Lösungen Struktogramme

***Hinweise**:*

Wir verwenden für die Aufgaben das Projekt „Girokonto“ (Abi-Projekt 2014, Aufgabe 2.3 (Girokonten, Kundenbetreuer …)). Alle Aufgaben sind in der Klasse Kundenbetreuer durchzuführen.

+++ Aufgaben +++

Bitte IMMER ZUERST ein Struktogramm anfertigen, bevor Sie zu programmieren beginnen!

Alle programmierten Methoden auf korrekte Lauffähigkeit prüfen (via Startklasse).

Struktogramm und Programmcode in ein (neues, altes?) Vorlagendokument einfügen (Code als "code" formatieren), korrekte Überschriften zuweisen, zwischendurch abspeichern.

----- (A) Einsteigeraufgaben -----

A1) Programmieren Sie eine Methode datenAusgeben(kontoinhaber:Kontoinhaber):void, die die Daten eines Kunden ausgibt in Form

Heini Müller - Kontostand: 3000 Euro

```
+datenAusgeben(kontoinhaber:Kontoinhaber):void
    Ausgabe : kontoinhaber.getVorname() + " " + kontoinhaber.getName() + " -
    Kontostand " + kontoinhaber.getGirokonto().getKontostand() + " Euro"
```

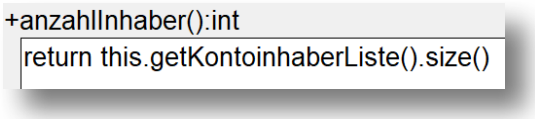
A2) Wie A1, nur wird der String nicht AUSGEGEBEN, sondern ZURÜCKGEGEBEN. Die Methode heißt dann datenZurueckgeben(kontoinhaber:Kontoinhaber):String

```
+datenZurueckgeben(kontoinhaber:Kontoinhaber):String
    return kontoinhaber.getVorname() + " " + kontoinhaber.getName() + " - Kontostand: " +
    kontoinhaber.getGirokonto().getKontostand()
```

oder mit lokaler Variable, die den Rückgabewert speichert:

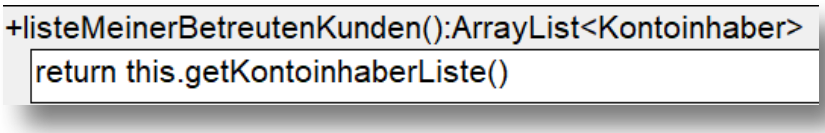
```
+datenZurueckgeben(kontoinhaber:Kontoinhaber):String
    rueckgabe:String = ""
    rueckgabe = rueckgabe + kontoinhaber.getVorname() + " " + kontoinhaber.getName() +
    " - Kontostand: " + kontoinhaber.getGirokonto().getKontostand()
    return rueckgabe
```

A3) Programmieren Sie eine Methode anzahlInhaber():int, die die Anzahl der von einem Betreuer betreuten Kontoinhaber zurückgibt.



A4) Programmieren Sie eine Methode listeMeinerBetreutenKunden():ArrayList<Kontoinhaber>, die eine Liste aller betreuten Kunden ZURÜCKGIBT.

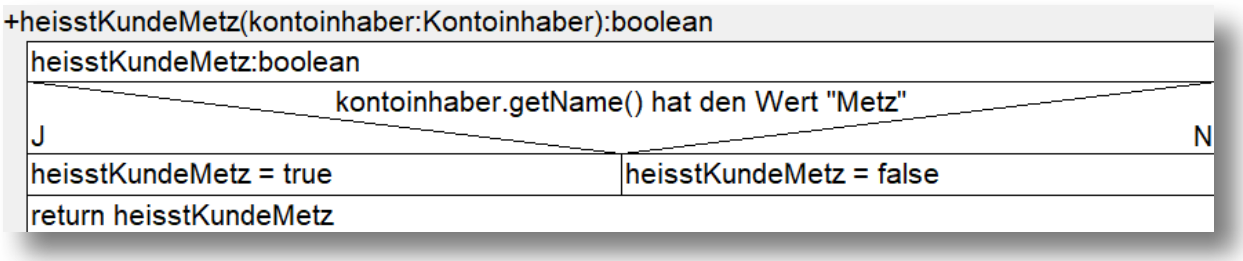
(Wenn Sie das in der Startklasse ausprobieren, benötigen Sie für die Anzeige eine foreach-Schleife.)



