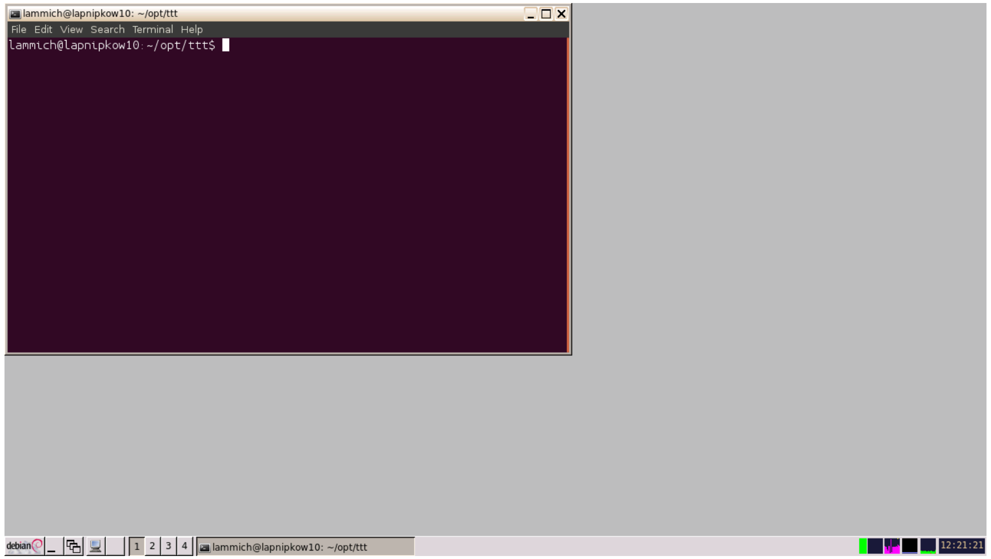
Script **generated by TTT**

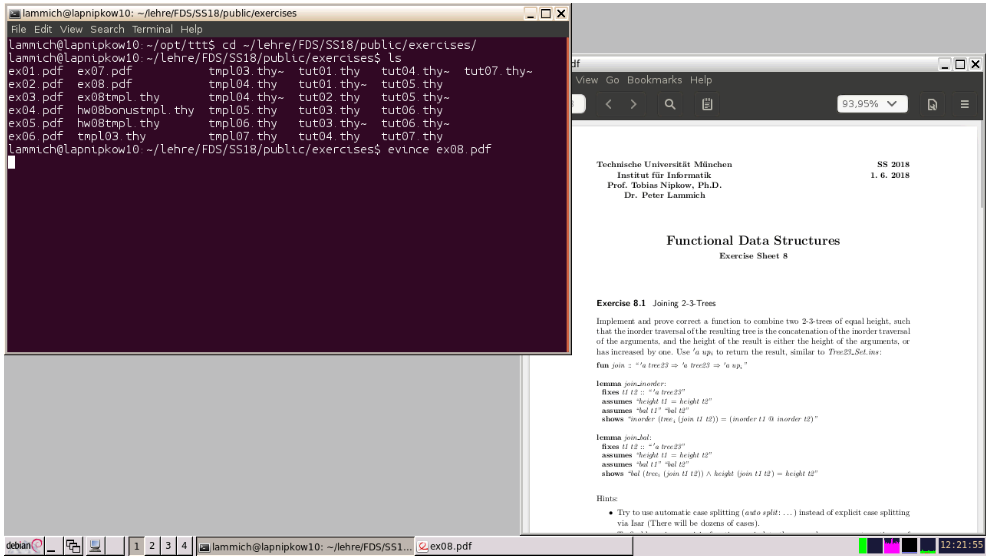
Title: Lammich: FDS Tutorial (01.06.2018)

