Prof. Dr. Harald Brandenburg
Hochschule für Technik und Wirtschaft (HTW)
Fachbereich 4 (Wirtschaftswissenschaften II)
Wilhelminenhofstraße 75 A
12459 Berlin (Oberschöneweide)
Raum WH C 605

Fon: (030) 50 19 - 23 17
Fax: (030) 50 19 - 26 71
h.brandenburg@htw-berlin.de

Dienstag, 10. Juni 2014

Programmierung 2

SS 2014

Aufgabe 9: **Gruppe 1:** 10.06.2014 **Gruppe 2:** 24.06.2014

Schreiben und dokumentieren Sie ein objektorientiertes Java-Programm, das folgendes Spiel realisiert: [1]

- Per Zufallszahlengenerator wird eine 4-stellige positive ganze Zahl **x** erzeugt, die folgenden Bedingungen genügt:

  - Jede der Ziffern 0, 1, ..., 9 kommt in **x** höchstens einmal vor.
  - Eine führende Null ist erlaubt, d.h. 0 darf in **x** als erste Ziffer auftreten.

  Die Zahl soll wahlweise entweder von den Benutzern und Benutzerinnen des Programms oder vom Computer selbst nach der unten angegebenen Lösungsstrategie gefunden werden.

- Zur Unterstützung der Benutzer und Benutzerinnen soll das Programm vor jedem Rateversuch die vorhergehenden Rateversuche und ihr Ergebnis ausgeben. Das Ergebnis eines Rateversuchs **v** besteht aus der Angabe, wie viele der Ziffern von **v** sich an derselben Stelle befinden wie in **x**, und wie viele weitere Ziffern von **v** in **x** vorkommen, sich dort aber an anderer Stelle befinden.

```
1:      0123    an richtiger Stelle:  0    an falscher Stelle:   2
2:      2345    an richtiger Stelle:  2    an falscher Stelle:   0
3:      2367    an richtiger Stelle:  1    an falscher Stelle:   0
4:      2849    an richtiger Stelle:  2    an falscher Stelle:   0
5:      2347    an richtiger Stelle:  2    an falscher Stelle:   0
6:      2043    an richtiger Stelle:  1    an falscher Stelle:   1
7:      3145    an richtiger Stelle:  1    an falscher Stelle:   2
8:      2843    an richtiger Stelle:  1    an falscher Stelle:   1
9:      1346    an richtiger Stelle:  3    an falscher Stelle:   0

Ihr Tipp:
```

- Der Computer soll dagegen folgende Lösungsstrategie anwenden:

  Der Reihe nach, von **0001** bis **9999**, soll der nächste "gute" Kandidat für einen Rateversuch gefunden und als nächster Versuch zum Raten von **x** benutzt werden. Dabei ist ein Kandidat **k** für den n-ten Versuch $\mathbf{v}_n$ genau dann "gut", wenn er folgende Bedingungen erfüllt:

1 Es handelt sich um eine Variante von *Mastermind*.

- Jede der Ziffern 0, 1, ..., 9 kommt in **k** höchstens einmal vor.
- **Für jeden** vorangehenden Rateversuch v_i, **i = 1, ..., n-1**, ist das Ergebnis des Vergleichs von **k** mit v_i identisch mit dem Ergebnis des Vergleichs von v_i mit **x**.

- Das Programm soll ohne großen Aufwand so geändert werden können, dass es auch für **n**-stellige Ratezahlen mit **n > 4** funktioniert (z.B. auch für 6-stellige Ratezahlen).

```
1:      012345     an richtiger Stelle:  0     an falscher Stelle:   4
2:      103267     an richtiger Stelle:  1     an falscher Stelle:   2
3:      120489     an richtiger Stelle:  2     an falscher Stelle:   2
4:      124578     an richtiger Stelle:  2     an falscher Stelle:   1
5:      130958     an richtiger Stelle:  6     an falscher Stelle:   0

Mit 5 Versuchen geraten! Herzlichen Glueckwunsch!
```

- Das Programm soll so lange zur Verfügung stehen, wie es die Benutzer wünschen.

[**Hinweise:**

- Fehleingaben jeglicher Art dürfen nicht zum Absturz des Programms führen.
- Bei dieser Aufgabe kommt es vor allem auf den Entwurf angemessen komplexer Klassen an. Denken Sie daran, dass Klassen Attribute haben können, die Instanzen anderer Klassen sind.
- Beachten Sie insbesondere, dass für die oben angegebene Lösungsstrategie **alle** vorangehenden Rateversuche zur Verfügung stehen müssen.
- Auf den Rechnern des Labors sind (in dieser Reihenfolge) zu präsentieren:
  - die mit Hilfe von **javadoc** erzeugte (HTML-)Dokumentation,
  - die Java-Dateien,
  - die Übersetzung des Programms,
  - die Ausführung des Programms.
- Selbstverständlich darf Ihr Programm auch mehr leisten als gefordert.

]