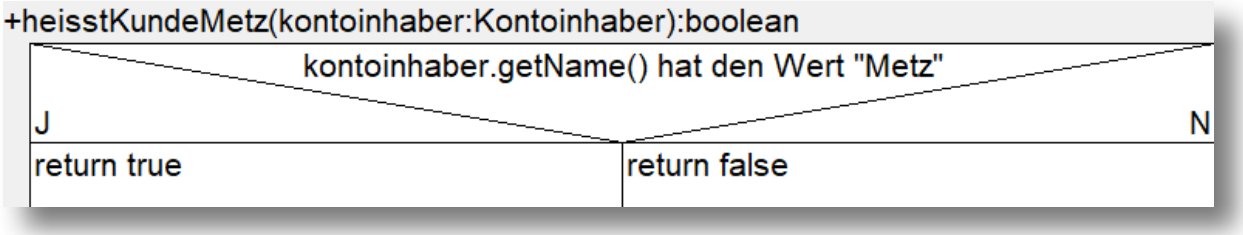
----- (B) Einfache if-Verzweigungen ------

B1) Programmieren Sie eine Methode heisstKundeMetz(kontoinhaber:Kontoinhaber):boolean, die prüft, ob der Nachname eines Kunden "Metz" ist. Zur Erinnerung: Verwenden Sie bei String-Vergleichen nicht "==", sondern "equals(…)", also bspw. kundename.equals("Metz")

Möglichkeit 1: Lokale Variable verwenden, die den Rückgabewert speichert:



Möglichkeit 2: Rückgabe direkt im Bedingungszweig vornehmen.

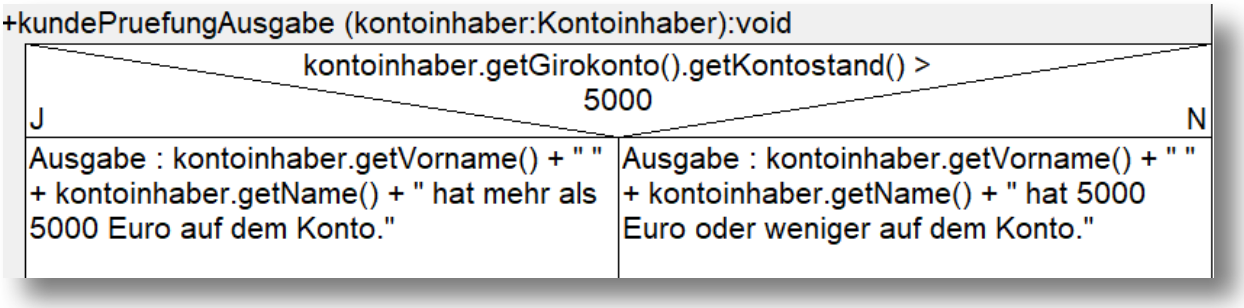


B2) Programmieren Sie eine Methode kundePruefungAusgabe(kontoinhaber:Kontoinhaber):void, die abhängig vom Kontostand ausgibt:

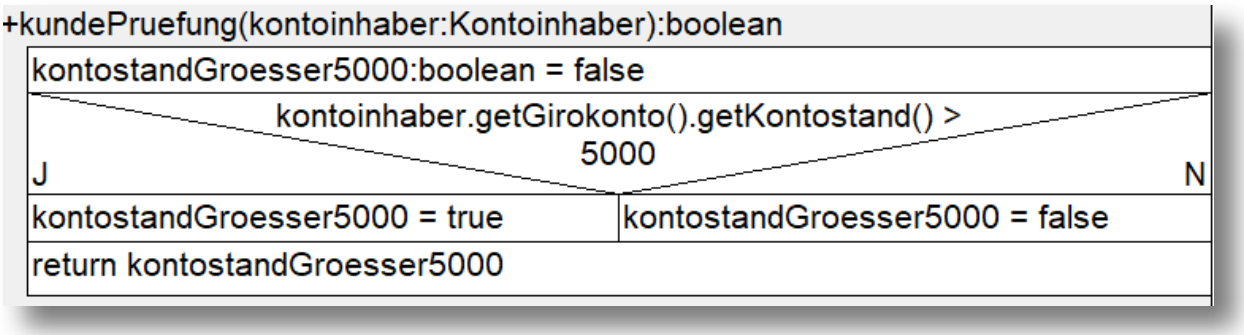
Heini Müller hat mehr als 5000 Euro auf dem Konto.

oder

Heini Müller hat 5000 Euro oder weniger auf dem Konto.



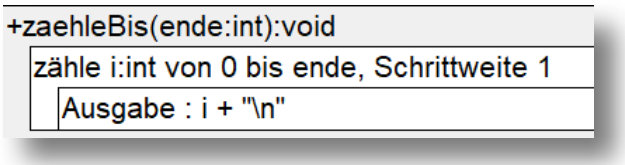
B3) Programmieren Sie eine Methode kundePruefung(kontoinhaber:Kontoinhaber):boolean, die abhängig vom Kontostand eines Kunden true (bei mehr als 5000 Euro) oder false (bei 5000 oder weniger) zurückgibt.



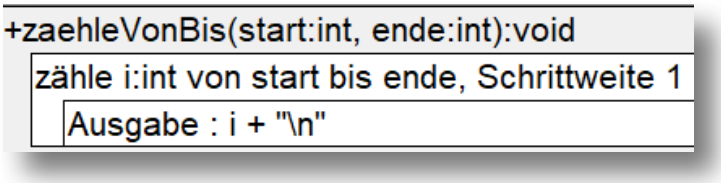
----- (C) Einfache Schleifen-----

----- (C1) For-Schleifen

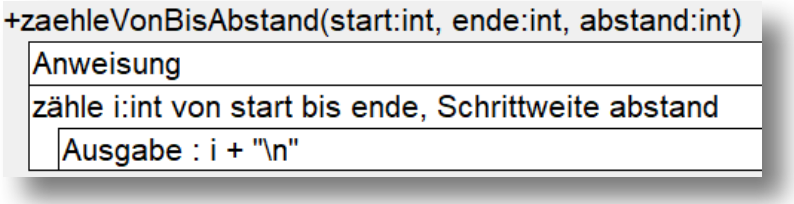
C1-1) Programmieren Sie eine Methode zaehleBis(ende:int), die von 0 bis ende zählt und die Zahlen ausgibt.



C1-2) Programmieren Sie eine Methode zaehleVonBis(start:int, ende:int), die von start bis ende zählt und die Zahlen ausgibt.

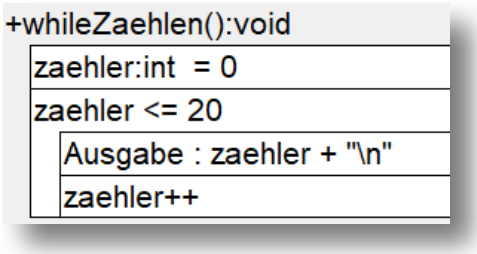


C1-3) Programmieren Sie eine Methode zaehleVonBisAbstand(start:int, ende:int, abstand:int), die von start bis ende zählt und die Zahlen ausgibt.