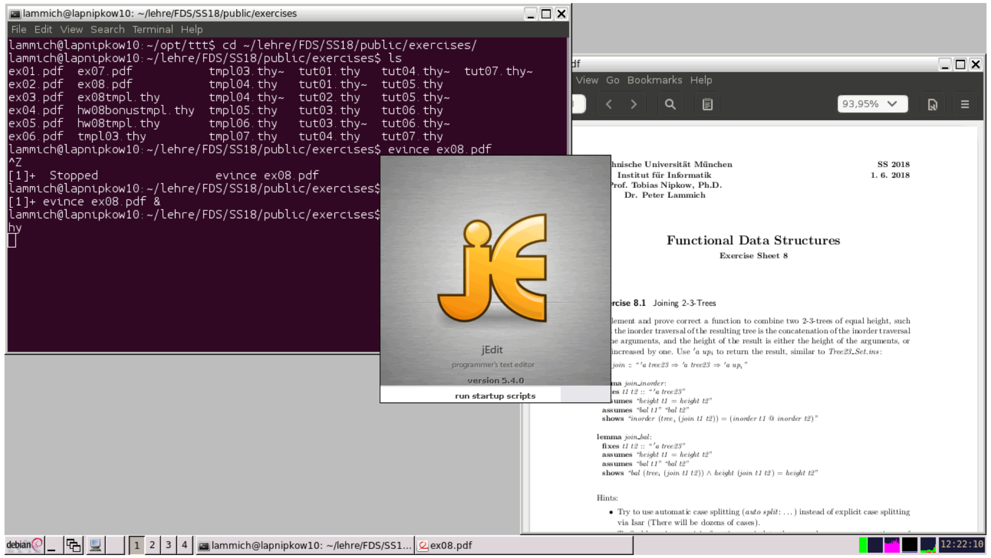
Date: Fri Jun 01 12:21:15 CEST 2018

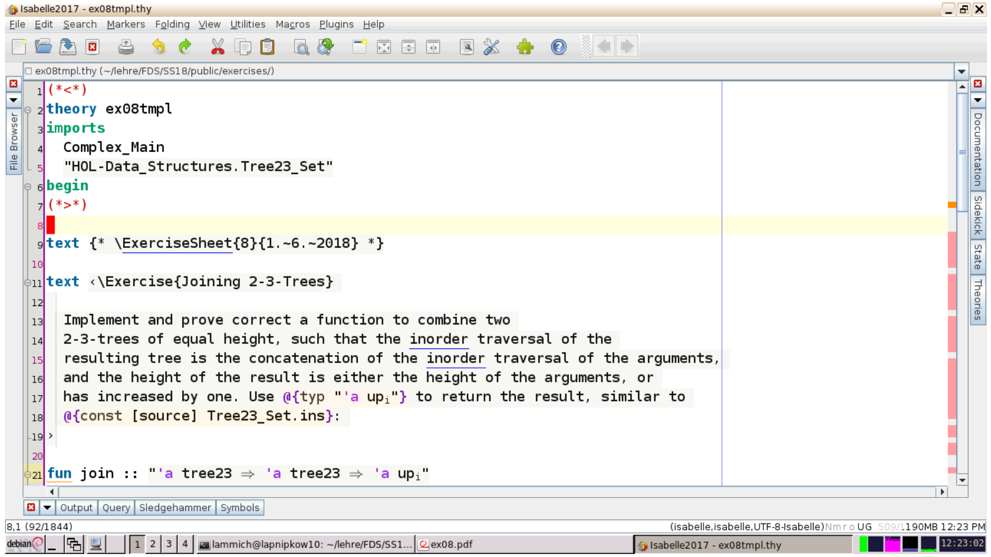
Duration: 82:50 min

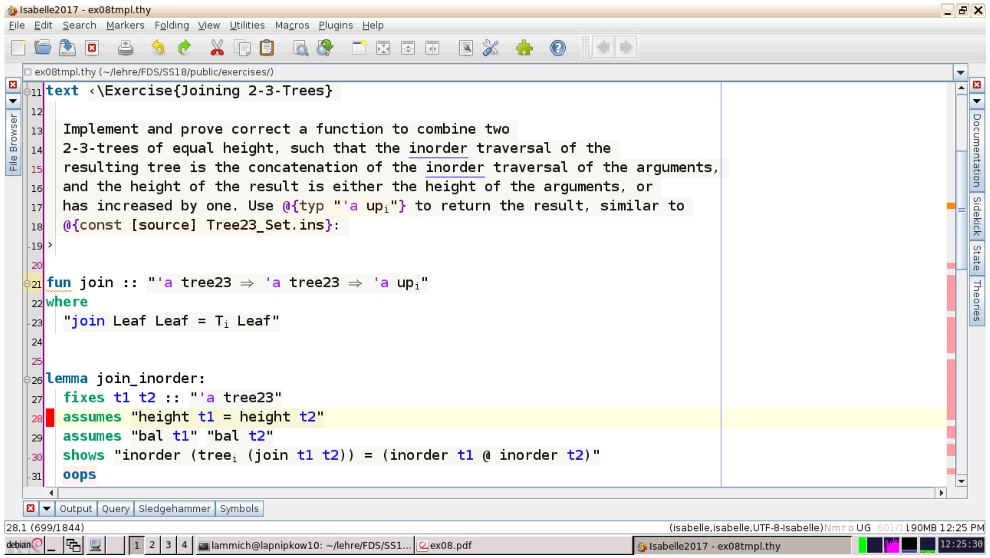
Pages: 84

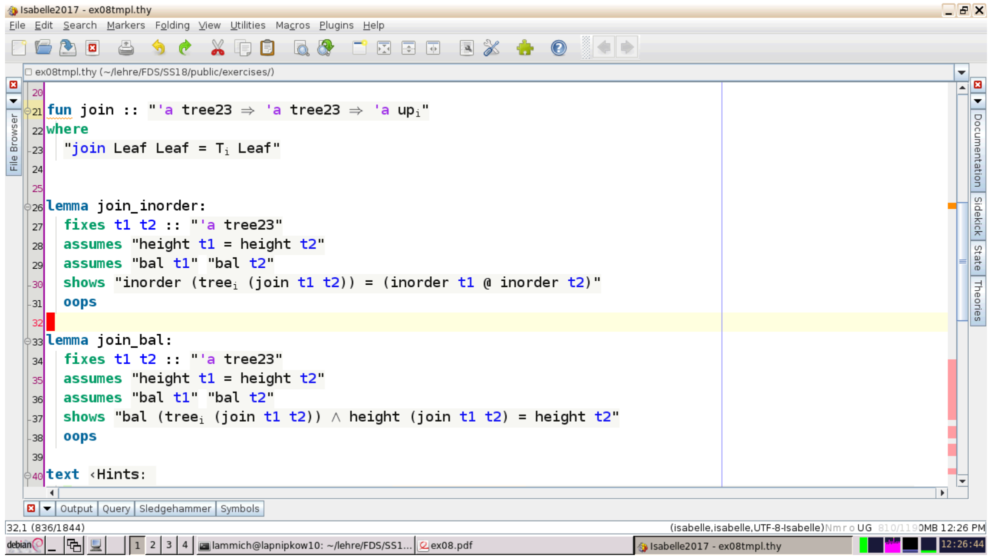


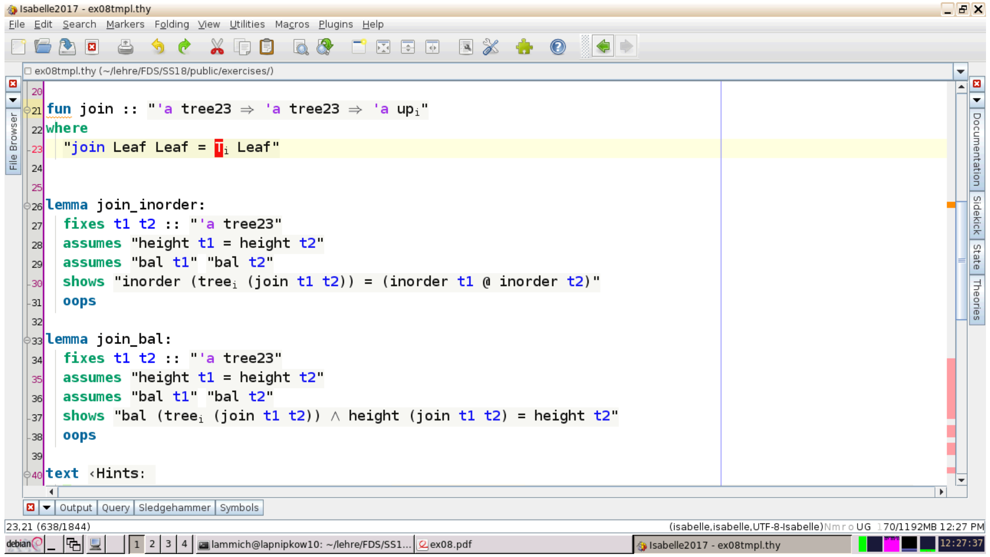
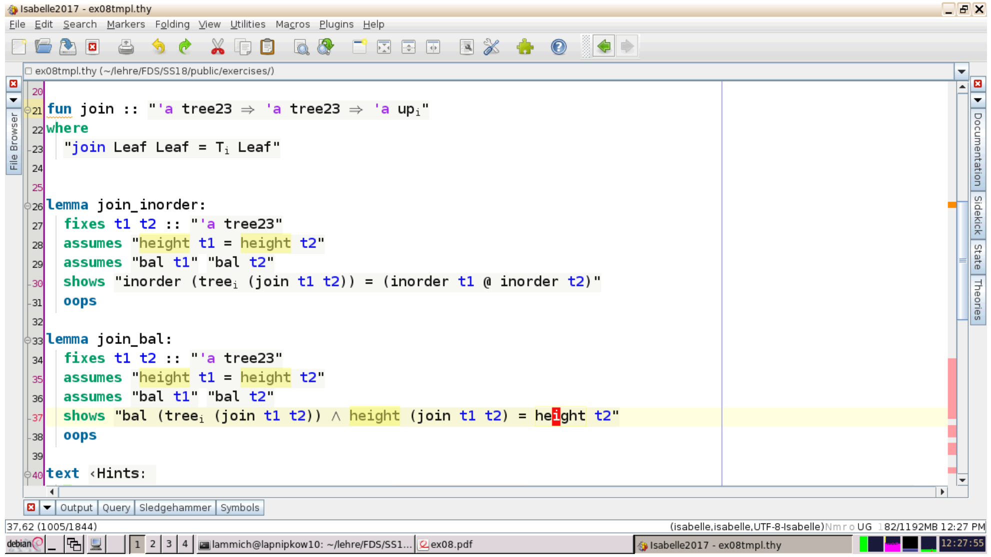
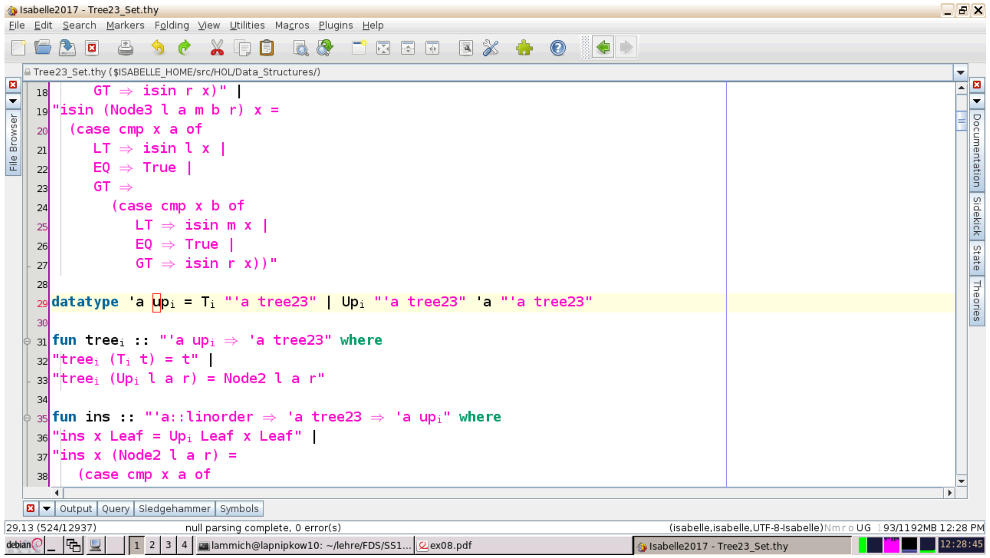
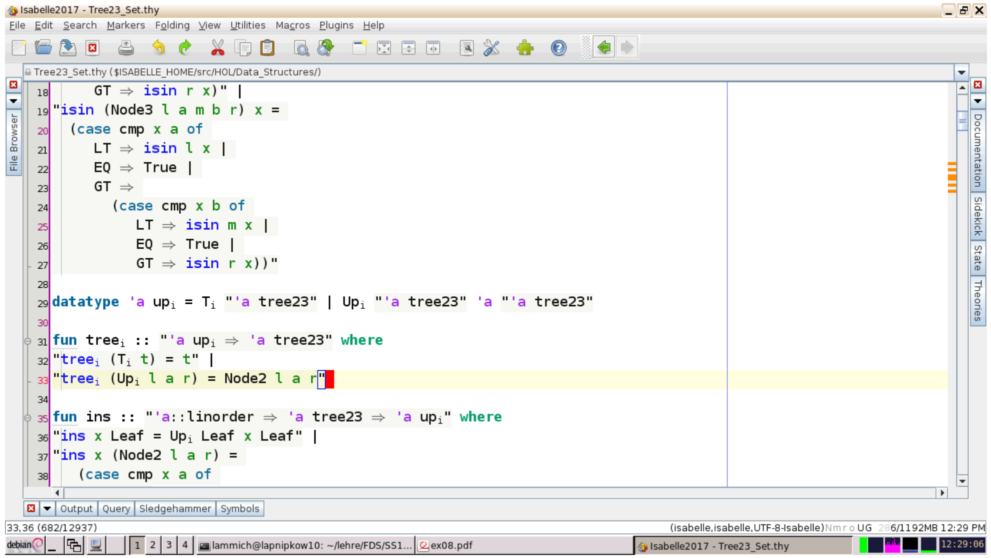


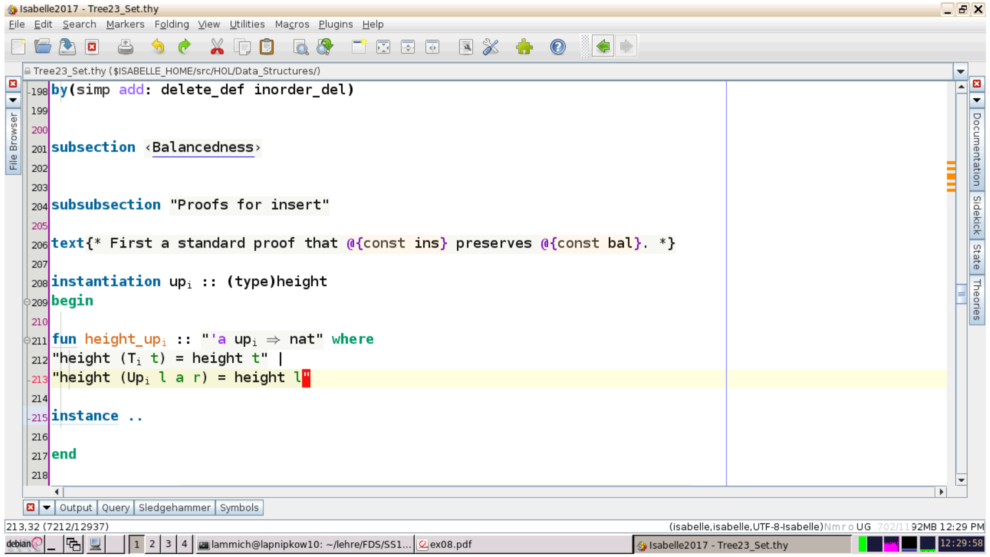

lammich@lapnipkow10: ~/lehre/FDS/SS18/public/exercises
File Edit View Search Terminal Help
lammich@lapnipkow10:~/opt/ttt$ cd ~/lehre/FDS/SS18/public/exercises/
lammich@lapnipkow10:~/lehre/FDS/SS18/public/exercises$ ls
lammich@lapnipkow10:~/lehre/FDS/SS18/public/exercises$ evince ex08.pdf
[1]+ Stopped evince ex08.pdf
[1]+ evince ex08.pdf &
jEdit
version 5.4.0
run startup scripts
Functional Data Structures
Exercise Sheet 8
SS 2018
1. 6. 2018

