Prof. Dr. Stephan Kleuker
Fachhochschule Osnabrück
Fakultät Ing.-Wissenschaften und Informatik
- Software-Entwicklung -

Software Quality Management

Sommersemester 2022
1. Aufgabenblatt

Nutzen Sie die unter http://kleuker.iui.hs-osnabrueck.de/kleukersSEU/index.html zur Verfügung stehenden SEU, die ohne Gewähr zum Download zur Verfügung steht. Das Praktikum findet unter Windows 10 statt, Abgaben müssen mit der gegebenen SEU lauffähig sein. Einige ergänzende Informationen für diese Veranstaltung sind in http://kleuker.iui.hs-osnabrueck.de/querschnittlich/SQMWerkzeuge.pdf enthalten.

Aufgabe 1
In einem Teil der Veranstaltung wird die Programmiersprache Go als Beispiel genutzt, zum einen um generell die QS-Möglichkeiten zu analysieren und zum anderen um Beispielcode zu schreiben. Dazu sind grundlegende Kenntnisse in Go notwendig, die Sie sich zum nächsten Praktikum aneignen müssen.

Analysieren Sie dazu zumindest folgende Quellen, sinnvoll in dieser Reihenfolge.

1. https://www.youtube.com/watch?v=C8LgvuEBraI
2. https://www.youtube.com/watch?v=uBjoTxosSys (ab ca. 25:42 Ausblick; für Praktika wird trotzdem LiteIDE zum Einstieg empfohlen)
3. https://golang.org/doc/code.html
4. https://www.youtube.com/watch?v=LvgVSSpwND8

In der oben genannten SEU ist die Entwicklungsumgebung LiteIDE integriert, die über StartLiteIDE.bat aufgerufen wird. Der Aufruf setzt den GOPATH auf einen passenden Eintrag auf C:\Users\<Nutzer>\go. Sollten Sie generell einen anderen GOPATH nutzen wollen, setzen Sie diesen bevor Sie LiteIDE starten.
Zur ersten Nutzung der LiteIDE können Sie dieses Video http://kleuker.iui.hs-osnabrueck.de/Videos/SQM/SQMLiteIDE.mp4 herunterladen oder hier https://youtu.be/iRGTGG7RJD0 schauen.

a) Gegeben Sie folgendes Programm, das auch von der Veranstaltungsseite erhältlich ist. Bringen Sie das Programm zum Laufen und führen Sie es mehrfach aus. Schreiben Sie dann mit 3 Sätzen auf, wie der Algorithmus funktioniert.

```
package main

import (
        "fmt"
        "math/rand"
        "time"
)

var n = 30
var max = 20
var pos = 0
var zeitfaktor = 20
var ergebnis = make([]int, n)

func warte(val int) {
        time.Sleep(time.Duration(val*zeitfaktor) * time.Millisecond)
        schreibe(val)
}

func schreibe(val int) {
        ergebnis[pos] = val
        pos = pos + 1
}
```

Prof. Dr. Stephan Kleuker
Fachhochschule Osnabrück
Fakultät Ing.-Wissenschaften und Informatik
- Software-Entwicklung -

Software Quality Management

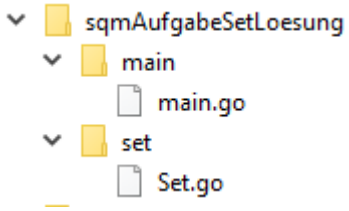
Sommersemester 2022
1. Aufgabenblatt

```
func main() {
        rand.Seed(time.Now().UnixNano())
        elemente := make([]int, n)
        for i := 0; i < n; i++ {
                elemente[i] = rand.Intn(max)
        }
        fmt.Println(elemente)
        for _, val := range elemente {
                go warte(val)
        }
        time.Sleep(time.Duration(max*zeitfaktor) * time.Millisecond)
        fmt.Println(ergebnis)
}
```

Erstellen Sie ein neues Go-Projekt (= Ordner) und realisieren Sie einen eigenen Typen Set für string-Werte auf der Basis von Slices.

b) es soll eine Funktion `New` zur Erzeugung eines neuen leeren string-Sets geben
c) es soll eine Methode `Add` geben, mit der ein string eingefügt werden kann, dies geschieht nur, wenn sich der string noch nicht im Set befindet; das Ergebnis gibt an, ob der Wert hinzugefügt wurde
d) es soll eine Methode `Remove` geben, mit der ein string gelöscht werden kann, dies geschieht nur, wenn sich der string im Set befindet; das Ergebnis gibt an, ob der Wert gelöscht wurde
e) es soll eine Methode `Has` geben, die einen string übergeben bekommt und als Ergebnis genau dann true liefert, wenn der string im Set enthalten ist
f) es soll eine Methode `Size` geben mit der die Anzahl der enthaltenen Elemente zurückgegeben wird
g) Schreiben sie in einem anderen Ordner eine main-Funktion mit der Sie Ihre Lösungen überprüfen können. Eine mögliche Projektstruktur ist rechts skizziert. Die main-Methode soll zumindest die folgenden Zeilen enthalten und die Ausgabe rechts liefern.

```
func main() {
    s := set.New()
    fmt.Println(s, s.Size(), s.Has("Fisch"))
    s.Add("Hai")
    s.Add("X")
    s.Add("Fisch")
    s.Add("X")
    fmt.Println(s, s.Size(), s.Has("Fisch"))
    s.Remove("X")
    s.Remove("Fisch")
    fmt.Println(s, s.Size(), s.Has("Fisch"))
}
```




```
{[]} 0 false
{[Hai X Fisch]} 3 true
{[Hai]} 1 false
```

Hinweis: Es gibt viele weitere Go-Informationen im Netz, recht kompakt und hilfreich sind z. B. https://golangbot.com/learn-golang-series/ und https://archium.org/index.php/Der_Golang-Spicker.