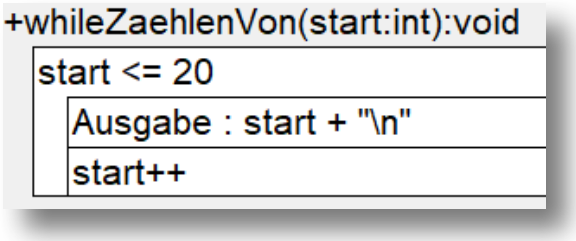


----- (C2) While-Schleifen

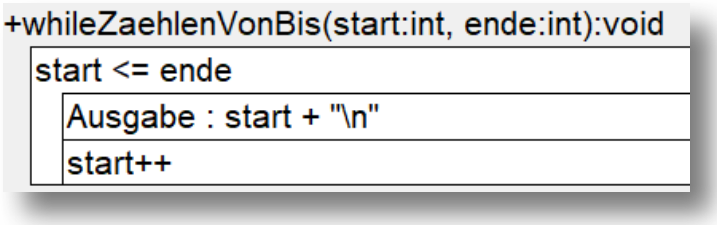
C2-1) Programmieren Sie eine Methode whileZaehlen(), die von 0 bis 20 zählt und die Zahlen ausgibt. Verwendung einer while-Schleife.



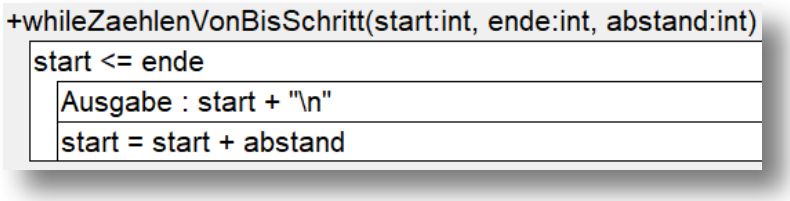
C2-2) Programmieren Sie eine Methode whileZaehlenVon(start:int), die von start bis 20 zählt und die Zahlen ausgibt. Verwendung einer while-Schleife.



C2-3) Programmieren Sie eine Methode whileZaehlenVonBis(start:int, ende:int), die von start bis ende zählt und die Zahlen ausgibt. Verwendung einer while-Schleife.



C2-4) Programmieren Sie eine Methode whileZaehlenVonBisSchritt(start:int, ende:int, abstand:int), die von start bis ende in Schrittweite abstand zählt und die Zahlen ausgibt. Verwendung einer while-Schleife.



----- (D) Einfache foreach-Schleifen ------

D2) Kundenliste

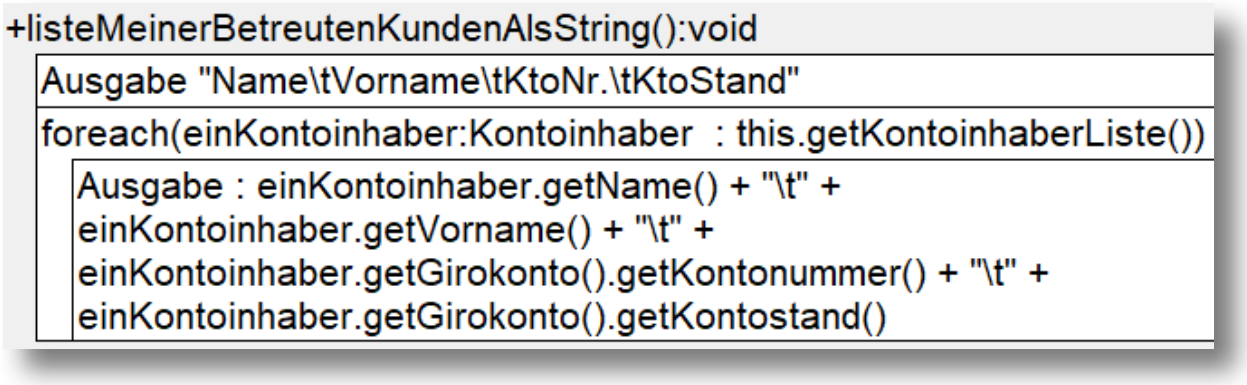
D2-1) Programmieren Sie eine Methode listeMeinerBetreutenKundenAlsString():void, die eine Liste aller betreuten Kunden AUSGIBT (name, vorname, kontonummer, Kontostand). Spalten mit \t erzeugen.

