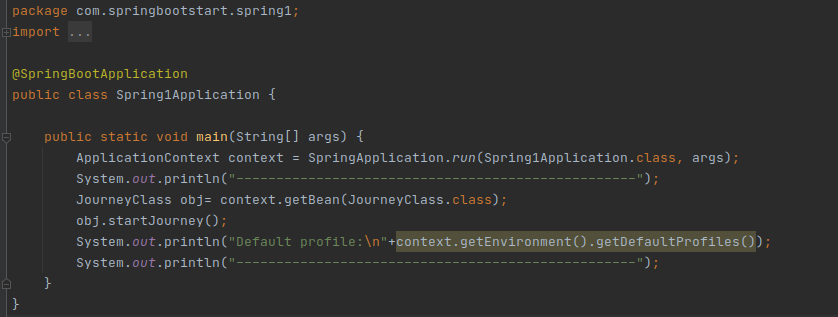


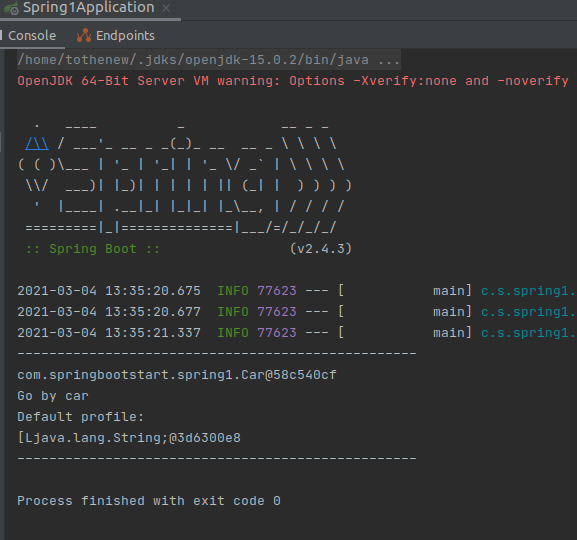
Adding @Component to Journey class and @Autowired to the setter method

Adding the dependency injection through ApplicationContext



Output: 

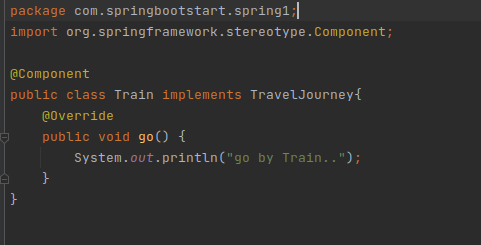
**Question 4: Get a Spring Bean from application context and display its properties.**