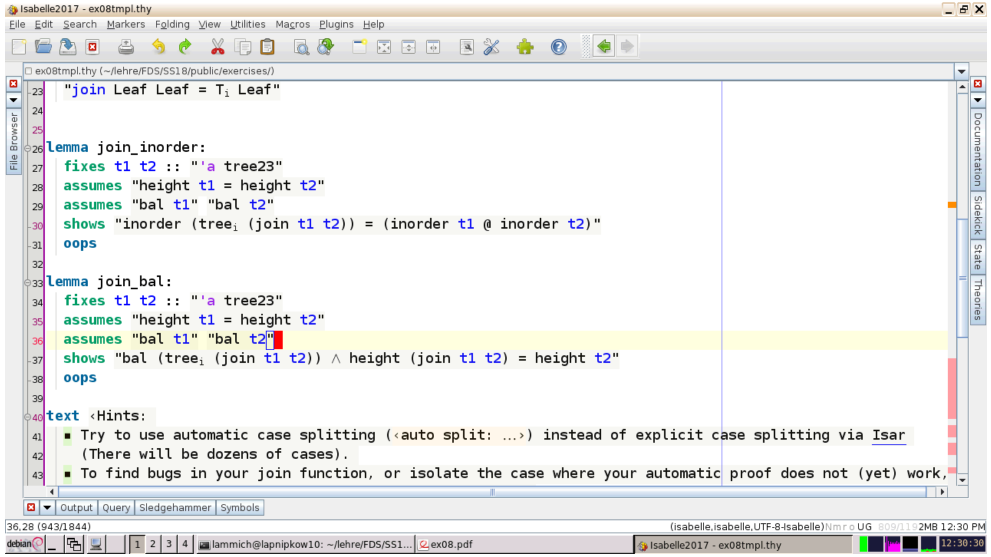

Isabelle2017 - ex08tmpl.thy
(*<*)
theory ex08tmpl
imports
Complex_Main
"HOL-Data_Structures.Tree23_Set"
begin
(*>*)
text {* \ExerciseSheet{8}{1.~6.~2018} *}
text ‹\Exercise{Joining 2-3-Trees}
Implement and prove correct a function to combine two
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a up₁"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a up₁"

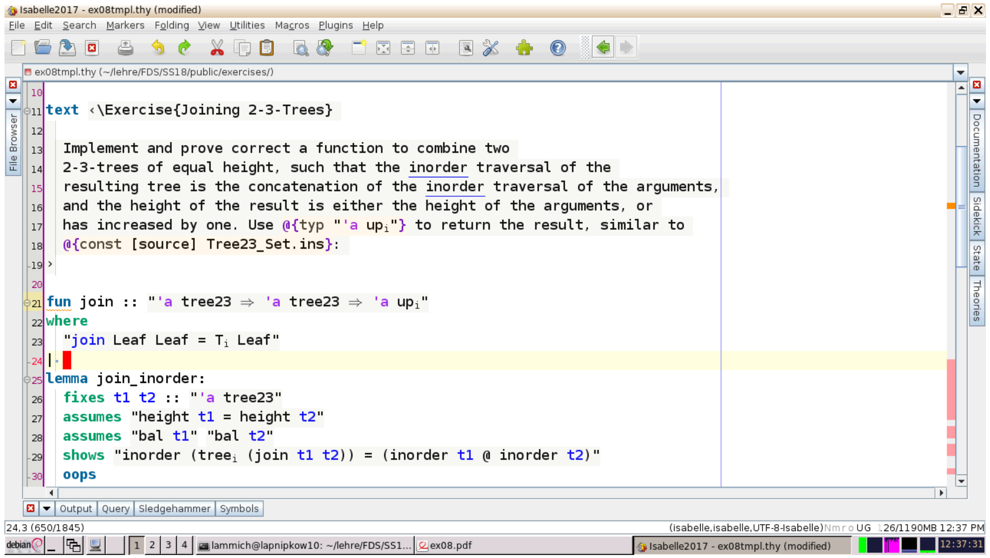

Isabelle2017 - ex08tmpl.thy
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a up₁"
where
"join Leaf Leaf = Tᵢ Leaf"
lemma join_inorder:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (treeᵢ (join t1 t2)) = (inorder t1 @ inorder t2)"
oops

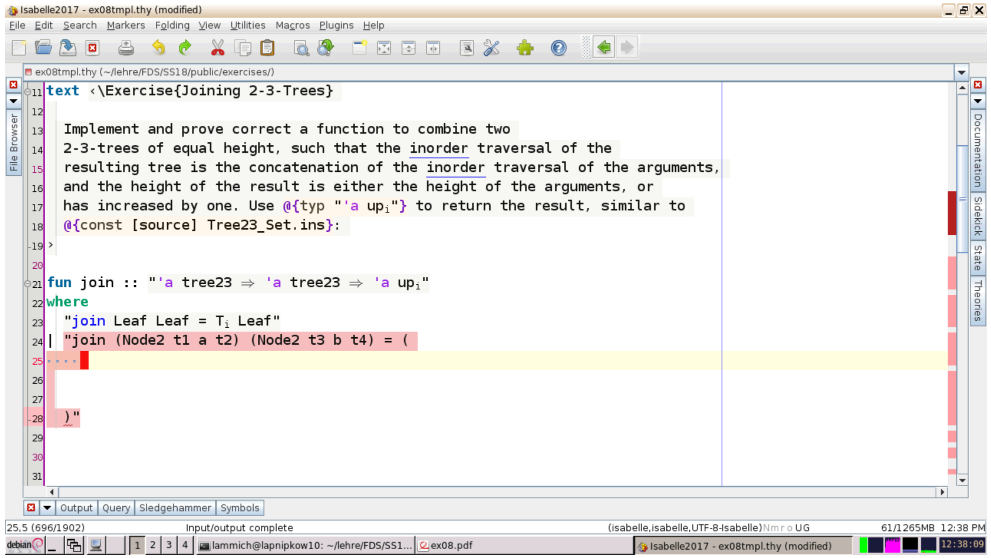

Isabelle2017 - ex08tmpl.thy
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a up₁"
where
"join Leaf Leaf = Tᵢ Leaf"
lemma join_inorder:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (treeᵢ (join t1 t2)) = (inorder t1 @ inorder t2)"
oops
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
oops
text ‹Hints:

