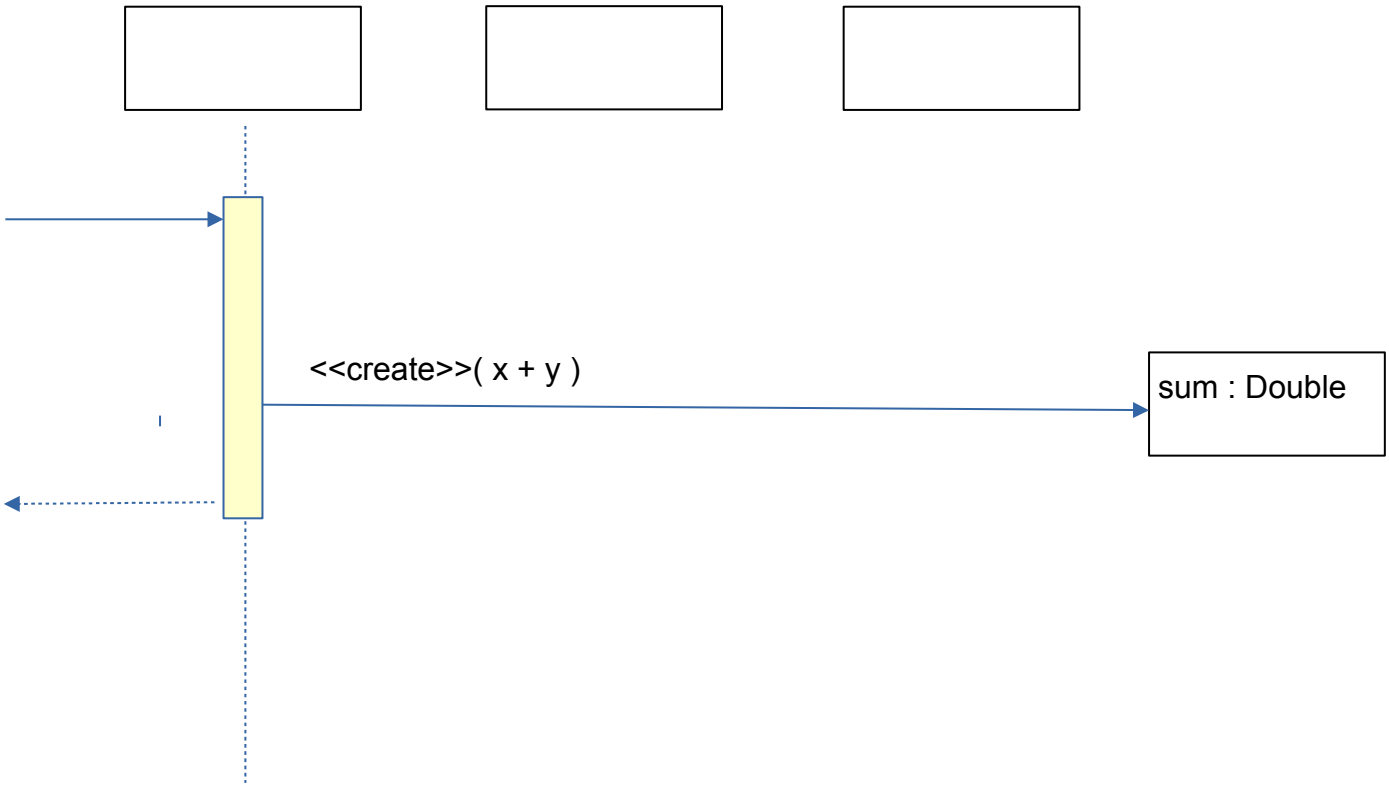
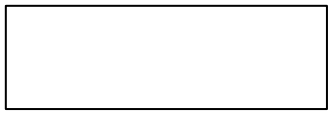
1. Draw a sequence diagram for this code. The class of the add method is unknown, so leave its box blank.

```
public Double add( Double a, Double b ) {
          double x = a.doubleValue();
          double y = b.doubleValue();
          Double sum = new Double(x+y);
          return sum;
     }
```



2. Draw a sequence diagram of what happens when **`sale.computeVat(m)`** is invoked with a **`Money`** reference **`m`**. Show the call to "computeVat(m)" as the "found" message. The two object boxes are sale and money.

```
class Sale {
     public Money computeVat( Money money ) {
          double amount = money.getValue( );
          String currency = money.getCurrency();
          // create new object to represent VAT tax
          Money tax = new Money(amount * 0.07, currency);
          return tax;
     }
```



In the `computeVat( )` code, what is **bad programming**? Explain why its bad.

3. Fill in the blanks to show how to use an *Iterator* to sum the **value** of a *Collection* of coins.

```
public double sum( Collection<Coin> coins ) {
     _____________ iterator = coins._____________;
     double sum = 0.0;
     while ( ____________________ ) {
          sum += __________. ___________ . _________________ ;
     }
     return sum;
}
```

4. Draw a UML sequence diagram for the previous exercise. Use one Coin object to represent the coins in the loop (even though in fact they would be different). Note that in a sequence diagram you don't have to show operations performed inside a method (like sum = sum + 1) but you *can* show them by writing the operation inside an oval next to the activation box.

5. Here is some code to create and show a window in JavaFX.

```
public class MyApp extends Application {

    public void launch(Stage stage) {
       final String FXMLFILE = "myview.fxml";
       // MyApp has a method named loadFxml that uses FXMLLoader
       // to load the FXML file
       Parent root = loadFxml( FXMLFILE );
       Scene scene = new Scene(root);
       stage.setScene(scene);
       stage.show( );
    }
```

Show what happens when launch( ) is invoked.