Exercise 2: Feature Composition

Software Engineering Design & Construction
WS 2016/17 - Dr. Michael Eichberg, M.Sc. Matthias Eichholz

The second task of this exercise will be graded. Please submit your solution until the **22nd of November, 23:59** via email to eichholz@st.informatik.tu-darmstadt.de. Make sure you zip your complete sbt project and make sure that it works out of the box by running `sbt run` and `sbt test`.

Although the other exercises are not graded, it is highly recommended to also do them on your own. Just looking at a solution is much easier in comparison to actually coming up with it. Support can be found in the forum:
`https://www.fachschaft.informatik.tu-darmstadt.de/forum/viewforum.php?f=234`

Task 1 Functional Sets

In the last exercise, we used binary search trees to represent sets. Another way to represent sets is via a single function `Set => Boolean` that defines whether a given element is in the set. In Scala, we can define a type alias, so that we can use the alias `Set` instead of the longer function type:

```
type Set = Int => Boolean
```

We can write a `contains` function as follows:

```
def contains(s: Set, elem: Int): Boolean = s(elem)
```

Since a set of type `Set` is just a function, `contains` simply applies that function to determine whether the element is in the set or not.

Task 1.1 Implementation

Implement a constructor, with the following signature, that creates a set containing a single element:

```
def Set(elem: Int): Set
```

Implement set union, intersection and difference with functions of the following signatures:

```
def union(s: Set, t: Set): Set
def intersect(s: Set, t: Set): Set
def diff(s: Set, t: Set): Set
```

Implement a filter function for our functional set representation similar to the filter method above:

```
def filter(s: Set, p: Int => Boolean): Set
```

Implement a function that maps every element in a given set S using a function f, so that the result is the set $\{f(x) \mid x \in S\}$:

```
def map(s: Set, f: Int => Int): Set
```

You can assume that for all elements $x \in S$: $-1000 \leq x \leq 1000$.

Task 2 (graded) + 3

Task 2 and 3 will be released later this week.