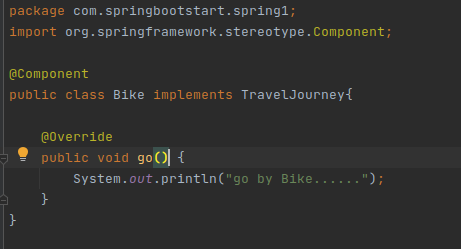


**Question 5: Demonstrate how you will resolve ambiguity while autowiring bean (Hint : @Primary)**

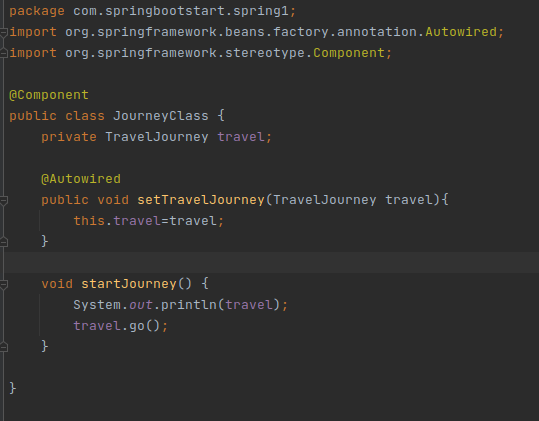
----------AMBIGUITY-------

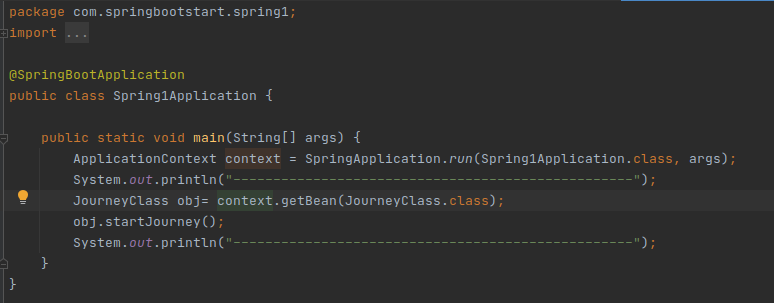
Adding @Component to train class



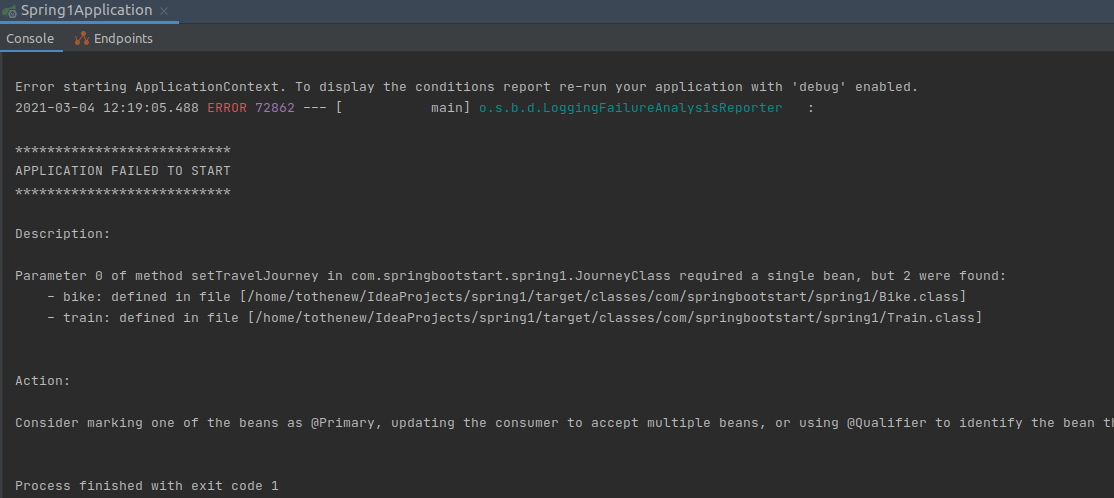
Adding @Component to Bike class

Adding @Component and @Autowired to JourneyClass



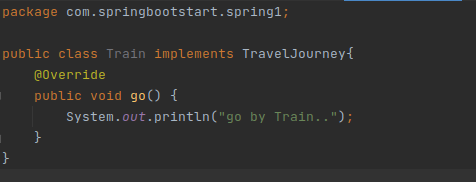
Main class

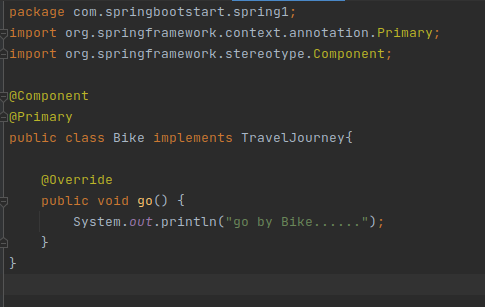
Output:

