AUFGABE 1: EINARBEITUNG IN DIE ENTWICKLUNGSUNMGEBUNG UND EINFACHES TESTEN

Die erste Aufgabe dient primär dem Kennenlernen des Build-Systems und der Entwicklungsumgebung. In dieser Aufgabe werden Sie eine sehr einfache Zustandsmaschine für eine Ampelschaltung implementieren und ihre Umsetzung mittels automatisierter Testfälle auf Fehler hin überprüfen. Außerdem sollen der Einsatz eines Debugger und die Entwicklung von Testfällen dazu verwendet werden, um Fehler in Fremd-Code (in einem Kommandozeilen-Parser) aufspüren.

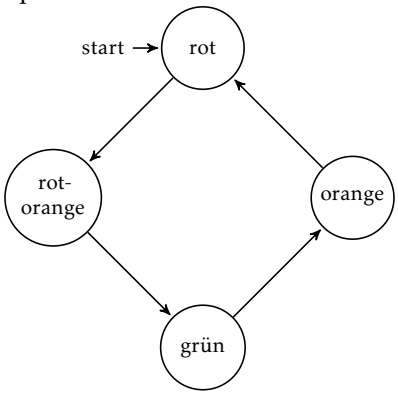
Aufgabenstellung

A. Verwenden Sie `git` um die Übungsvorgabe herunterzuladen:

```
git clone ssh://i4gerrit/vezs_ss13/vorgabe
```

B. Implementieren Sie die vorgegebene Schnittstelle und beachten Sie die Dokumentation der Schnittstelle bei der Implementierung. Die Details des grundlegenden Datentyps `TLight` sind bewusst offengelassen. Die Implementierung der Funktion `tlight_set_next_phase()` soll den Zustandsautomaten in folgender Abbildung repräsentieren:

☞ `make doxy`



C. Wie Sie dem Zustandsgraphen entnehmen können, besitzt die Zustandsmaschine vier Zustände und jeder dieser Zustände hat genau einen Folgezustand. *Wie können Sie erzwingen, dass Ihr*

Datentyp nur diese vier Zustände annehmen kann? Wie können Sie sicherstellen, dass nur legale Zustandsübergänge auftreten? Achten Sie bei Ihrer Implementierung auf Typsicherheit.

D. Entwerfen und implementieren Sie mindestens einen Testfall pro Funktion. Die Testfälle sollen wenigstens vollständige Codeüberdeckung erreichen. Überprüfen Sie die Überdeckung mit gcov bzw. `lcov`.

☞ `make lcov`

E. *Testen und Fehlersuche in Fremd-Code:* Betrachten Sie nun das vorgegebene Modul `cmdline`. Es soll die Aufgabe haben, eine Kommandozeile bestehend aus einem Programmaufruf und optionalen Schaltern (Flags) korrekt zu parsen. Allerdings haben sich bei der Implementierung einige Fehler eingeschlichen. Implementieren Sie Testfälle für die modulinternen Funktionen sowie für die Funktionen, die die Schnittstelle umsetzen sollen. Diese Testfälle sollen – wie in der vorheringen Teilaufgabe – mindestens eine vollständige Codeüberdeckung erreichen.
Zum Finden der Fehler können Sie einen Debugger (z. B. den in der Übung vorgestellten `gdb`) verwenden. Welche Fehler sind in dem Modul `cmdline` enthalten? Beheben Sie diese Fehler und testen Sie, ob Ihre Korrekturen zur erfolgreichen Ausführung Ihrer Testfälle führen.

Hinweise

- Erforderliche Dateien: wie vorgegeben
- Bearbeitung: Gruppenarbeit
- Abgabezeit: 16.05.2013
- Fragen bitte an `i4ezs@lists.cs.fau.de`