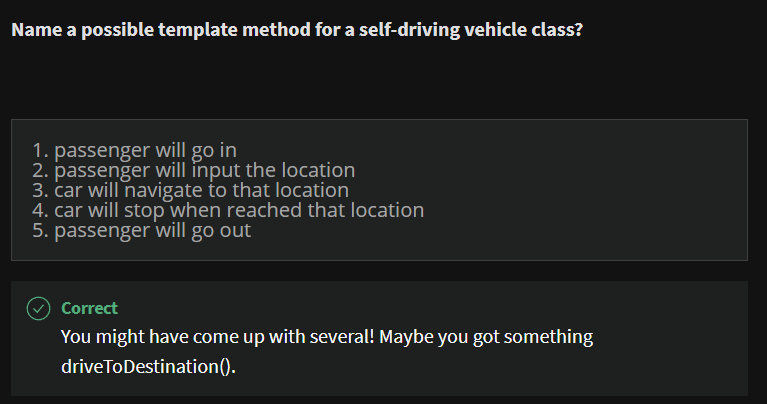
Behavioral

* Roles
* Identifies common communication patterns among objects

Template Method

* Algorithm steps
* Assignment of responsibilities
* Generalization and inheritance

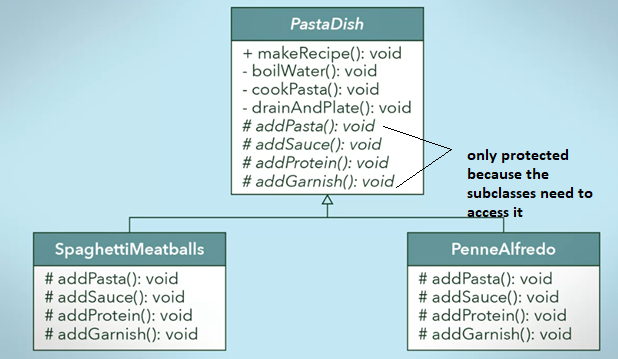


Template method example

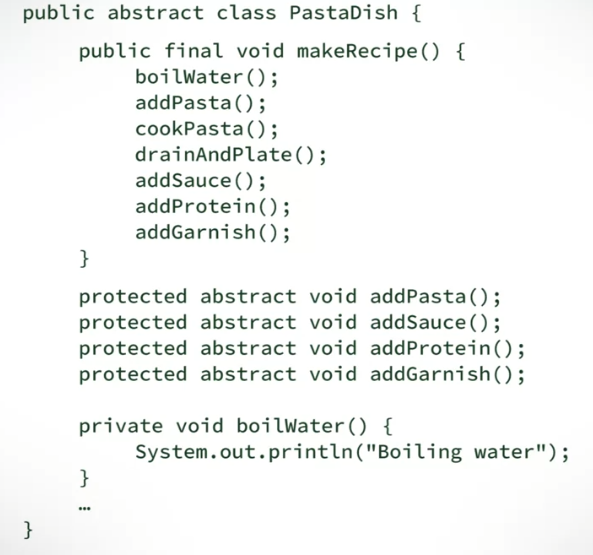
Drive to destination

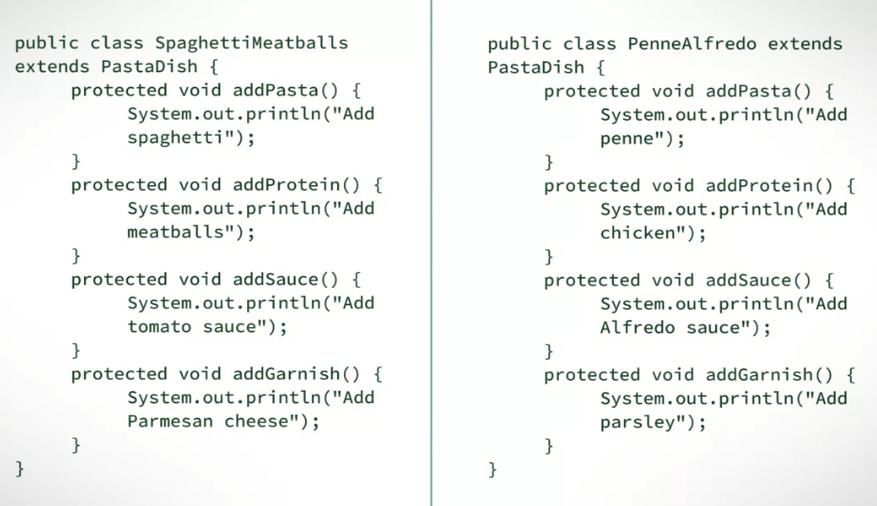
1. Steering()
   1. Different implementations when car/motorcycle is used
2. driveToDestination()
3. accelerating()