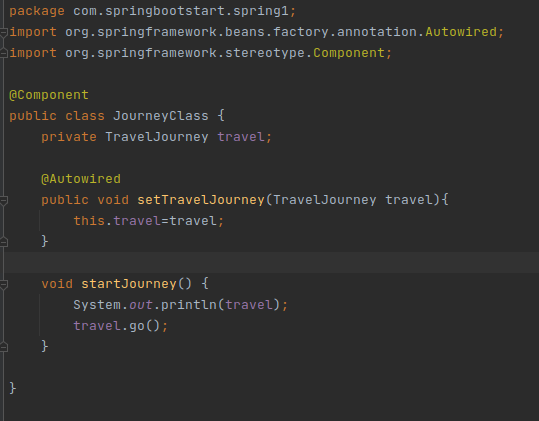


RESOLVING AMBIGUITY

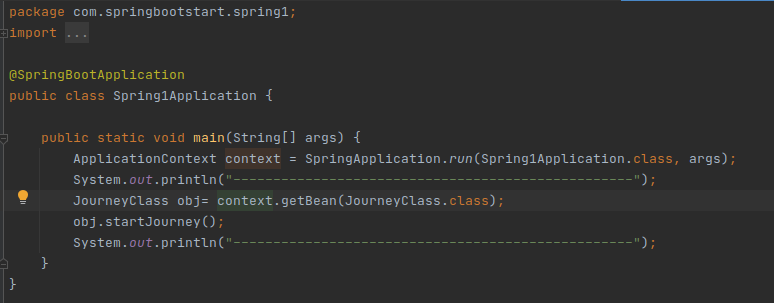
Train Class



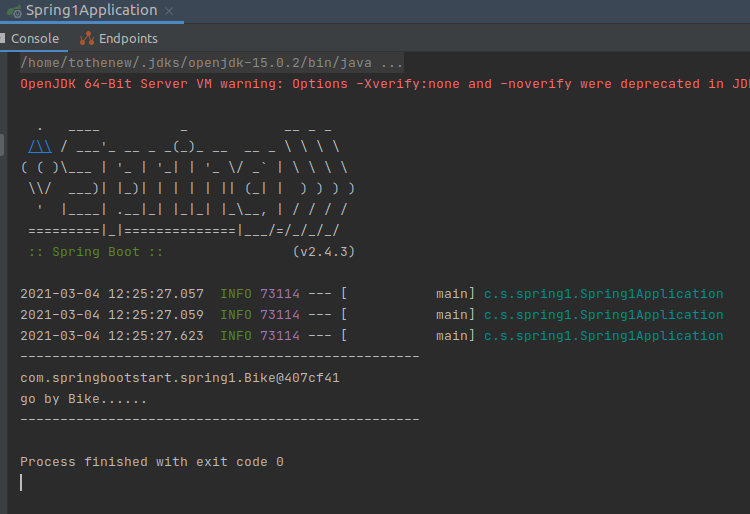
Adding @Primary to the Bike Class and the ambiguity is resolved.

JourneyClass: 

Main class:

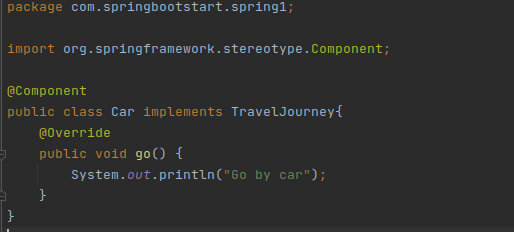


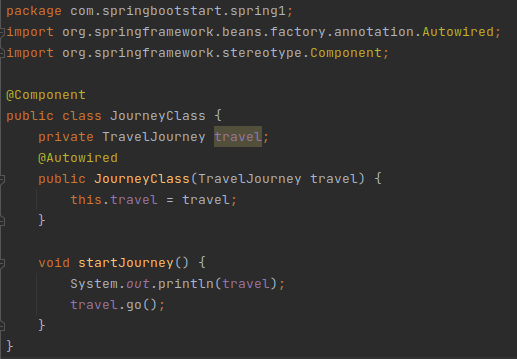
Output:



**Question 6: Perform Constructor Injection in a Spring Bean.**

Adding @Component to the car



@Autowired on Constructor/ constructor injection

Output:

