Prof. Dr. A. Poetzsch-Heffter
M.Sc. Peter Zeller
Dipl.-Inf. C. Feller

University of Kaiserslautern
Department of Computer Science
Software Technology Group

Exercise Sheet 12: Specification and Verification with Higher-Order Logic (Summer Term 2014)

Exercise 1 Heap-manipulating OO programs

In this exercise we will use the calculus from the lecture to do some pen- and paper exercises about the following JavaKE program:

```
1  interface Cell {
2          int set(int val);
3          int get();
4  }
5
6
7  class StandardCell implements Cell {
8          int x;
9          int set(int par) {
10                 x = par;
11         }
12         int get() {
13                 res = x;
14         }
15 }
16
17
18
19
20
```

```
21 class PrevCell implements Cell {
22         boolean f;
23         int x1;
24         int x2;
25         int set(int par) {
26                 boolean t;
27                 t = f;
28                 f = !t;
29                 if (!t) x1 = par;
30                 else x2 = par;
31         }
32         int get(){
33                 if (f) res = x1;
34                 else res = x2;
35         }
36         int getPrev(){
37                 if (f) res = x2;
38                 else res = x1;
39         }
40 }
```

a) Give a weak precondition P so that the following triple is valid in the context of the above program:

{ P }
    c1.set(4);
    c2.set(2);
    int x = c1.get();
    int y = c2.get();
    res = x∗10 + y;
{ res = 42 }

b) Define a heap abstraction cell(e,x) which states that reference "e" points to a cell object with a current value of "x".

c) Prove the following statement on paper, using the rules from the lecture:

{c ≠ **null**} r = c.set(x) {cell(c,x)}

You can assume that the static type of `c` in this program part is `Cell`.
For every step in the proof, write down the name of the used rule.
Try to get to the following intermediate goal first:

{c ≠ **null** ∧ c = C ∧ x = X} r = c.set(x) {cell(C,X)}