Informatik 2 für Regenerative Energien

Klausur vom 23. September 2019

Jörn Loviscach

Versionsstand: 23. September 2019, 12:17

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 (auch nicht wearable), kein Handy und Ähnliches.

Name	Vorname	Matrikelnummer	E-Mail-Adresse

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. Die Methode `Test.Teste` des (korrigierten) Code aus dem Programmlisting im Anhang wird ausgeführt. Welche Werte stehen am Ende in den Variablen `x`, `y`, `z`? Beschreiben Sie in jeweils einem Satz, wie Sie zu diesen drei Werten kommen.

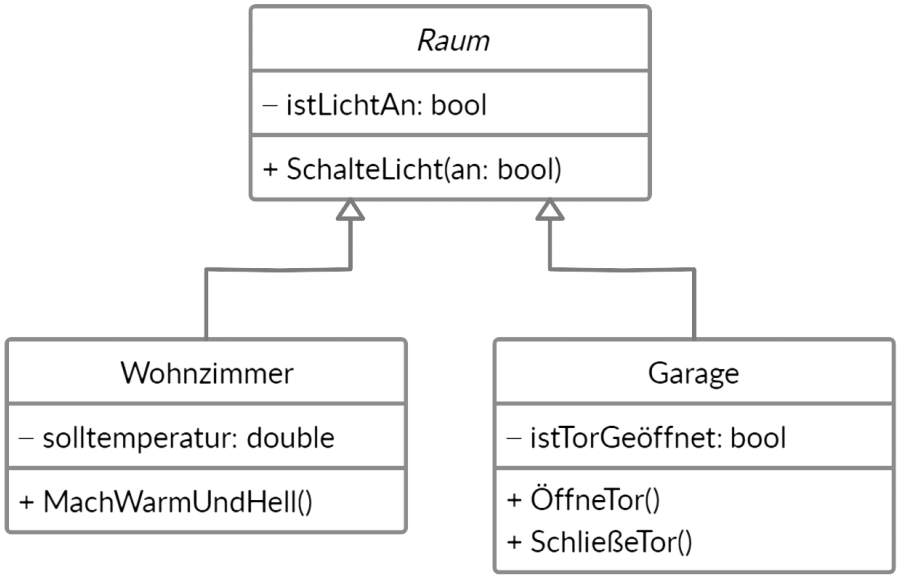
3. Die Methode `SpeichereUndBewerteAntwort` der Klasse `Abfrage` soll eine Exception werfen, wenn die übergebene Zahl keine Nummer einer der möglichen Antworten ist. Was ändern Sie an dazu dieser Klasse?

4. Ergänzen Sie die Klasse `Vokabeltrainer` des korrigierten Code aus dem Programmlisting um eine öffentliche Methode `double BerechneFehlerquote(Vokabel v)`, die entsprechend zu der vorhandenen Methode `double BerechneFehlerquote()` arbeitet, aber die Fehlerquote nur für die übergebene Vokabel berechnet. Was ändern Sie dazu wie an dem (korrigierten) Code aus dem Programmlisting?

5. Schreiben Sie eine von der Klasse `Vokabel` des (korrigierten) Code aus dem Programmlisting abgeleitete Klasse `UnregelmäßigesVerb`. Den Konstruktor dieser neuen Klasse soll man nach dem Muster `UnregelmäßigesVerb("gehen", "go", "went", "gone")` aufrufen können, also mit der Angabe von Past Simple und Past Participle. Der Aufruf `GibText(Sprache.Englisch)` soll dann Zeichenketten nach dem Muster `"go, went, gone"` liefern. Was ändern Sie dazu wie an dem (korrigierten) Code aus dem Programmlisting?