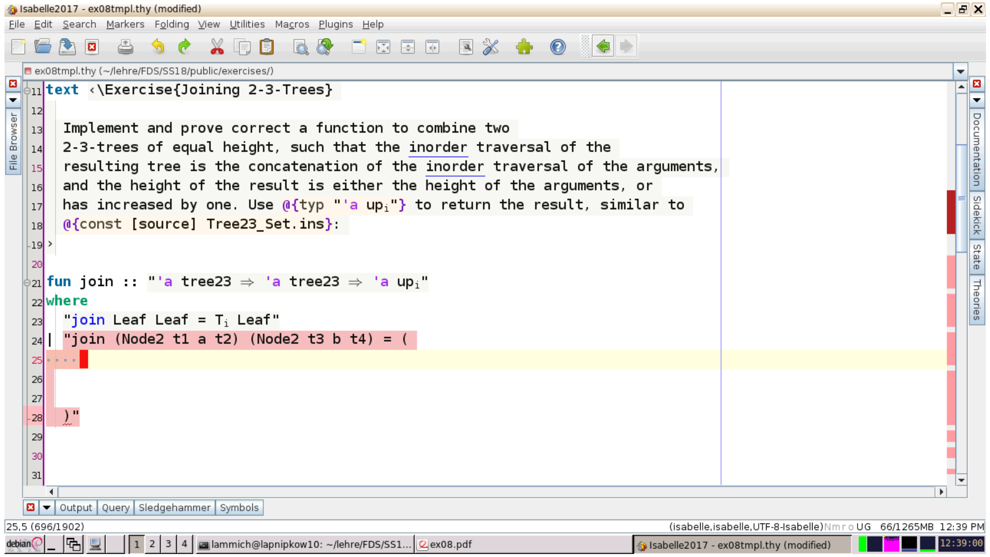
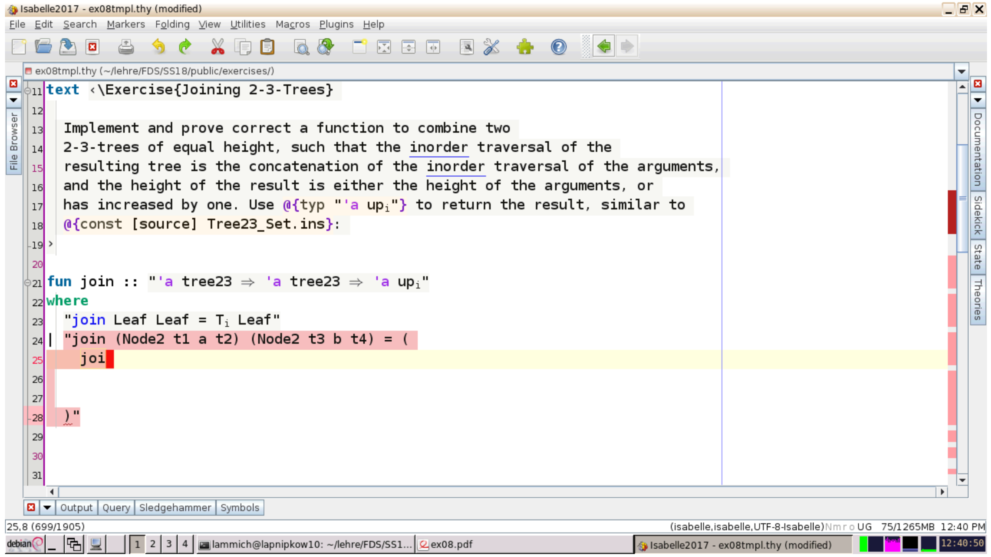
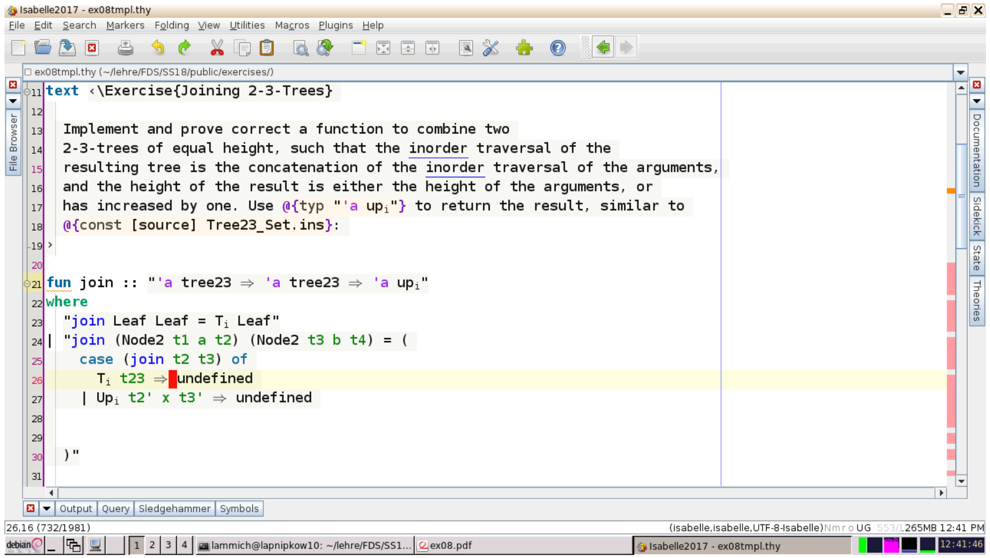
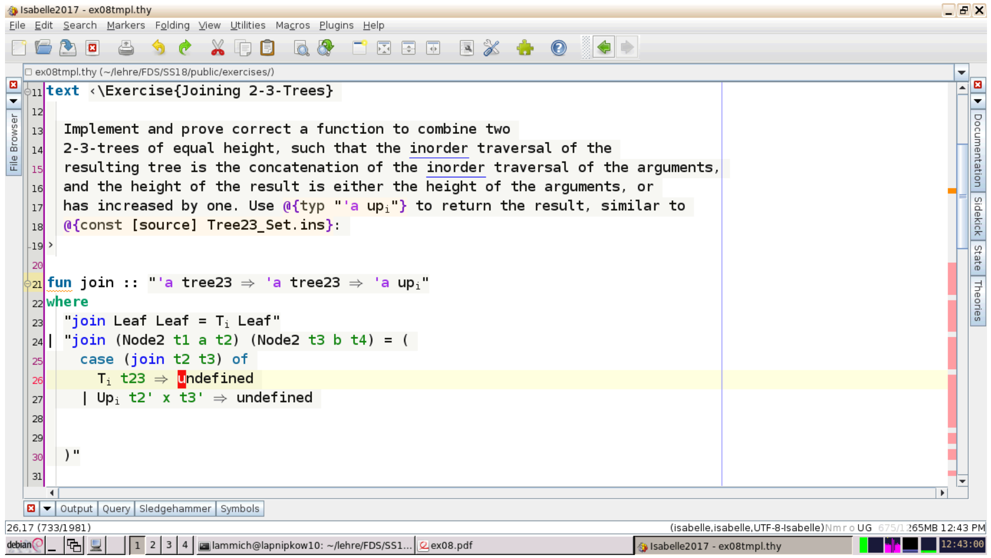

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a upᵢ"
where
"join Leaf Leaf = Tᵢ Leaf"
lemma join_inorder:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (treeᵢ (join t1 t2)) = (inorder t1 @ inorder t2)"
oops
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
oops
text ‹Hints:
Output Query Sledgehammer Symbols

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a upᵢ"
where
"join Leaf Leaf = Tᵢ Leaf"
lemma join_inorder:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (treeᵢ (join t1 t2)) = (inorder t1 @ inorder t2)"
oops
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
oops
text ‹Hints:
Output Query Sledgehammer Symbols

Isabelle2017 - Tree23_Set.thy
Tree23_Set.thy ($ISABELLE_HOME/src/HOL/Data_Structures/)
GT ⇒ isin r x)" |
"isin (Node3 l a m b r) x =
(case cmp x a of
LT ⇒ isin l x |
EQ ⇒ True |
GT ⇒
(case cmp x b of
LT ⇒ isin m x |
EQ ⇒ True |
GT ⇒ isin r x))"
datatype 'a upᵢ = Tᵢ "'a tree23" | Upᵢ "'a tree23" 'a "'a tree23"
fun treeᵢ :: "'a upᵢ ⇒ 'a tree23" where
"treeᵢ (Tᵢ t) = t" |
"treeᵢ (Upᵢ l a r) = Node2 l a r"
fun ins :: "'a::linorder ⇒ 'a tree23 ⇒ 'a upᵢ" where
"ins x Leaf = Upᵢ Leaf x Leaf" |
"ins x (Node2 l a r) =
(case cmp x a of
Output Query Sledgehammer Symbols

Isabelle2017 - Tree23_Set.thy
Tree23_Set.thy ($ISABELLE_HOME/src/HOL/Data_Structures/)
GT ⇒ isin r x)" |
"isin (Node3 l a m b r) x =
(case cmp x a of
LT ⇒ isin l x |
EQ ⇒ True |
GT ⇒
(case cmp x b of
LT ⇒ isin m x |
EQ ⇒ True |
GT ⇒ isin r x))"
datatype 'a upᵢ = Tᵢ "'a tree23" | Upᵢ "'a tree23" 'a "'a tree23"
fun treeᵢ :: "'a upᵢ ⇒ 'a tree23" where
"treeᵢ (Tᵢ t) = t" |
"treeᵢ (Upᵢ l a r) = Node2 l a r"
fun ins :: "'a::linorder ⇒ 'a tree23 ⇒ 'a upᵢ" where
"ins x Leaf = Upᵢ Leaf x Leaf" |
"ins x (Node2 l a r) =
(case cmp x a of
Output Query Sledgehammer Symbols

