1. (Duck → MallardDuck) → IS-A
   Justification: MallardDuck is a subclass of Duck, this shows that MallardDuck is a type of Duck and inherits all its properties and behaviors. This is an example of the IS-A relationship.
2. (Duck → FlyBehavior) → HAS-A
   Justification: Duck has a reference to FlyBehavior, that means a Duck has an instance of FlyBehavior This is an example of the HAS-A relationship, where Duck possesses a FlyBehavior.
3. (Duck → QuackBehavior) → HAS-A
   Justification: Similar to the previous relationship, Duck also has a reference to QuackBehavior. A Duck has a QuackBehavior which shows how it quacks. This is another instance of the HAS-A relationship.
4. (RubberDuck → Duck) → IS-A
   Justification: RubberDuck is a type of Duck, that means it inherits from the Duck class. This forms an IS-A relationship, where RubberDuck is a Duck, but with special behavior.
5. (DecoyDuck → Duck) → IS-A
   Justification: DecoyDuck extends Duck, shows that DecoyDuck is a type of Duck. This also follows the IS-A relationship, where DecoyDuck inherits from the base Duck class.
6. (FlyWithWings → FlyBehavior) → IS-A
   Justification: FlyWithWings is a solid implementation of the FlyBehavior interface, meaning that FlyWithWings is a FlyBehavior. It implements the calculateFlight() method of the FlyBehavior interface, indicating it is a special type of FlyBehavior.
7. (Quack → QuackBehavior) → IS-A
   Justification: Quack is a solid implementation of the QuackBehavior interface, meaning Quack is a QuackBehavior that defines a specific quacking behavior. This follows the IS-A relationship, where Quack is a type of QuackBehavior.