Name Vorname KtoNr. KtoStand

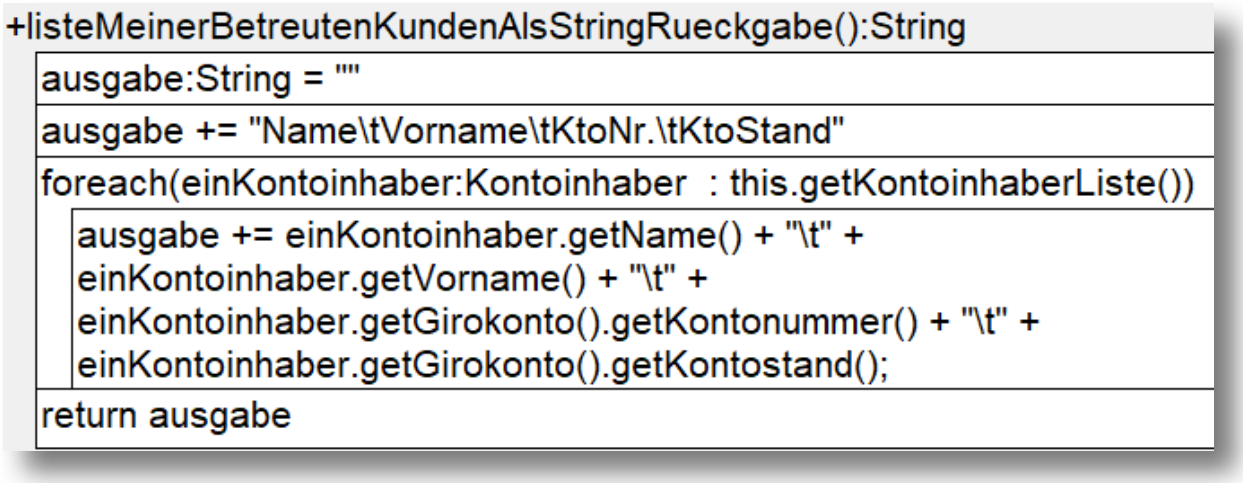
Schmitt Heini 3321314 45 Euro

Müller Bilal 587373 28348 Euro

(Es kann dabei zu Verschiebungen kommen, wenn Elemente besonders lang oder kurz sind)

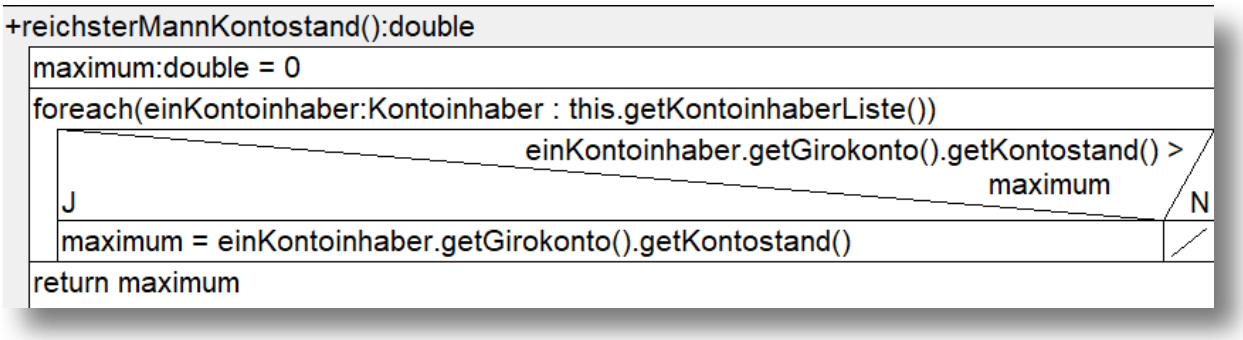


D2-2) Programmieren Sie eine Methode listeMeinerBetreutenKundenAlsStringRueckgabe():String, die eine Liste wie oben ZURÜCKGIBT. Sie müssen dazu D2-1 nur geringfügig umarbeiten.