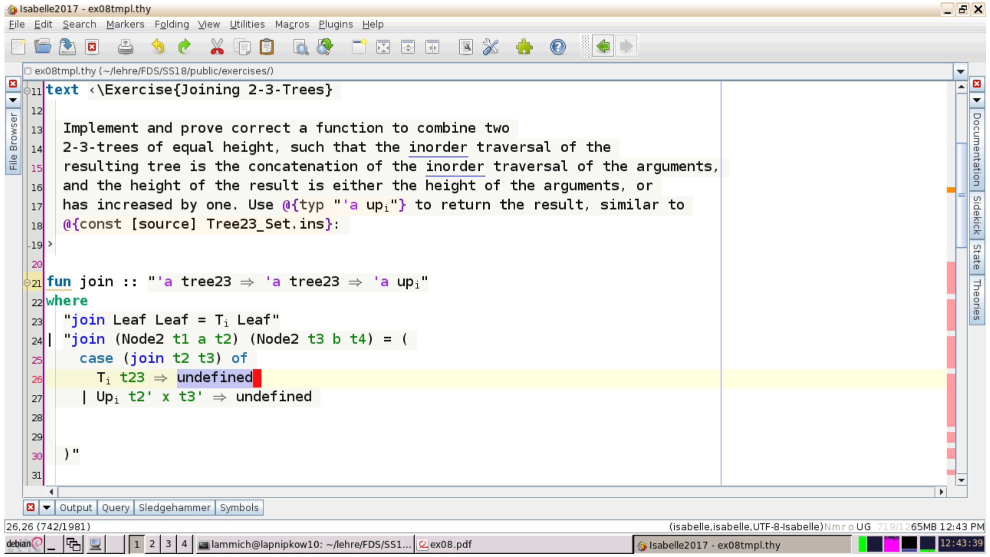
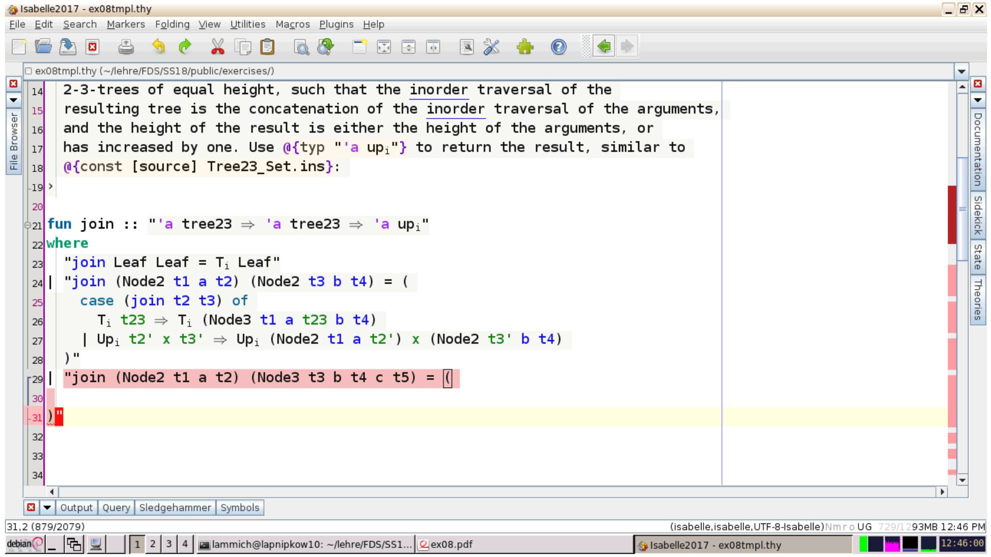
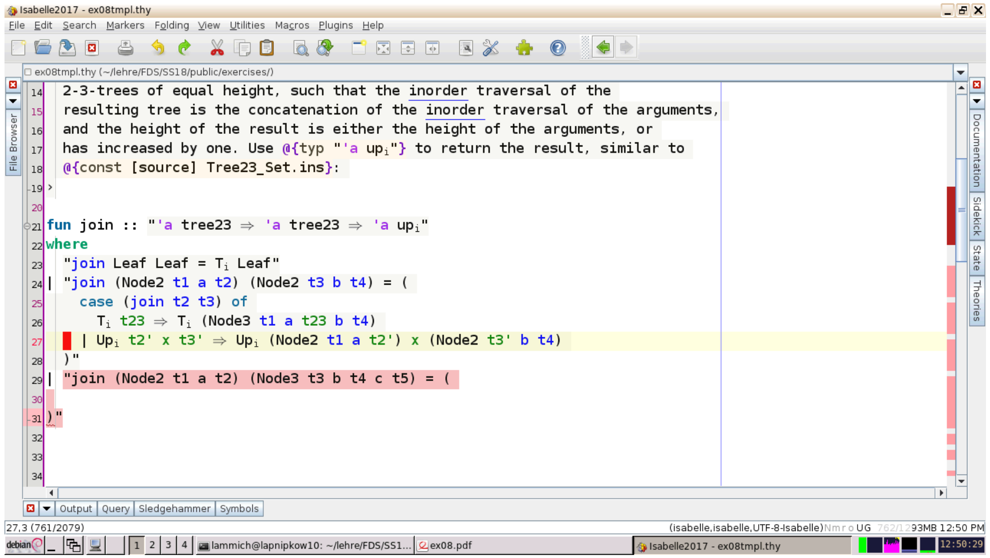
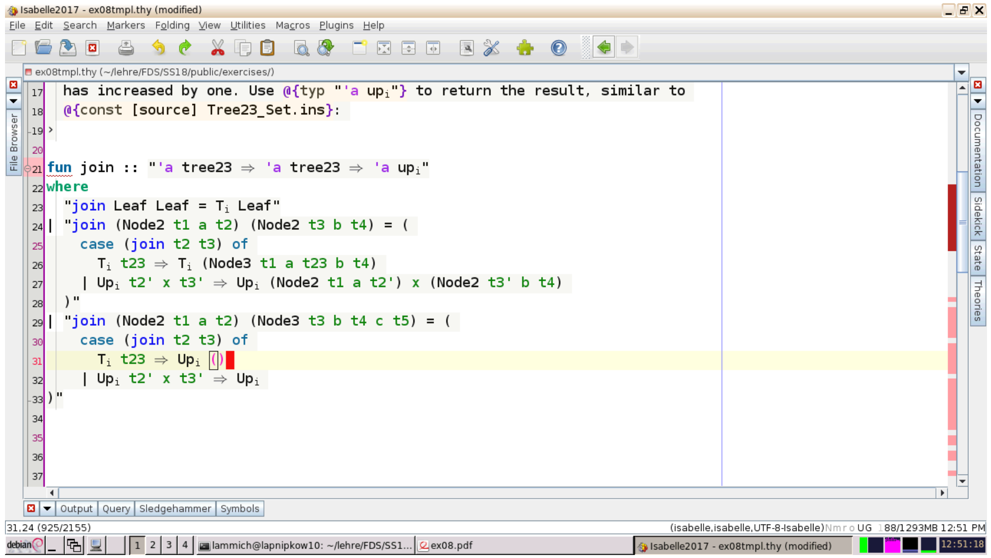

Isabelle2017 - Tree23_Set.thy
Tree23_Set.thy ($ISABELLE_HOME/src/HOL/Data_Structures/)
by(simp add: delete_def inorder_del)
subsection ‹Balancedness›
subsubsection "Proofs for insert"
text{* First a standard proof that @{const ins} preserves @{const bal}. *}
instantiation upᵢ :: (type)height
begin
fun height_upᵢ :: "'a upᵢ ⇒ nat" where
"height (Tᵢ t) = height t" |
"height (Upᵢ l a r) = height l"
instance ..
end
Output Query Sledgehammer Symbols

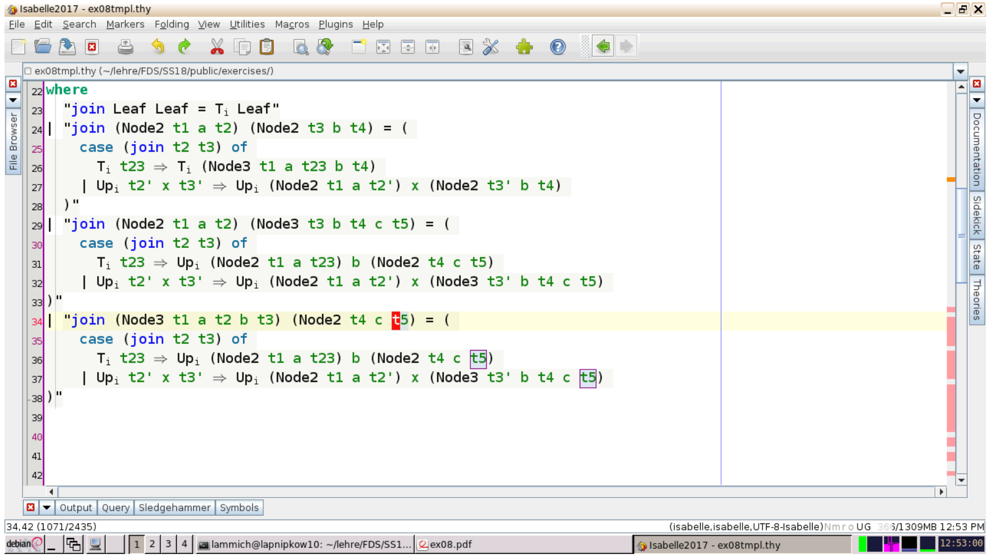

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
"join Leaf Leaf = Tᵢ Leaf"
lemma join_inorder:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (treeᵢ (join t1 t2)) = (inorder t1 @ inorder t2)"
oops
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
oops
text ‹Hints:
▪ Try to use automatic case splitting (‹auto split: …›) instead of explicit case splitting via Isar
(There will be dozens of cases).
▪ To find bugs in your join function, or isolate the case where your automatic proof does not (yet) work,
Output Query Sledgehammer Symbols

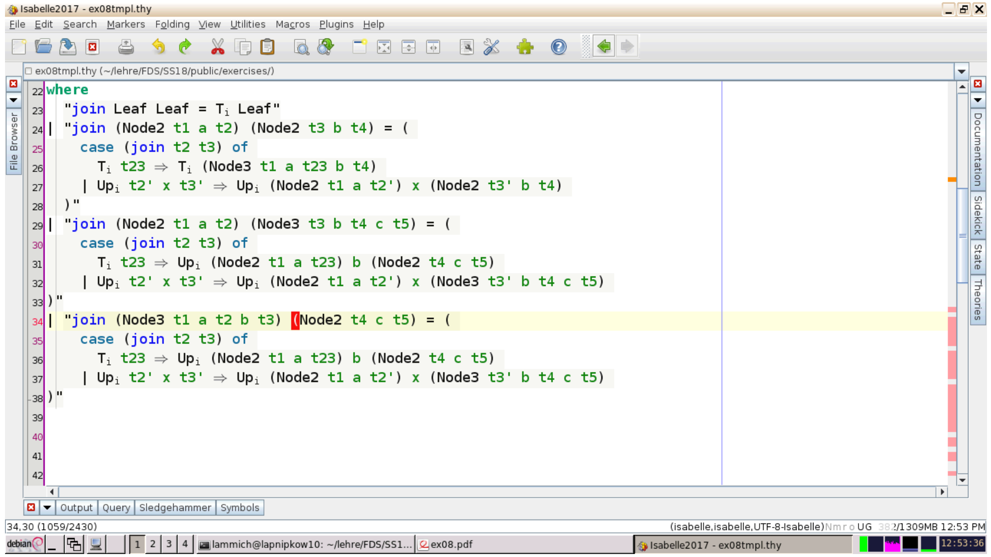

Isabelle2017 - ex08tmpl.thy (modified)
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
text ‹\Exercise{Joining 2-3-Trees}
Implement and prove correct a function to combine two
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a upᵢ"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a upᵢ"
where
"join Leaf Leaf = Tᵢ Leaf"
|
lemma join_inorder:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (treeᵢ (join t1 t2)) = (inorder t1 @ inorder t2)"
oops
Output Query Sledgehammer Symbols