6. Die Klasse `Vokabeltrainer` des korrigierten Code aus dem Programmlisting soll eine öffentliche Methode `FindeSchwierigeVokabeln` erhalten, welche eine Liste aller `Vokabeln` zurückliefert, deren jeweils letzte Abfrage fehlerhaft war. (Vokabeln, die nie abgefragt worden sind, sollen dabei nicht beachtet werden.) Was ändern Sie dazu wie an dem (korrigierten) Code aus dem Programmlisting?

7. Schreiben Sie C#-Klassen samt Feldern und Methoden für das folgende UML-Diagramm. Die Methode `MachWarmUndHell` soll dabei die Solltemperatur auf den Wert 21 stellen und das Licht anschalten.



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.

```
Queue<int> a = new Queue<int>();
a.Enqueue(10);
a.Enqueue(20);
Queue<int> b = new Queue<int>();
b.Enqueue(30);
b.Enqueue(40);
Queue<int> c = a;
c.Enqueue(50);
List<Queue<int>> d = new List<Queue<int>>();
d.Add(a);
d.Add(b);
d.Add(c);
int x = d[0].Dequeue();
int y = d[1].Count;
int z = d[2].Dequeue();
```

Dieses Listing enthält 15 Fehler!

Dieses Programm soll ein einfacher Vokabeltrainer sein. Er soll ein Wort auf Deutsch oder Englisch anzeigen und dann drei Auswahlmöglichkeiten für die Übersetzung anbieten: zwei falsche und die eine richtige, in zufälliger Reihenfolge.