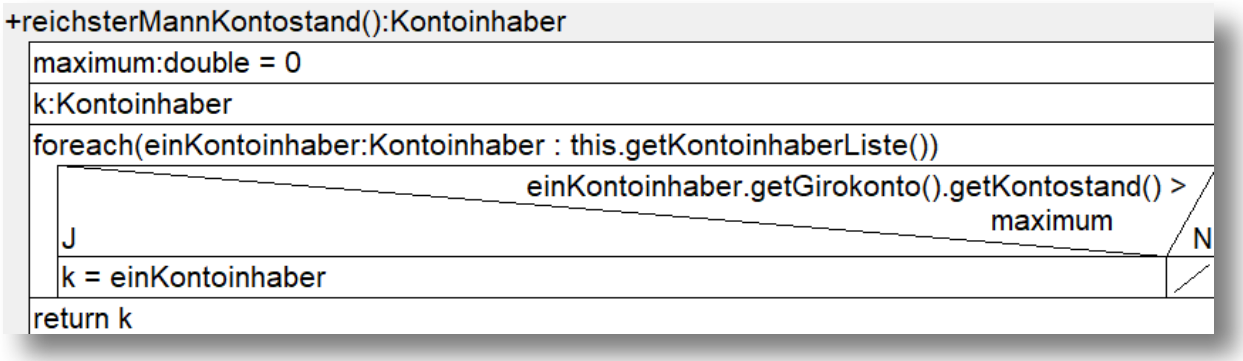


D3) Wir wollen wissen, wer den höchsten Kontostand hat.

D3-1) Programmieren Sie eine Methode reichsterMannKontostand():double, die den Kontostand des Kontoinhabers mit dem höchsten Kontostand zurückgibt.

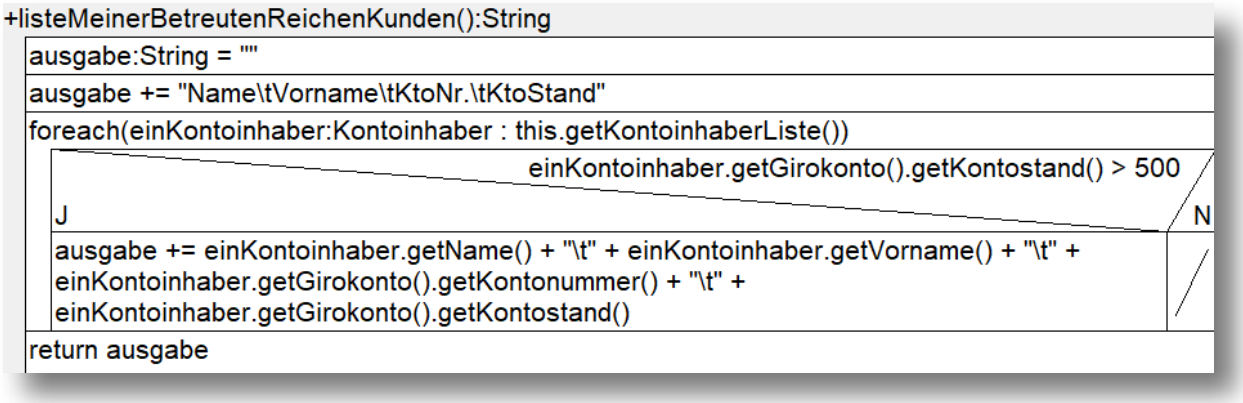


D3-2) Programmieren Sie eine Methode reichsterMann():Kontoinhaber, die den Kontoinhaber mit dem höchsten Kontostand zurückgibt.



----- (E) Kombination: Schleife/if ------

Programmieren Sie eine Methode listeMeinerBetreutenReichenKunden():String, die eine Liste wie in Aufgabe D2-1 ZURÜCKGIBT (als String!), allerdings sind dort nur die Kontoinhaber mit einem Kontostand von mehr als 500 Euro enthalten.



----- (F) Schwierige Aufgaben ------

F1) Programmieren Sie eine Methode reichsteKunden():ArrayList<Kontoinhaber>, die eine Liste aller Kunden zurückgibt, deren Kontostand über dem durchschnittlichen Kontostand aller Kunden liegt.