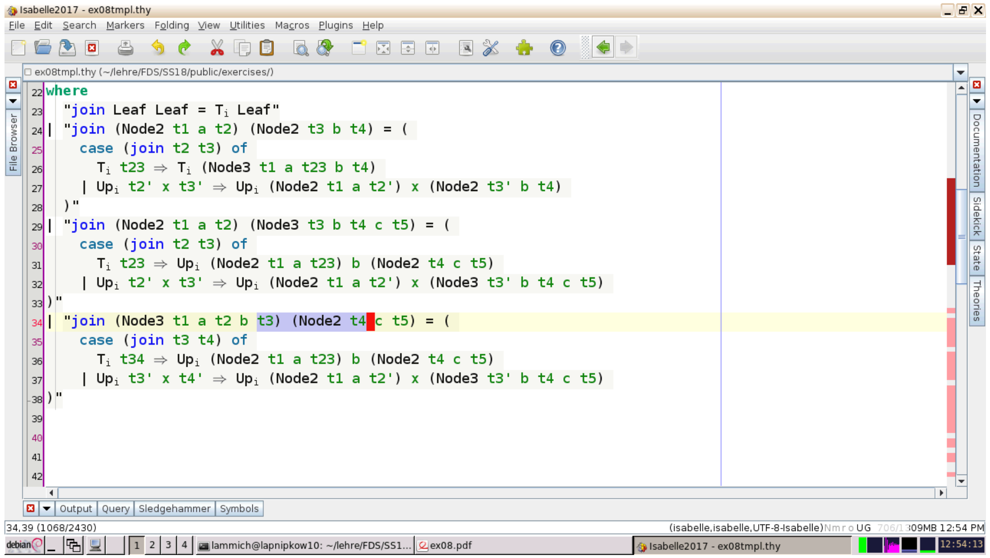

Isabelle2017 - ex08tmpl.thy (modified)
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
text ‹\Exercise{Joining 2-3-Trees}
Implement and prove correct a function to combine two
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a upᵢ"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a upᵢ"
where
"join Leaf Leaf = Tᵢ Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
)"
Output Query Sledgehammer Symbols
Input/output complete

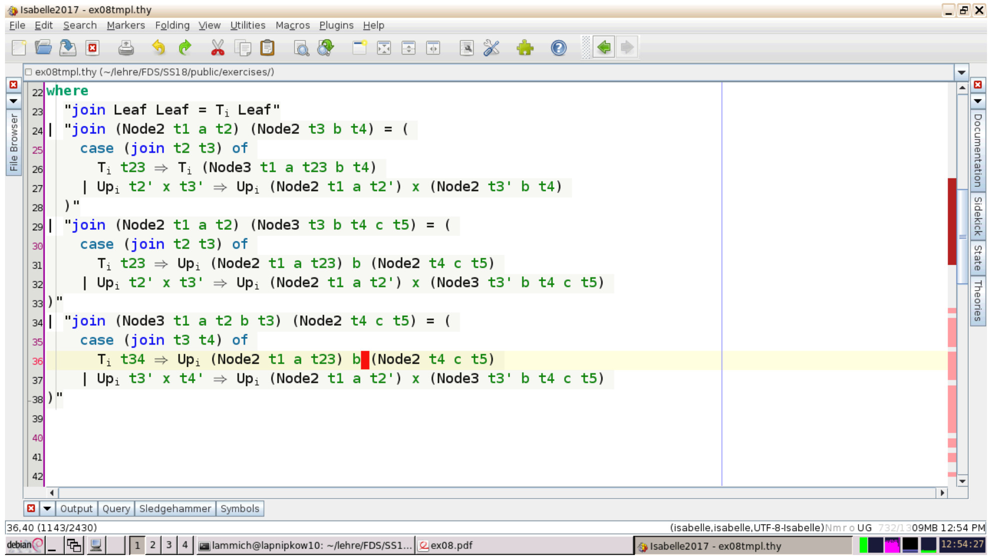

Isabelle2017 - ex08tmpl.thy (modified)
text ‹\Exercise{Joining 2-3-Trees}
Implement and prove correct a function to combine two
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a upᵢ"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a upᵢ"
where
"join Leaf Leaf = Tᵢ Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
)"
Output Query Sledgehammer Symbols
25,5 (696/1902)
12:39 PM

Isabelle2017 - ex08tmpl.thy (modified)
text ‹\Exercise{Joining 2-3-Trees}
Implement and prove correct a function to combine two
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a upᵢ"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a upᵢ"
where
"join Leaf Leaf = Tᵢ Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
joi
)"
Output Query Sledgehammer Symbols
25,8 (699/1905)
12:40 PM

Isabelle2017 - ex08tmpl.thy
text ‹\Exercise{Joining 2-3-Trees}
Implement and prove correct a function to combine two
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a upᵢ"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a upᵢ"
where
"join Leaf Leaf = Tᵢ Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
Tᵢ t23 ⇒ undefined
| Upᵢ t2' x t3' ⇒ undefined
)"
Output Query Sledgehammer Symbols
26,16 (732/1981)
12:41 PM

Isabelle2017 - ex08tmpl.thy
text ‹\Exercise{Joining 2-3-Trees}
Implement and prove correct a function to combine two
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a upᵢ"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a upᵢ"
where
"join Leaf Leaf = Tᵢ Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
Tᵢ t23 ⇒ undefined
| Upᵢ t2' x t3' ⇒ undefined
)"
Output Query Sledgehammer Symbols
26,17 (733/1981)
12:43 PM

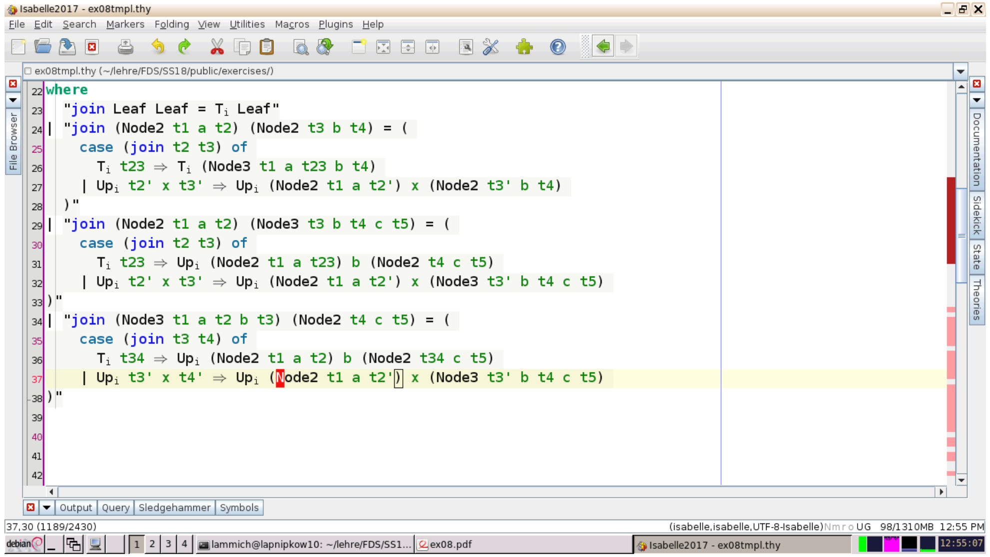

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
text ‹\Exercise{Joining 2-3-Trees}
Implement and prove correct a function to combine two
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a up_i"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a up_i"
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ undefined
| Up_i t2' x t3' ⇒ undefined
)"
Output Query Sledgehammer Symbols

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a up_i"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a up_i"
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
)"
Output Query Sledgehammer Symbols

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
2-3-trees of equal height, such that the inorder traversal of the
resulting tree is the concatenation of the inorder traversal of the arguments,
and the height of the result is either the height of the arguments, or
has increased by one. Use @{typ "'a up_i"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a up_i"
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
)"
Output Query Sledgehammer Symbols

Isabelle2017 - ex08tmpl.thy (modified)
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
has increased by one. Use @{typ "'a up_i"} to return the result, similar to
@{const [source] Tree23_Set.ins}:
›
fun join :: "'a tree23 ⇒ 'a tree23 ⇒ 'a up_i"
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i ()
| Up_i t2' x t3' ⇒ Up_i
)"
Output Query Sledgehammer Symbols

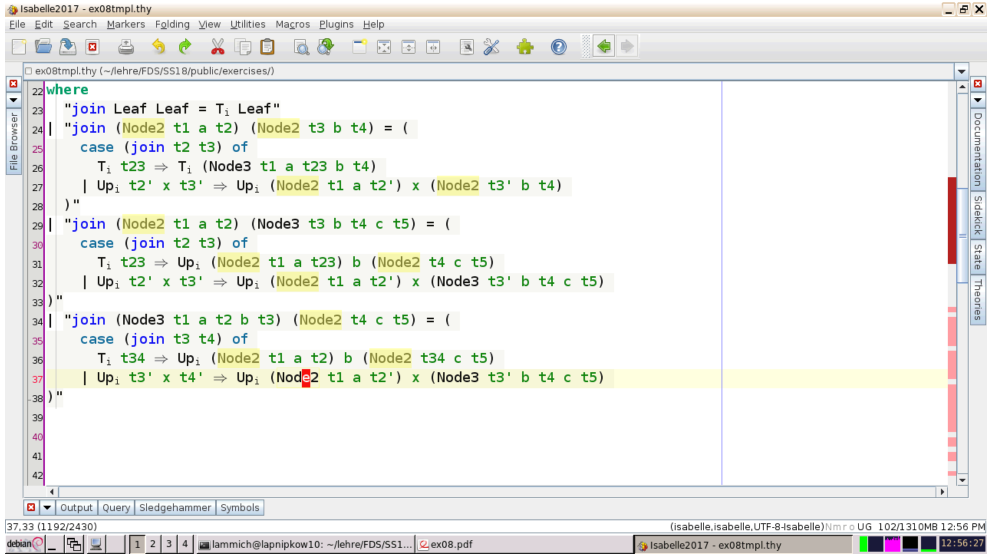

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"

