DHBW
Duale Hochschule
Baden-Württemberg
Mannheim

Abschlussklausur: Programmierung 1

(Lehrveranstaltung)

Fakultät für Wirtschaft

Studiengang: Wirtschaftsinformatik – Application Management

Datum: 12.12.2024

Matrikelnummer:

Dozent: Reiner Hüchting

Kurs: WWI24AMA **Semester:** 2

Hilfsmittel: Ein A4-Blatt **Bearbeitungszeit:** 120 Min.

Bewertung:

Note: Signum:

Anmerkungen:

Aufgabe		maximale Punkte	erreichte Punkte	Bemerkungen
Theorie				
1		5		
2		5		
3		10		
4		5		
5		5		
6		10		
Summe		**40**		
Labor				
1		10		
2		10		
3		10		
4		10		
5		10		
6		10		
Summe		**60**		

Aufgabe 1: Grundlagen (5 Punkte)

Welche der folgenden Behauptungen über die Programmiersprache Go sind wahr, welche falsch?

Behauptung	**wahr**	**falsch**
`void` ist ein Datentyp in Go.		
Bei der Definition von Variablen muss der Datentyp immer angegeben werden.		
Eine Funktionssignatur enthält immer den Namen der Funktion.		
Eigene Datentypen werden immer mit `type` definiert.		
Jede Schleife muss einen Zähler haben.		

Anmerkung: Korrekt angekreuzte Zeilen geben einen Punkt, für falsch angekreuzte Zeilen wird ein Punkt abgezogen.

Aufgabe 2: Signaturen (5 Punkte)

Betrachten Sie das folgende Programmfragment:

```
1   x1 := Foo1(1, 2)
2   x2 := Foo2(x1, "Hallo")
3   if Foo3(x2) {
4     Foo4(x1 < 42)
5   }
6   return Foo5(Foo2(x1, fmt.Sprintf("%t", x2)) && x2)
```

Welche Signaturen haben die Funktionen `Foo1` bis `Foo5`?

Anmerkung: Die *Signatur* einer Funktion ist die erste Zeile, in der die Argument- und Rückgabetypen definiert werden. Hier ist also gefragt, welche Typen die Funktionen erwarten und liefern. Sie können davon ausgehen, dass Funktionen, deren Ergebnis nicht verwendet wird, auch keinen Rückgabetyp haben.

Aufgabe 3: Fehlersuche: Compilerfehler (10 Punkte)

Der folgende Code enthält eine Reihe an Fehlern, durch die er nicht compiliert. Markieren Sie alle Zeilen, die einen Fehler enthalten und erläutern Sie kurz, was jeweils falsch ist.

```
1  package fehlersuche1
2
3  import "fmt"
4
5  func Foo(x int) string {
6    return x
7  }
8
9  func Bar(x, y int) string {
10   for i := 0; i < x; i++ {
11     y += i
12   }
13   return Foo(y), fmt.Sprint(x)
14 }
15
16 func FooBar[] {
17   s := "Huhu"
18   for x := range []int{10, 20, 30, 40, 50} {
19     s += Bar(x, x+s)
20   }
21   s += "y"
22   fmt.println(s)
23 }
```

Hinweis: Es geht hier nur um Syntaxfehler. Für falsch markierte Zeilen gibt es Punktabzug!

Aufgabe 4: Fehlersuche: Inhaltliche Fehler (5 Punkte)

Die folgende Funktion ist zwar syntaktisch korrekt, sie erfüllt aber nicht ihre Aufgabe. Erläutern Sie den/die Fehler und machen Sie einen Vorschlag zur Korrektur.

```
1 // Sorted liefert true, falls die Liste aufsteigend sortiert ist.
2 func Sorted(list []int) bool {
3   for i := range list[1:] {
4     if i > list[i] {
5       return false
6     }
7   }
8   return true
9 }
```

Anmerkung: Ihre Korrektur muss nicht syntaktisch korrekt sein. Eine Erklärung in Worten genügt.

Aufgabe 5: Programmverständnis (5 Punkte)

Erläutern Sie, was die Funktion `Foo` im folgenden Programmfragment berechnet. Geben Sie eine möglichst allgemeine bzw. abstrakte Erklärung an.

```
1  func Foo(n int) bool {
2    for i := 1; i < n; i++ {
3      for j := 1; j < n; j++ {
4        if i*i+j*j == n {
5          return true
6        }
7      }
8    }
9    return false
10 }
```

Aufgabe 6: Rekursion (10 Punkte)

Betrachten Sie die folgende Funktion:

```
1  func Foo(s1, s2 string) int {
2    if len(s1) == 0 || len(s2) < len(s1) {
3      return 0
4    }
5    if s2 == s1+s2[len(s1):] {
6      return 1 + Foo(s1, s2[1:])
7    }
8    return Foo(s1, s2[1:])
9
10 }
```

Beschreiben Sie in Worten, was die Funktion berechnet.
Hinweis: Falls Sie die Lösung nicht sehen, berechnen Sie beispielhaft Werte, z.B. für:

- `Foo("ab", "abab")`
- `Foo("ab", "aba")`
- `Foo("a", "aba")`