Die Methode `Teste` der Klasse `Test` macht die Benutzung der Klassen vor. Dies ist der Programmcode der Klassen:

```
1  class Test
2  {
3      public static void Teste()
4      {
5          Würfel.SchalteTestmodusEin;
6          Vokabeltrainer vt = new Vokabeltrainer();
7          Vokabel vo1 = new Vokabel("Tag", "day");
8          vt.FügeHinzu(vo1);
9          Vokabel vo2 = new Vokabel("Sonne", "sun");
10         vt.FügeHinzu(vo2);
11         vt.FügeHinzu(new Vokabel("Nacht", "night"));
12         Abfrage a = vt.ErzeugeNeueAbfrage();
13         string x = ErzeugeAbfragetext();
14         // Der Mensch würde die Zeichenkette x lesen
15         // und dann zum Beispiel 2 eingeben.
16         bool y = a.SpeichereUndBewerteAntwort(2);
17         double z = vt.BerechneFehlerquote();
18     }
19 }
20
21 enum Sprache { Deutsch, Englisch }
22
23 abstract class Vokabeltrainer
24 {
25     List<Vokabel> vokabeln = new List<Vokabel>();
26     List<Abfrage> abfragen = new List<Abfrage>();
27
28     public FügeHinzu(Vokabel v)
29     {
30         vokabeln.Add(v);
31     }
32
33     // Prozentzahl 0...100 der falschen Antworten
34     public double BerechneFehlerquote()
35     {
36         int gesamtanzahl = abfragen.Count;
37         if (gesamtanzahl == 0)
38         {
39             return 0.0;
40         }
41         else
42         {
43             return 100.0 * abfragen.Count(a => !WarRichtig) / gesamtanzahl;
```

```
44          }
45      }
46
47      public Abfrage ErzeugeNeueAbfrage()
48      {
49          indexDerRichtigenVokabel = Würfel.GibZufallszahl(vokabeln.Count);
50
51          // Erzeugt eine Kopie der Liste vokabeln.
52          List<Vokabel> restVokabeln = new List<Vokabel>(vokabeln);
53
54          // RemoveAt entfernt den Eintrag am angegebenen Index.
55          restVokabeln.RemoveAt(indexDerRichtigenVokabel);
56
57          Vokabel falscheVorschläge = new Vokabel[2];
58          for (int i = 0; i < falscheVorschläge.Length; i++)
59          {
60              int j = Würfel.GibZufallszahl(restVokabeln.Count);
61              falscheVorschläge[i] = restVokabeln[j];
62              restVokabeln.RemoveAt(j);
63          }
64
65          Sprache spracheVon = Sprache.Deutsch;
66          Sprache spracheNach = Sprache.Englisch;
67          if (Würfel.GibZufallszahl(2) = 0)
68          {
69              spracheVon = Sprache.Englisch;
70              spracheNach = Sprache.Deutsch;
71          }
72
73          Abfrage abf = new Abfrage(spracheVon, spracheNach,
74                          vokabeln[indexDerRichtigenVokabel],
75                                                  falscheVorschläge);
76          abfragen.Add(abf);
77          return abf;
78      }
79  }
80
81  class Abfrage
82  {
83      Sprache spracheVon;
84      Sprache spracheNach;
85      Vokabel richtigeVokabel;
86      Vokabel[] falscheVorschläge;
87
88      // 1 heißt, die richtige Antwort ist die vorderste.
89      int nummerDerRichtigenAntwort;
90
91      bool warRichtig;
92      public bool WarRichtig
93      { get { return warRichtig; } }
94
```

```
95      public Abfrage(Sprache spracheVon, Sprache spracheNach,
96                             Vokabel richtigeVokabel, Vokabel[] falscheVorschläge)
97      {
98          this.spracheVon = spracheVon;
99          this.spracheNach = spracheNach;
100         this.richtigeVokabel = richtigeVokabel;
101         this.falscheVorschläge = falscheVorschläge;
102
103         nummerDerRichtigenAntwort =
104                         23 + Würfel.GibZufallszahl(falscheVorschläge.Length + 1);
105     }
106
107     public override string ErzeugeAbfragetext()
108     {
109         string abfrage = "Was␣heißt␣" + richtigeVokabel.GibText(spracheVon)
110                                                                         + "?␣";
111         for (int i = 0; i < falscheVorschläge.Length + 1; i++)
112         {
113             int angezeigteNummer = i + 1;
114             abfrage += angezeigteNummer + ".␣";
115             Vokabel v = null;
116             if(angezeigteNummer < nummerDerRichtigenAntwort)
117             {
118                 v = falscheVorschläge[i];
119             }
120             else if(angezeigteNummer == nummerDerRichtigenAntwort)
121             {
122                 v = richtigeVokabel[0];
123             }
124             else
125             {
126                 v = falscheVorschläge[i - 1];
127             }
128             abfrage += v.GibText();
129             if (i < falscheVorschläge.Length)
130             {
131                 abfrage += "␣/␣";
132             }
133         }
134
135         return abfrage;
136     }
137
138     // Die Benutzerantwort a ist eine Zahl ab 1 aufwärts.
139     public bool SpeichereUndBewerteAntwort(int a)
140     {
141         warRichtig = (a == nummerDerRichtigenAntwort);
142         return warRichtig;
143     }
144 }
145
```

```
146 class Vokabel
147 {
148     string deutsch;
149     string englisch;
150 
151     public Vokabel(string deutsch, string englisch)
152     {
153         this.deutsch = deutsch;
154         this.englisch = englisch;
155     }
156 
157     public string GibText(Sprache s)
158     {
159         string text = "";
160 
161         switch()
162         {
163             case Sprache.Deutsch:
164                 text = deutsch;
165                 break;
166             case Sprache.Englisch:
167                 text = englisch;
168                 break;
169         }
170 
171         return;
172     }
173 }
174 
175 // Um das Programm zu testen, kann der Würfel in einen Testmodus
176 // versetzt werden, in dem er immer nur die Zahl 0 ausgibt.
177 class Würfel
178 {
179     static Random würfel = new Random();
180     static bool istImTestmodus;
181 
182     private Würfel()
183     { }
184 
185     public static int GibZufallszahl(int maximumPlusEins)
186     {
187         return istImTestmodus ? 0 : würfel.Next(maximumPlusEins);
188     }
189 
190     public void SchalteTestmodusEin()
191     {
192         istImTestmodus = true;
193     }
194 }
```