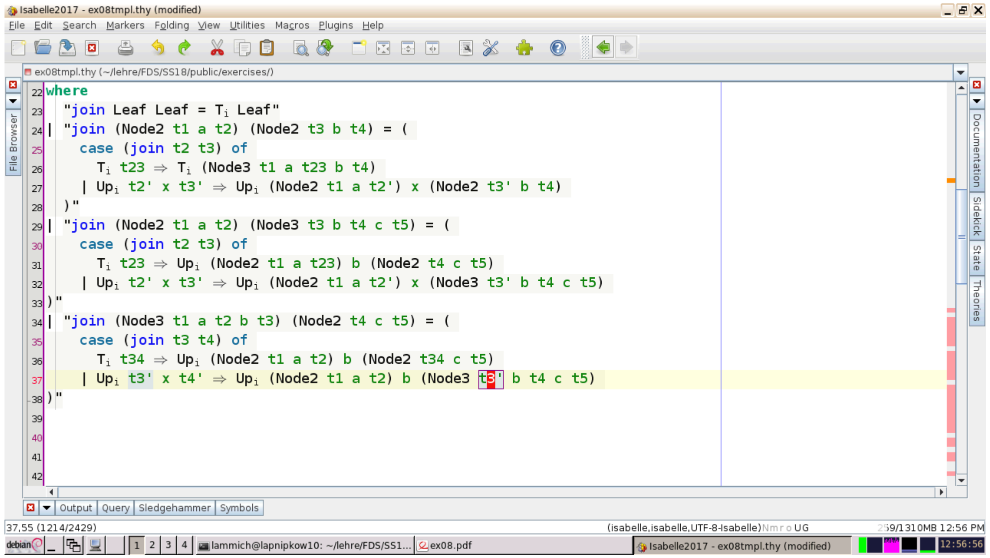

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"

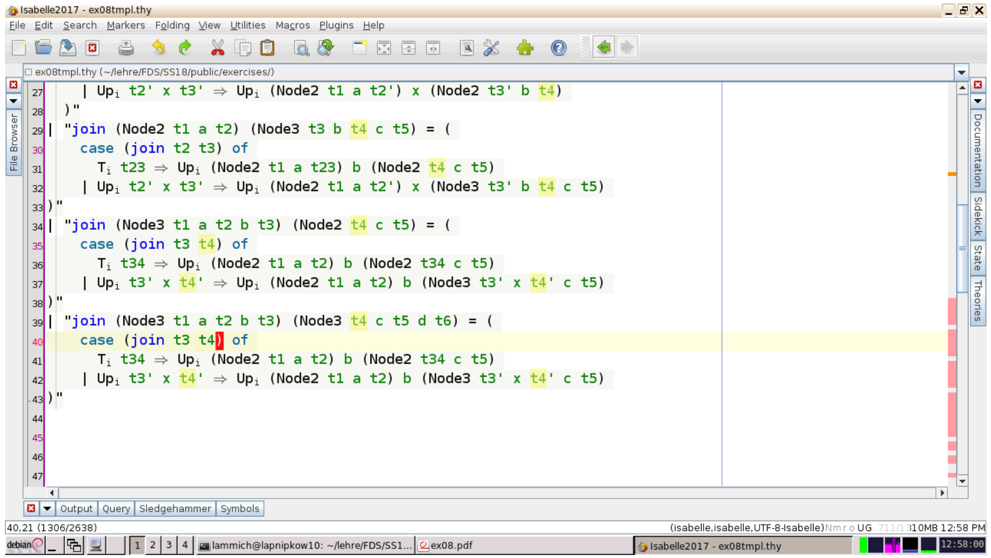

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"

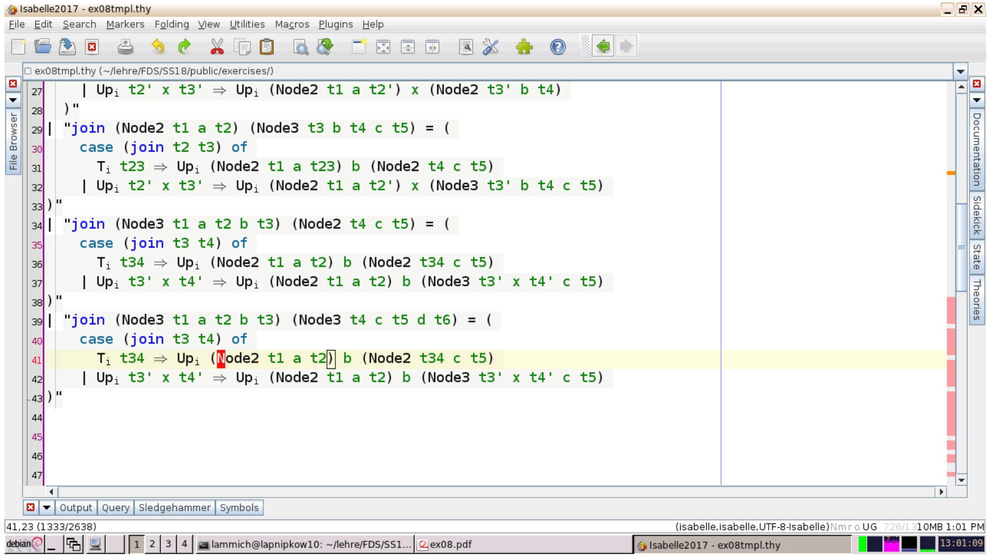

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"

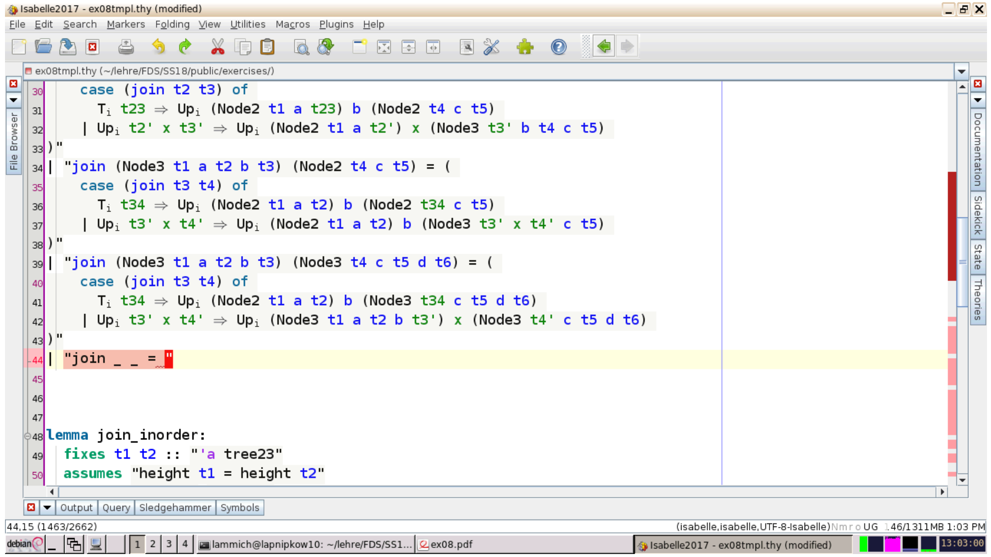

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t2) b (Node2 t34 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
37,30 (1189/2430)

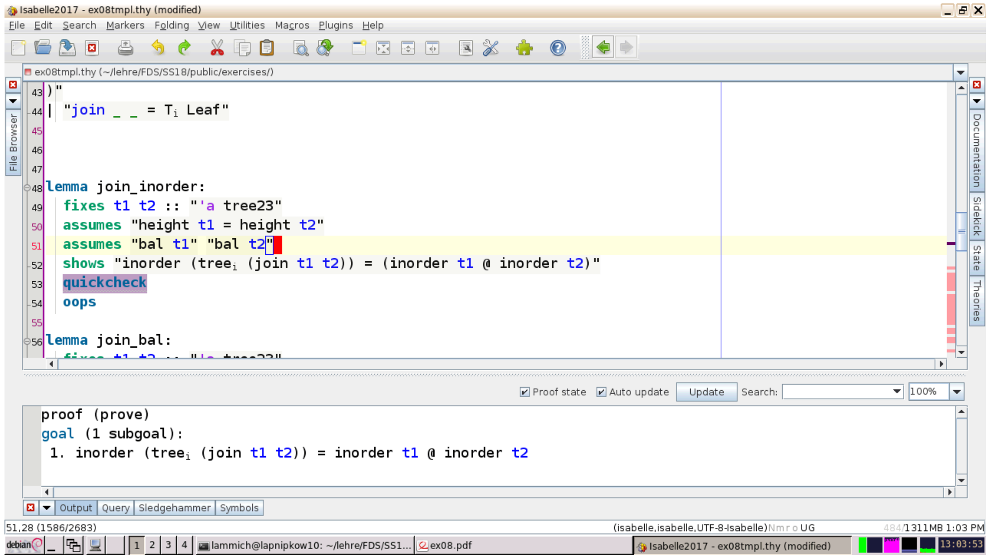

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t2) b (Node2 t34 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
37,33 (1192/2430)

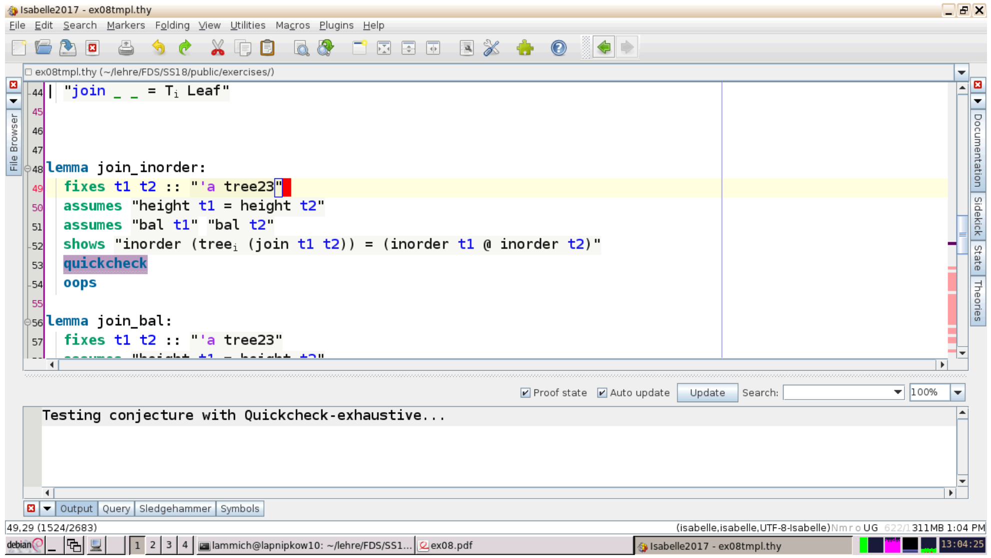

Isabelle2017 - ex08tmpl.thy (modified)
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
where
"join Leaf Leaf = T_i Leaf"
| "join (Node2 t1 a t2) (Node2 t3 b t4) = (
case (join t2 t3) of
T_i t23 ⇒ T_i (Node3 t1 a t23 b t4)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t2) b (Node2 t34 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2) b (Node3 t3' b t4 c t5)
)"
37,55 (1214/2429)