UML of PastaDish

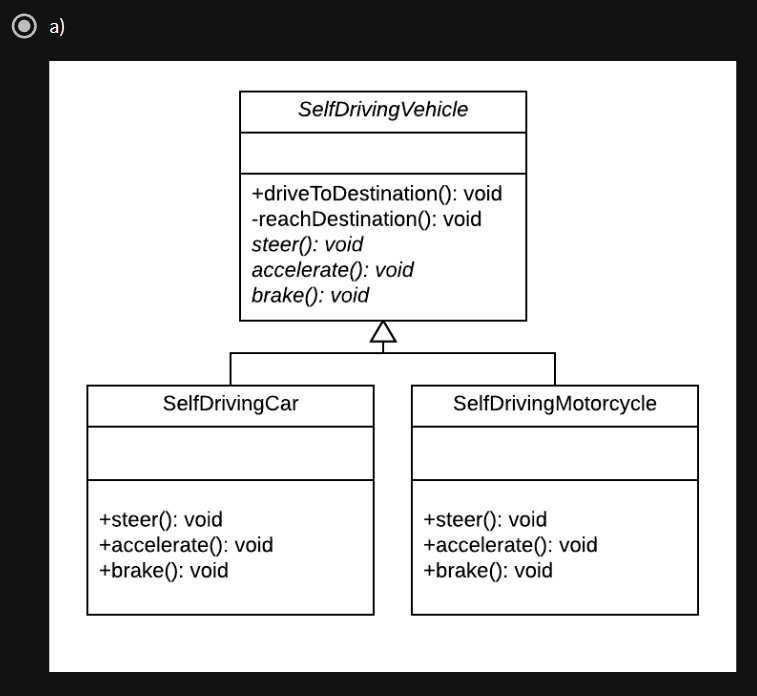


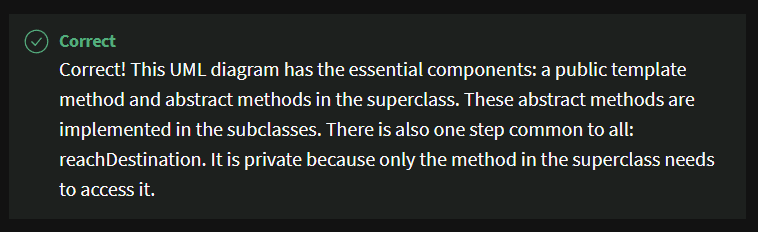
Java Code for PastaDish



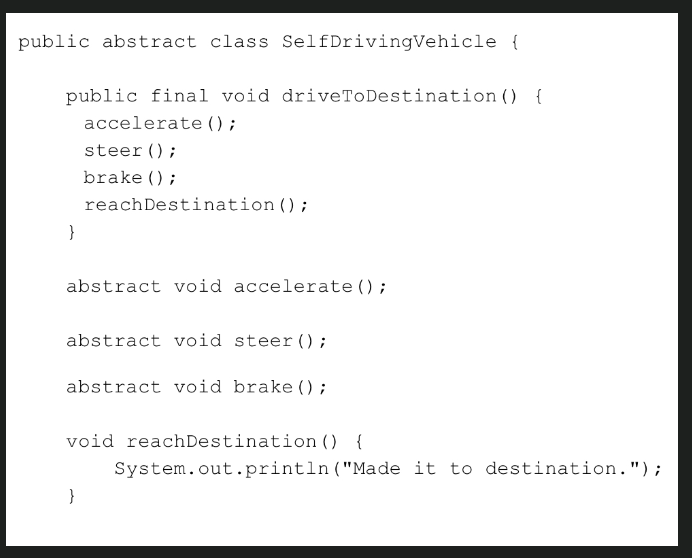


UML of SelfDrivingVehicle



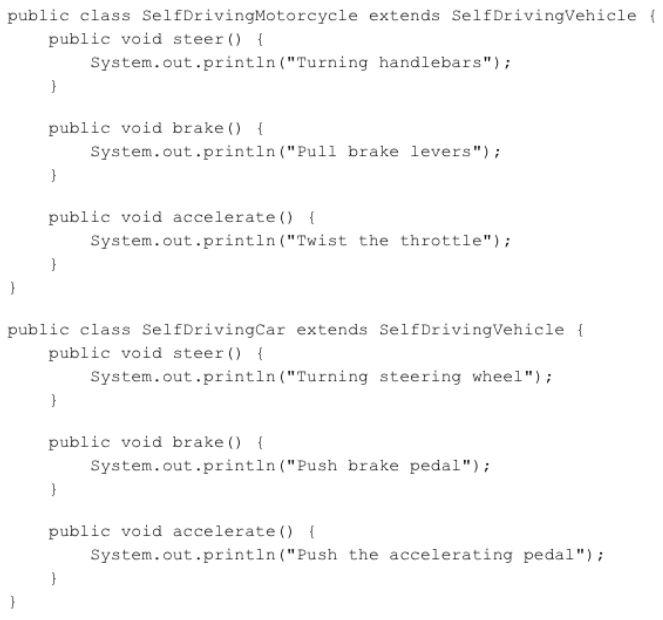


Java Code for SelfDrivingVehicle



If your answer looked something like this, good job! You answer **must** have 2 main components:

1. A Template method that has several steps (driveToDestination in this example)
2. The steps which are defined by the subclasses (accelerate, steer, and brake in this example) are defined abstractly!
3. Your answer **may** have some steps that are common to all the subclasses. These are defined in the superclass (reachDestination in this example)



In this question, you need to implement the steer, brake and accelerate methods for the subclasses. Two possible subclasses are the self-driving car and the self-driving motorcycle. For each of these subclasses, you must implement the abstract methods, in very different ways.