**Question 1: Inheritance vs. Composition**

I like to think of inheritance as a classification method. For example, imagine you are unloading the dishwasher and putting away the silverware. You have forks, knives, and spoons, and each type of silverware has a specific slot in the drawer. Here, *silverware* is the parent and *forks*, *knives*, and *spoons* are the children.

The potential drawback with inheritance is that parents and their children are *tightly coupled*. This means that a change in the parent class directly impacts the child, and could break your code. Take the following example: usually when I unload the dishwasher, I put the silverware in the silverware drawer, in the correct slots. However, when it is time to move out of my apartment, I will take the silverware from the dishwasher and put them in a box instead. The box does not have slots, and as such, the child classes will no longer work.

class Silverware {

func toKitchenBox() { }

}

class Fork: Silverware {

func toLeftSlot() { }

}

class Knife: Silverware {

func toMiddleSlot() { }

}

class Spoon: Silverware {

func toRightSlot() { }

}

class Silverware {

func toSilverwareDrawer() { }

}

class Fork: Silverware {

func toLeftSlot() { }

}

class Knife: Silverware {

func toMiddleSlot() { }

}

class Spoon: Silverware {

func toRightSlot() { }

}

*Version 1: Standard*  *Version 2: Moving Day*