

# Final Exam

Semantics

14.02.2020

First name: \_\_\_\_\_

Last name: \_\_\_\_\_

Student-Id (Matrikelnummer): \_\_\_\_\_

Signature: \_\_\_\_\_

1. You may only use a pen/pencil, eraser, and one A4 sheet of notes to solve the exam. Switch off your mobile phones!
2. Please use the empty pages to answer the questions, preferably. If this space is not sufficient, you may use the extra sheets that have been handed out. Please state it clearly, below the question, when doing so.
3. You have 120 minutes to solve the exam.
4. Please put your student ID and ID-card or driver's license on the table until we have checked it.
5. Please do not leave the room in the last 15 minutes of the exam.

**Proof Guidelines:** We expect detailed, rigorous, mathematical proofs — but we do not ask you to write Isabelle proof scripts! You are welcome to use standard mathematical notation; you do not need to follow Isabelle syntax. Proof steps should be explained in ordinary language like a typical mathematical proof.

Major proof steps, especially inductions, need to be stated explicitly. For each case of a proof by induction, you must list the **inductive hypotheses** assumed (if any), and the **goal** to be proved.

Minor proof steps (corresponding to `by simp`, `by blast` etc) need not be justified if you think they are obvious, but you should say which facts they follow from. You should be explicit whenever you use a function definition or an introduction rule for an inductive predicate — especially for functions and predicates that are specific to an exam question. (You need not reference individual lemmas for standard concepts like integer arithmetic, however, and in any case we do not ask you to recall lemma names from any Isabelle theories.)

# 1 Induction

**Question 1** Consider the following inductive definition:

```
inductive M :: "nat => bool" where
  "M (n+2) ==> M n" |
  "M n ==> M (n+5) ==> M (n+1)"
```

a) Write down the Isabelle proof state after the following two commands:

```
lemma "M a ==> a = 7"
  apply(induction a rule: M.induct)
```

b) Is the formula  $M\ a \implies \text{False}$  provable? Give a short justification of your answer.

**Question 2** Consider the following function definitions:

```
fun f :: "nat => nat" where
  "f 0 = 0" |
  "f (Suc 0) = 1" |
  "f (Suc(Suc n)) = f n + f (Suc n)"
fun g :: "nat => nat => nat => nat" where
  "g a b 0 = b" |
  "g a b (Suc n) = g b (a+b) n"
```

Find terms ?1 and ?2 such that the equation

$$g\ a\ b\ n = f\ n * ?1 + f\ (Suc\ n) * ?2$$

becomes true. Give a proof of the resulting equation.



## 2 While-Loops

Consider the the big-step semantics of IMP. We will denote the set of variables of a command  $c$  by  $vars\ c$ .

**Question 1** Show:

$$(WHILE\ b\ DO\ c,\ s) \Rightarrow t \implies vars\ c \cap vars\ b = \{\} \implies \neg\ bval\ b\ s$$

*Hint:* You may use the following fact:

$$vars\ c \cap vars\ b = \{\} \implies (c,\ s) \Rightarrow t \implies bval\ b\ s \iff bval\ b\ t \quad (1)$$

**Question 2** Define a function  $no :: com \Rightarrow bool$  such that  $no\ c$  holds if and only if  $c$  contains no while loops. Show:

$$no\ c \implies \forall s. \exists t. (c,\ s) \Rightarrow t$$

*Hint:* You may skip the cases for *SKIP* and assignment when performing an induction.



### 3 Hoare-Logic

We replace the assignment in IMP by a command  $REL\ R$  that performs an arbitrary state transition according to relation  $R :: (state \times state)\ set$ .

In the big-step semantics, we remove the *assign*-rule, and add the following rule:

$$Rel: (s, s') \in R \implies (REL\ R, s) \Rightarrow s'$$

#### Questions

1. Is the semantics deterministic, i.e., does the following hold (proof or counterexample):

$$(c, s) \Rightarrow t \implies (c, s) \Rightarrow t' \implies t=t'$$

2. What does the weakest precondition  $wp\ (REL\ R)\ Q$  look like?
3. Specify a Hoare-rule for  $REL$ .
4. Prove:  $\vdash \{wp\ (REL\ R)\ Q\}\ REL\ R\ \{Q\}$ .

#### Hints

- Question 2: Recall the definition of the weakest precondition:

$$wp\ c\ Q = (\lambda s. \forall t. (c, s) \Rightarrow t \longrightarrow Q\ t)$$

Now we want an equation that shows how to expand  $wp$  syntactically, i.e., the right hand side should not contain the Big/Small-step semantics. You need **not** prove your equation here.

- Question 4: The main lemma in the completeness proof of Hoare logic is  $\vdash \{wp\ c\ Q\}\ c\ \{Q\}$ . Here you have to prove the case for the  $REL$ -command. Use your equation for  $wp$  from Question 2 here!









## 5 Post-fixed Points

Recall that a complete lattice is a type  $'a$  with a partial order  $\leq$  such that every set  $X :: 'a \text{ set}$  has a greatest lower bound, denoted  $\sqcap X$ . This means that  $\forall x \in X. \sqcap X \leq x$  and  $\forall y. ((\forall x \in X. y \leq x) \longrightarrow y \leq \sqcap X)$ .

Prove that for a complete lattice and a monotone function  $f :: 'a \Rightarrow 'a$  on it, the set of post-fixed points of  $f$  is closed under  $\sqcap$ :

$\forall X :: 'a \text{ set}. ((\forall x \in X. f x \leq x) \longrightarrow f (\sqcap X) \leq \sqcap X)$

