Machen Sie sich nochmals den Algorithmus für die Berechnung der Prüfziffernberechnung klar. Als nächsten Schritt überführen Sie das Programm in die Objektorientierung wie folgt:

1. Modellieren Sie den Sachverhalt mit folgenden Objekten: Bank(-filiale), Bankkunde, Kreditkarte, Bankautomat. Diskutieren Sie die Beziehungen zu den Objekten. Gehen Sie von folgendem vereinfachten Sachverhalt aus: Eine Bank emittiert eine Kreditkarte. Ein Bankkunde beantragt eine Kreditkarte und bekommt diese nach erfolgreicher Prüfung seines Bankkontos. Auf dem Bankkonto sollte mindestens ein Betrag von x Euro liegen, um eine Genehmigung der Kreditkarte zu bekommen.

Bevor ein Kunde am Bankautomat Geld abheben kann, wird eine Prüfziffernberechnung vorgenommen. Bei erfolgreicher Prüfung, erfolgt die Auszahlung des Geldes.

1. Implementieren Sie die Klassen. Zunächst in einem Paket und alles public. Später kapseln Sie die Attribute. Diskutieren Sie in der Gruppe, welche Getter und Setter Methoden sinnvoll sind. Welche Methoden darüber hinaus sollten die Klassen besitzen? Probieren Sie, einzelne Objekte zu erzeugen und Methoden zu testen.
2. Schreiben Sie ein Programm, welches notwendige Objekte erzeugt und verschiedene Use-Cases abbildet. Ein Sequenzdiagramm hilft, Use-Cases zu definieren und abzubilden. (https://de.wikipedia.org/wiki/Sequenzdiagramm).
3. Überlegen Sie sich den Sachverhalt zu erweitern. Hier könnte man auch eine Bankaufsicht (BAFIN, eine Behörde) hinzufügen, die Banken überprüfen/beaufsichtigen. Eine Bank muss, um geschäftsfähig zu sein, Bedingungen erfüllen. Dies gilt es von der BAFIN zu prüfen. Recherchieren Sie, was eine BAFIN beispielsweise prüft. Die BAFIN gibt es nur einmal (-> Singelton). Auch wären noch explizit Kreditkartenunternehmen modellierbar. Eine weitere Idee wäre, dass Privatpersonen/Geschäftskunden mit der Kreditkarte einkaufen. Eine Kreditkarte kann auch mal verloren gehen. Lassen Sie ein wenig der Phantasie freien Lauf, aber bleibe Sie fokussiert – also nicht zu viel auf einmalmachen.
4. Bevor Sie die BAFIN und andere Use-Cases implementieren, erweitern Sie das zugehörige UML Diagramm und erstellen Sie zugehörige Sequenzdiagramme.
5. Implementieren Sie die Erweiterungen.