LV 4342
Skriptsprachen-Praktikum
Übung 11 (Finale!)

Software-Entwicklung mit Ruby:

Unit Tests mit Test::Unit

Dokumentation mit RDoc

Organisatorisches

- Arbeitsverzeichnis:
  `~/lv/skriptspr/11/`

- Dateinamen:

```
11-lotto.rb      # Entwickeln und abgeben
11-lotto.tar.gz  # Verzeichnis incl. RDoc
sets.rb          # Beispiel-Code für Unit testing
```

- Werkzeuge:

```
ruby             # Der Interpreter
emacs            # mit Ruby-Mode. Auch X-Emacs ok
rdoc, irb, ri    # Neu: rdoc (!)
```

- Abzugeben:

```
11-lotto.rb
11-lotto.tar.gz
```

A: Vorbemerkungen

- Einige Anforderungen an professionelle Software
  - Fehlerfrei, auch nach Änderungen
  - Wartbar, auch in Teams
  - Leicht anpassbar an neue Anforderungen
- Konsequenzen
  - Test-Konzept, insb. „unit tests“
  - Dokumentation, die automatisch aktuell bleibt
  - Rückverfolgbarkeit aller Änderungen
  - Eine Sprache, die Änderungen erleichter (z.B. OO-Sprache)
- Ruby-Lösungen
  - Modul „**test/unit**“: Eingebaute Unterstützung für Unit-Tests
  - **RDoc**-System: Erzeugt HTML-Dokumentation aus Quelltexten
  - CVS, Subversion etc. (natürlich auch für Ruby nutzbar)
  - Ruby an sich - eine der besten „agilen“ Sprachen

A: *Unit testing*

- Teilschritte
  - Legen Sie Datei 11-lotto.rb durch Kopieren von 03-lotto.rb an
  - Entfernen Sie die Kommandozeilen-Anteile für den Dialog, sodass nur noch die Implementierung der Klasse Lotto übrig bleibt.
  - Ergänzen Sie nun Test-Code mittels Modul Test::Unit

- Hinweise & Vorgaben
  - Erzeugen Sie eine Testklasse „TestSet“ und darin für jede zu testende Methode eine eigene Test-Methode gleichen Namens!
  - Für konkrete Tests benötigen Sie bestimmte Eingabewerte. Wählen Sie (a) typische Werte (wie 6 und 49), (b) Werte im Grenzbereich des Zulässigen, (c) Werte knapp außerhalb der zulässigen Bereiche.
  - Nicht übertreiben: Je ein gut gewähltes Beispiel soll genügen…

- Testen Sie die Tests:
  - Verfälschen Sie die Implementierung einer Methode vorübergehend. Wird die Störung erkannt?

A: *Unit testing*

- **Tests für „initialize“ bzw. „new“:**
  - Lässt sich Lotto.new mit 0 und 1 Parameter aufrufen?
  - Erscheint ein Fehler, wenn mehr Parameter übergeben werden?
- **Tests für „draw_one“**
  - Erhalten Sie wirklich stets eine Zahl zwischen 1 und „max“ ?
  - Erhalten Sie jede Zahl wirklich nur einmal?? (Optional!)
- **Tests für „draw_all“**
  - Erhalten Sie ein Array aus n Zahlen zwischen 1 und „max“?
  - Testen Sie auch die Fälle n<0, n=0 und n>max!
- **Tests für „draw“ (auslassen!)**
  - Wie bei „draw_all“, aber hier müssen Sie das Rückgabe-Array erst selbst bilden.
- **Tests für „draw_sorted“ (auslassen!)**
  - Wie bei „draw“. Zusätzlich: Sind die Zahlen wirklich sortiert?

A: *Unit testing* – ein Beispiel

- Nutzen Sie die Datei „sets.rb“ als Beispiel-Code
  - Anlegen einer Klasse TestSet durch Ableiten von Test::Unit::TestCase genügt.

- Online-Dokumentation finden Sie hier:
  - Standard-Bibliothek, Einstiegsseite:
    http://www.ruby-doc.org/stdlib/libdoc/test/unit/rdoc/classes/Test/Unit.html
  - Standard-Bibliothek, *Assertions*:
    http://www.ruby-doc.org/stdlib/libdoc/test/unit/rdoc/classes/Test/Unit/Assertions.html

- Wer es hat: Pickaxe-Buch (2. Auflage!), Kap. 12

B: Dokumentation

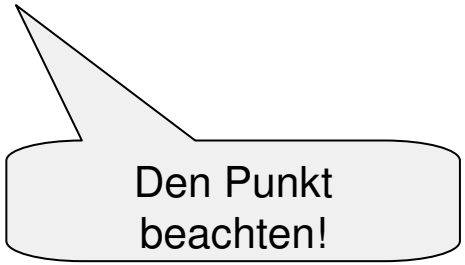
- Teilschritte
  - Fügen Sie Dokumentation als Kommentare in den Quellcode ein
  - Wenden Sie Kommando „**rdoc**“ (bzw. „rdoc1.8“) auf Ihre Datei an
  - Prüfen Sie den erzeugten HTML-Output auf Vollständigkeit, Verständlichkeit und korrektes Funktionieren Ihrer Formatierungen

- Hinweise & Vorgaben
  - Die Beschreibung für rdoc finden Sie hier:
    http://rdoc.sourceforge.net/doc/index.html
  - Lesen Sie insbesondere Abschnitte „Example“ und „Markup“!
  - Beschreiben Sie die Klasse „Lotto“ mit einem Kommentarblock
  - Fügen Sie Kommentar für jede Methode von „Lotto“ ein, achten Sie darauf, dass Eingabe-Parameter und Rückgabewerte klar beschrieben sind.
  - Verwenden Sie auch etwas „Markup“ zur Formatierung
  - Verhindern Sie die Erzeugung von Dokumentation zu den Test-Methoden!

Abgabe

- Geben Sie zunächst 11-lotto.rb separat ab
  - Die Unit-Tests sollten funktionieren (keine Syntaxfehler, alle „ok“)
  - Kommentare für RDoc sollten erhalten sein

- Erzeugen Sie dann ein tar-Archiv 11-lotto.tar.gz, das Ihr Arbeitsverzeichnis zu dieser Aufgabe enthält.
  - Das von rdoc erzeugte Unterverzeichnis „doc“ können Sie auf diese einfache Weise also ebenfalls abgeben
  - Tipp: `tar cvzf 11-lotto.tar.gz .`



- Anmerkungen
  - Machen Sie sich nicht zuviel Arbeit mit Unit-Tests: Wir wollen hier nur das Prinzip einüben, die **Betonung liegt auf RDoc**! Einige wenige assertions pro Fall genügen, um zu zeigen, dass Sie dies können.