Informatik 2 für Regenerative Energien

Klausur vom 10. Juli 2014

Jörn Loviscach

Versionsstand: 9. Juli 2014, 00:18

15 Punkte für die erste Aufgabe; 3 Punkte für alle weiteren Aufgaben. Mindestpunktzahl zum Bestehen: 20 Punkte. Hilfsmittel: maximal vier einseitig oder zwei beidseitig beschriftete DIN-A4-Spickzettel beliebigen Inhalts, möglichst selbst verfasst oder zusammengestellt; kein Skript, keine anderen Texte, kein Taschenrechner, kein Computer, kein Handy und Ähnliches.

Name	Vorname	Matrikelnummer	E-Mail-Adresse, falls nicht in ILIAS

1. Im C#-Programmlisting im Anhang sind 15 Fehler, darunter keine Tippfehler und höchstens ein Fehler pro Zeile. Erstellen Sie eine Liste mit 15 Zeilen aus den Fehlern und ihren jeweiligen Korrekturen, nach dem folgenden Muster:

Zeile	korrekter Programmtext
123	`public void foo()`
543	`int a = 42;`

2. Mit dem (korrigierten) Code aus dem Programmlisting im Anhang wird die Methode `Test.Teste()` ausgeführt. Welche Zeichenkette steht in den Zeilen 27 bis 29 in der Variablen `s`, welche Zahlenwerte stehen in den Variablen `kmAnton` und `kmBerta`? Beschreiben Sie gegebenenfalls, wie Sie zu Ihrer Antwort kommen.

3. Die Methode `BeendeFahrt` der Klasse `Kfz` soll nur nach `BeginneFahrt` aufgerufen werden können, nicht mehrfach hintereinander. Erweitern Sie die Klasse so, dass ein falscher Aufruf von `BeendeFahrt` zu einer Exception führt.

4. Erweitern Sie in die Klasse `Kfz` aus dem (korrigierten) Listing im Anhang um eine öffentliche Methode `string ListeFahrerUndKilometer()`, die die Fahrer und ihre gefahrenen Kilometer nach folgendem Muster als Zeichenkette zurückgibt, samt Zeilenumbrüchen:

   ```
   Doris: 2354
   Egon: 354
   Fred: 5432
   ```

   Sie können dabei die bereits vorhandenen Methoden benutzen!

5. Schreiben Sie für das (korrigierte) Listing im Anhang eine Klasse `FahrtInBielefeld`, die von `Fahrt` erbt, aber als Startort `"Bielefeld"` enthält, mit einem entsprechenden Konstruktor. Die Methode `KommeAn` soll in dieser Klasse auch als Zielort `"Bielefeld"` eintragen. Gegebenenfalls sind auch an der Klasse `Fahrt` Änderungen nötig. Welche?

6. Die Objekte der Klasse `Kfz` aus dem (korrigierten) Listing im Anhang sollen in ihrer Liste `fahrtenbuch` nicht nur die Fahrten speichern, sondern auch die Zeitpunkte von Inspektionen. Beschreiben Sie in wenigen Sätzen, welche Klassen man wie anlegen und verwenden sollte, um das zu verwirklichen.

7. Stellen Sie das Folgende mit Klassen, Attributen und Vererbung als UML-Diagramm dar: *Jeder Punkt hat eine x- und eine y-Koordinate als Gleitkommazahl; jede Strecke hat einen Anfangs- und einen Endpunkt; jedes Dreieck hat drei Eckpunkte. Punkt, Strecke und Dreieck sind geometrische Objekte.* Damit Kursivschrift (falls nötig) zu erkennen ist, umkringeln Sie diese oder benutzen Sie eine andere Farbe dafür.

8. Welche Zahlen stehen nach Ausführung dieses C#-Programmfragments in den Variablen `x`, `y` und `z`? Geben Sie möglichst auch Zwischenschritte an, damit Ihr Gedankengang nachvollziehbar ist.

   ```
   List<List<int>> a = new List<List<int>>();
   List<int> b = new List<int>();
   b.Add(3);
   b.Add(4);
   b.Add(5);
   a.Add(b);
   a.Add(new List<int>());
   a.Add(b);
   a[0][2] = 6;
   a.RemoveAt(1);
   int x = a.Count;
   int y = a[0].Count;
   int z = a[1][2];
   ```

Dieses Listing enthält 15 Fehler!

