Aufgabe 1

Betrachten Sie die folgende Funktion:

```
1  func Foo(list []int, x int) ([]int, []int) {
2      r1 := make([]int, 0)
3      r2 := make([]int, 0)
4
5      for _, v := range list {
6          if v <= x {
7              r1 = append(r1, v)
8          } else {
9              r2 = append(r2, v)
10         }
11     }
12
13     return r1, r2
14 }
```

Erläutern Sie möglichst allgemein/abstrakt, was die Funktion berechnet.

Die Funktion zerlegt die Liste in zwei Teillisten. Die erste Teilliste enthält alle Elemente, die kleiner oder gleich `x` sind. Die zweite Teilliste enthält alle Elemente, die größer als `x` sind.

Aufgabe 2

Die folgende Funktion sollte in einer Liste alle Elemente verdoppeln. Sie compiliert nicht und würde so auch nicht richtig funktionieren. Erläutern Sie die Fehler.

```
1 func DoubleAll([]int list) {
2     for i,v := range l {
3         l[i] * 2
4     }
5 }
```

Lösung zu Aufgabe 2

Hier ist eine korrekte Version der Funktion inkl. Kommentaren zu den Fehlern.

```
1 func DoubleAll(list []int) { // Erst Name, dann Typ
2     for i := range list { // Die liste heißt list, nicht l; v
          wurde nicht benutzt
3         list[i] *= 2 // *= statt =, sonst hat die Anweisung
              keinen Effekt.
4     }
5 }
```

Aufgabe 3

Betrachten Sie die folgende Funktion:

```
1 func Foo1(n, m int) int {
2     if n >= m {
3         return Foo1(n-m, m)
4     }
5     if n < 0 {
6         return Foo1(n+m, m)
7     }
8     return n
9 }
```

Erläutern Sie möglichst allgemein/abstrakt, was die Funktion berechnet.

Die Funktion berechnet den Modulo-Operato `n % m` rekursiv.

Aufgabe 4

Betrachten Sie die folgende Funktion:

```
1 func Foo2(n, m int) int {
2     if m == 0 {
3         return 1
4     }
5     return n * Foo2(n, m-1)
6 }
```

Erläutern Sie möglichst allgemein/abstrakt, was die Funktion berechnet.

Die Funktion berechnet die Potenz n^m rekursiv.

Aufgabe 5

Im folgenden wird ein Struct `Person` mit einer Methode `SetBirthday()` definiert. Der Aufruf `p1.SetBirthday()` in `DoFoo()` sollte dazu führen, dass anschließend Tag, Monat und Jahr korrekt in `p1` eingetragen sind. Das funktioniert aber nicht. Erläutern Sie den Fehler.

```
1  type Person struct {
2      name              string
3      day, month, year int
4  }
5
6  func (p Person) SetBirthday(day, month, year int) {
7      p.day, p.month, p.year = day, month, year
8  }
9
10 func DoFoo() {
11     p1 := Person{name: "Donald Knuth"}
12     p1.SetBirthday(10, 1, 1938)
13
14     fmt.Println(p1)
15 }
```

Lösung zu Aufgabe 5

Der Fehler ist, dass die Methode `SetBirthday()` keinen Pointer-Receiver hat. Dadurch wird auf eine Kopie der Person zugegriffen, die in `DoFoo()` angelegt wurde.