OTE: Ohjelmointitekniikka
Programming Techniques

course homepage: `http://www.cs.uku.fi/~mnykanen/OTE/`

Week 48/2008

Exercise 1.

(a) What is the wp semantics of the *empty* **if fi** command which has no branches? Why?

(b) What about the empty **do od** loop?

Exercise 2. The exponentiation algorithm in Figure 10 has an annoying design wrinkle: When p is odd, it subtracts 1 from p, making p even. Then it tests whether p is even or odd, even though this is already known.

(a) Rewrite it to remove this wrinkle.

(b) Prove that your algorithm in part (a) is also correct. What parts of the proof for the original algorithm can you recycle?

Exercise 3. Could the *bound* in Theorem 14 be real-valued instead? Why?

Exercise 4. Suppose that while verifying checkpoint 2 for a **do** $\mathcal{B}$ **od** loop you managed to prove

$$\mathit{invariant} \land \mathit{guard} \Longrightarrow \text{FALSE}$$

for one of its branches. What does this mean?

Exercise 5.

(a) How would you express in logic that u is the largest number found in the one-dimensional array a?

(b) Write an algorithm in GCL to find this u from a.

(c) Prove your algorithm correct.

Exercise 6. Consider the following code:

```
{ $\text{lower}(b) = \text{lower}(a) \land \text{upper}(b) = \text{upper}(a)$ }
$s\,,\ i\ :=\ 0\,,\ \text{lower}(a)$;
do $i \leq \text{upper}(a) \rightarrow$
    $b[i]\ :=\ s + a[i]$;
    $s\,,\ i\ :=\ b[i]\,,\ i+1$;
od;
{ $\forall i \in \text{indices}(a).b[i] = \sum_{j=\text{lower}(a)}^{i} a[j]$ }
```

(a) What do its pre- and postconditions claim that it computes?

(b) The programmer (the lazy sod. . .) has omitted its loop invariant and bound. Add them.

(c) If you think that this loop is correct, then prove it. If not, then give an input a which reveals a bug in it.