Dies soll ein Programm für eine elektronisches Fahrtenbuch werden. Die Methode `Teste` macht die Handhabung der Klassen vor.

```
1  class Test
2  {
3      public static void Teste()
4      {
5          Kfz auto1 = new Kfz("AB123XY", 1234, "Bielefeld");
6          int km = 2345;
7          Kfz auto2 = new Kfz("CD234UV", km, "Bielefeld");
8          Kfz auto3 = auto2;
9          Nachrichtenzentrale nz = new Nachrichtenzentrale();
10         Kfz.SetzeNachrichtenzentrale(nz);
11
12         auto1.BeginneFahrt("Anton");
13         auto1.BeendeFahrt("Hamburg", 1434);
14
15         for (int i = 0; i < 300; i++)
16         {
17             auto2.BeginneFahrt("Berta");
18             km += 100;
19             auto2.BeendeFahrt("Hannover", km);
20             auto2.BeginneFahrt("Clara");
21             km += 100;
22             auto2.BeendeFahrt("Bielefeld", km);
23         }
24
25         auto2.RegistriereInspektion("Anton");
26
27         string s = nz.HoleMeldungen();
28         int kmAnton = auto2.GefahreneKilometer("Anton");
29         int kmBerta = auto3.GefahreneKilometer("Berta");
30     }
31 }
32
33 class Kfz
34 {
35     string kennzeichen;
36     int kilometerstand;
37     string aktuellerOrt;
38     List<Fahrt> fahrtenbuch = new List<Fahrt>;
39     int kmLetzteInspektion;
40     bool inspektionAngefordert;
41     static int Nachrichtenzentrale nachrichtenzentrale;
42
43     public Kfz(string kennzeichen, int kilometerstand, string aktuellerOrt)
44     {
45         this.kennzeichen = kennzeichen;
46         this.kilometerstand = kilometerstand;
47         this.aktuellerOrt = aktuellerOrt;
```

```
48      }
49
50      public void BeginneFahrt(string fahrer)
51      {
52          fahrtenbuch.Add(Fahrt(aktuellerOrt, fahrer));
53      }
54
55      public void BeendeFahrt(string zielort, int kilometerstand)
56      {
57          fahrtenbuch[fahrtenbuch.Count - 1].KommeAn(zielort,
58                                          kilometerstand + this.kilometerstand);
59          this.kilometerstand = kilometerstand;
60          if(this.kilometerstand - kmLetzteInspektion > 10000
61             && !inspektionAngefordert
62             && nachrichtenzentrale != null)
63          {
64              Melde(kennzeichen+": Inspektion fällig!");
65              inspektionAngefordert = true;
66          }
67      }
68
69      public void RegistriereInspektion()
70      {
71          kmLetzteInspektion = kilometerstand;
72          inspektionAngefordert = false;
73      }
74
75      public string[] HoleFahrer()
76      {
77          List<string> fahrerListe = new List<string>();
78          foreach (Fahrt f in fahrtenbuch)
79          {
80              if(!fahrerListe.Contains(f.Fahrer))
81              {
82                  fahrerListe.Add();
83              }
84          }
85          return fahrerListe.ToArray();
86      }
87
88      public int GefahreneKilometer(string fahrer)
89      {
90          int km;
91          foreach (Fahrt f in fahrtenbuch)
92          {
93              if(f.Fahrer == fahrer)
94              {
95                  km += f.holeStreckenlänge();
96              }
97          }
98          return km;
```

```
99      }
100
101     public static void SetzeNachrichtenzentrale(n)
102     {
103         nachrichtenzentrale = n;
104     }
105 }
106
107 abstract class Fahrt
108 {
109     string startort;
110     string zielort;
111     DateTime beginn;
112     DateTime ende;
113
114     string fahrer;
115     string Fahrer
116     { get { return fahrer; } }
117
118     int streckenlänge;
119     public int holeStreckenlänge()
120     {
121         return streckenlänge;
122     }
123
124     public void Fahrt(string startort, string fahrer)
125     {
126         this.startort = startort;
127         beginn = DateTime.Now;
128         this.fahrer = fahrer;
129     }
130
131     public void KommeAn(string zielort, string streckenlänge)
132     {
133         this.zielort = zielort;
134         ende = DateTime.Now;
135         this.streckenlänge = streckenlänge;
136     }
137 }
138
139 class Nachrichtenzentrale
140 {
141     List<string> nachrichten = new List<string>();
142
143     public void Melde(string meldung)
144     {
145         Add(meldung);
146     }
147
148     public string HoleMeldungen()
149     {
```

```
150             string s = "";
151             foreach (m in nachrichten)
152             {
153                 s += m + "; ";
154             }
155             return s;
156         }
157     }
```