Last Time

- The automated proof method \texttt{wp}
- The C Parser and translating C into Simpl
- AutoCorres and translating Simpl into monadic form
- The option and exception monads
Exam

➔ 24h take-home exam (same as previous years)
Exam

→ 24h take-home exam (same as previous years)

→ Open book: can use any passive resource (books, slides, google, etc)
→ Not allowed to ask for help from anyone
→ starts 8am AEST, Wed 2 Dec 2020, ends 7:59am AEST, Thu 3 Dec 2020

Should be doable in about 4-6 hours. The 24h are for flexibility not for you to stay awake actual 24h.

Recommend to start early, finish the easy questions first.
Take breaks. Don’t forget to eat :-)
If there are clarification questions email the lecturers.
Exam

→ 24h take-home exam (same as previous years)

→ Open book: can use any passive resource (books, slides, google, etc)
→ **Not** allowed to ask for help from anyone
→ starts 8am AEST, Wed 2 Dec 2020, ends 7:59am AEST, Thu 3 Dec 2020

→ Should be doable in about 4-6 hours.
   The 24h are for flexibility not for you to stay awake actual 24h.
→ Recommend to start early, finish the easy questions first.
→ Take breaks. Don’t forget to eat :-)
→ If there are clarification questions email the lecturers.
Content

→ Foundations & Principles
  • Intro, Lambda calculus, natural deduction [1,2]
  • Higher Order Logic, Isar (part 1) [2,3a]
  • Term rewriting [3,4]

→ Proof & Specification Techniques
  • Inductively defined sets, rule induction, datatype induction, primitive recursion [4,5]
  • General recursive functions, termination proofs [7b]
  • Proof automation, Hoare logic, proofs about programs, invariants [8]
  • C verification [9,10]
  • Practice, questions, examp prep [10c]

a1 due; a2 due; a3 due
COMP4161: Advanced Topics in Software Verification

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have learned so far...

- $\lambda$ calculus syntax
- free variables, substitution
- $\beta$ reduction
- $\alpha$ and $\eta$ conversion
- $\beta$ reduction is confluent
- $\lambda$ calculus is very expressive (turing complete)
- $\lambda$ calculus results in an inconsistent logic
COMP4161: Advanced Topics in Software Verification

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have learned so far...

- Simply typed lambda calculus: \( \lambda \rightarrow \)
- Typing rules for \( \lambda \rightarrow \), type variables, type contexts
- \( \beta \)-reduction in \( \lambda \rightarrow \) satisfies subject reduction
- \( \beta \)-reduction in \( \lambda \rightarrow \) always terminates
- Types and terms in Isabelle
COMP4161: Advanced Topics in Software Verification

\[ \lambda \rightarrow \] and HOL

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
What we have learned so far...

- natural deduction rules for $\land$, $\lor$, $\rightarrow$, $\neg$, iff...
- proof by assumption, by intro rule, elim rule
- safe and unsafe rules
- indent your proofs! (one space per subgoal)
- prefer implicit backtracking (chaining) or rule_tac, instead of back
- prefer and defer
- oops and sorry
We have learned so far...

- Isar style proofs
- proof, qed
- assumes, shows
- fix, obtain
- moreover, ultimately
- forward, backward
- mixing proof styles
We have learned today ...

- Defining HOL
- Higher Order Abstract Syntax
- Deriving proof rules
- More automation
- Equations and Term Rewriting
COMP4161: Advanced Topics in Software Verification

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have seen today...

→ Equations and Term Rewriting
→ Confluence and Termination of reduction systems
→ Term Rewriting in Isabelle
COMP4161: Advanced Topics in Software Verification

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have learned today ...

→ Conditional term rewriting
→ Congruence rules
→ AC rules
→ More on confluence
COMP4161: Advanced Topics in Software Verification

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have learned today ... 

- Sets
- Type Definitions
- Inductive Definitions
COMP4161: Advanced Topics in Software Verification

{ }

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have learned today ... 

- Formal background of inductive definitions
- Definition by intersection
- Computation by iteration
- Formalisation in Isabelle
We have seen today ...

- Datatypes
- Primitive recursion
- Case distinction
- Structural Induction
COMP4161: Advanced Topics in Software Verification

fun

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have seen today ...

- General recursion with *fun*/function
- Induction over recursive functions
- How *fun* works
- Termination, partial functions, congruence rules
We have seen today ...

→ sledgehammer
→ nitpick
→ quickcheck
COMP4161: Advanced Topics in Software Verification

\{P\} \ldots \{Q\}

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have seen today ...

- Syntax of a simple imperative language
- Operational semantics
- Program proof on operational semantics
- Hoare logic rules
- Soundness of Hoare logic
COMP4161: Advanced Topics in Software Verification

\{ \text{P} \} \ldots \{ \text{Q} \}

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have seen today ...

- Weakest precondition
- Verification conditions
- Example program proofs
- Arrays, pointers
COMP4161: Advanced Topics in Software Verification

Gerwin Klein, Johannes Åman Pohjola, Christine Rizkallah, Miki Tanaka
T3/2020
We have seen today

- Deep and shallow embeddings
- Isabelle records
- Nondeterministic State Monad with Failure
- Monadic Weakest Precondition Rules
Today we have seen

- The automated proof method $wp$
- The C Parser and translating C into Simpl
- AutoCorres and translating Simpl into monadic form
- The option and exception monads