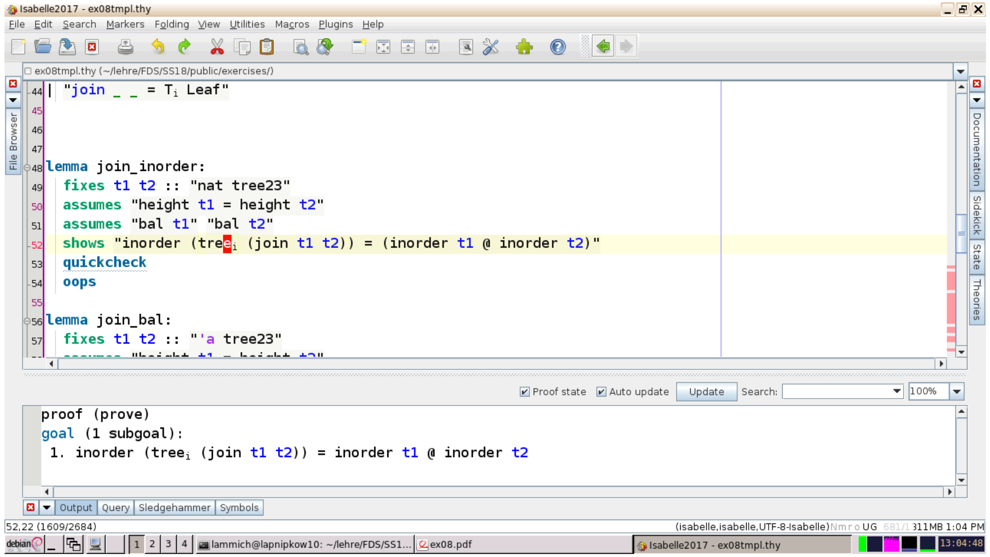

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t2) b (Node2 t34 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2) b (Node3 t3' x t4' c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node3 t4 c t5 d t6) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t2) b (Node2 t34 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2) b (Node3 t3' x t4' c t5)
)"
40,21 (1306/2638)

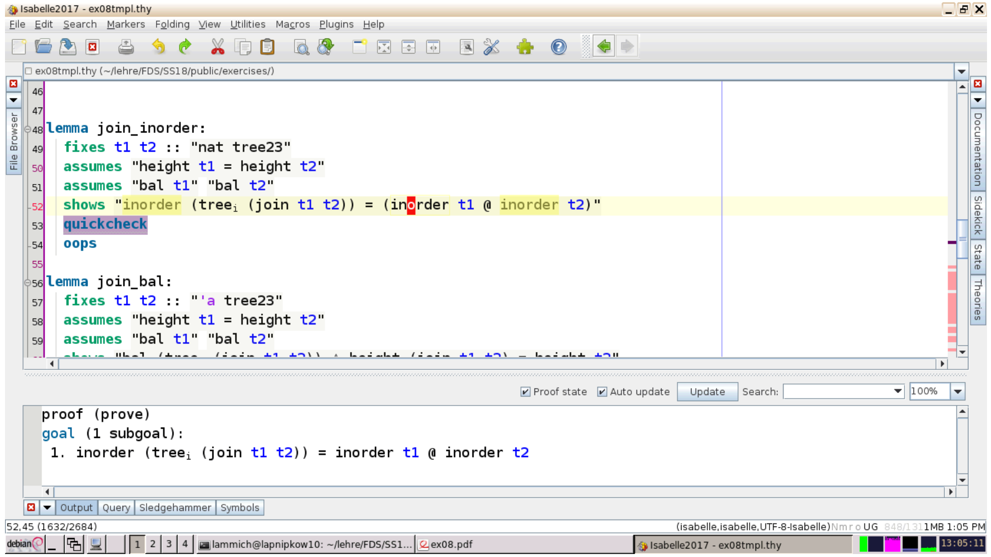

Isabelle2017 - ex08tmpl.thy
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node2 t3' b t4)
)"
| "join (Node2 t1 a t2) (Node3 t3 b t4 c t5) = (
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t2) b (Node2 t34 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2) b (Node3 t3' x t4' c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node3 t4 c t5 d t6) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t2) b (Node2 t34 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2) b (Node3 t3' x t4' c t5)
)"

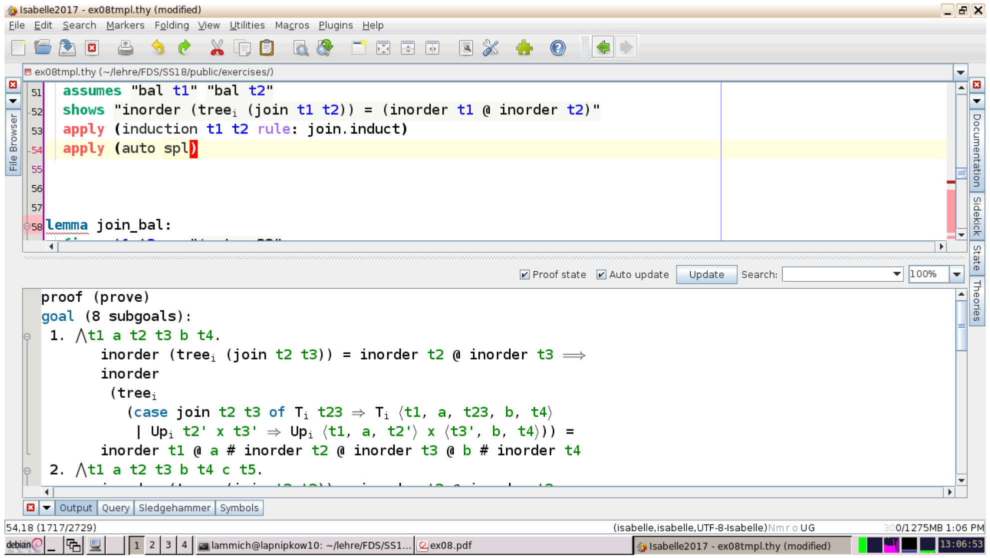

Isabelle2017 - ex08tmpl.thy (modified)
case (join t2 t3) of
T_i t23 ⇒ Up_i (Node2 t1 a t23) b (Node2 t4 c t5)
| Up_i t2' x t3' ⇒ Up_i (Node2 t1 a t2') x (Node3 t3' b t4 c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node2 t4 c t5) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t2) b (Node2 t34 c t5)
| Up_i t3' x t4' ⇒ Up_i (Node2 t1 a t2) b (Node3 t3' x t4' c t5)
)"
| "join (Node3 t1 a t2 b t3) (Node3 t4 c t5 d t6) = (
case (join t3 t4) of
T_i t34 ⇒ Up_i (Node2 t1 a t2) b (Node3 t34 c t5 d t6)
| Up_i t3' x t4' ⇒ Up_i (Node3 t1 a t2 b t3') x (Node3 t4' c t5 d t6)
)"
| "join _ _ = "
lemma join_inorder:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"

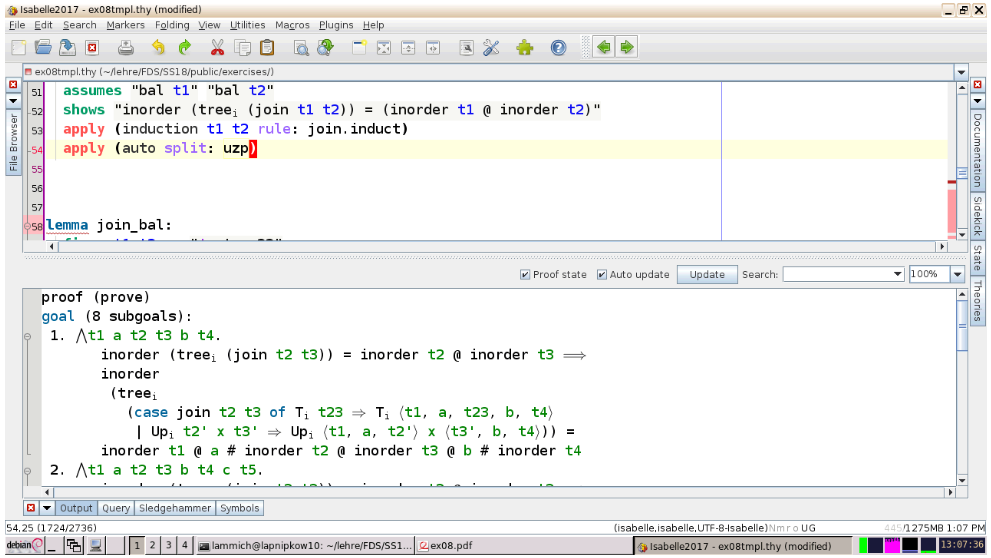

Isabelle2017 - ex08tmpl.thy (modified)
)"
| "join _ _ = T_i Leaf"
lemma join_inorder:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (tree_i (join t1 t2)) = (inorder t1 @ inorder t2)"
quickcheck
oops
lemma join_bal:
proof (prove)
goal (1 subgoal):
1. inorder (tree_i (join t1 t2)) = inorder t1 @ inorder t2

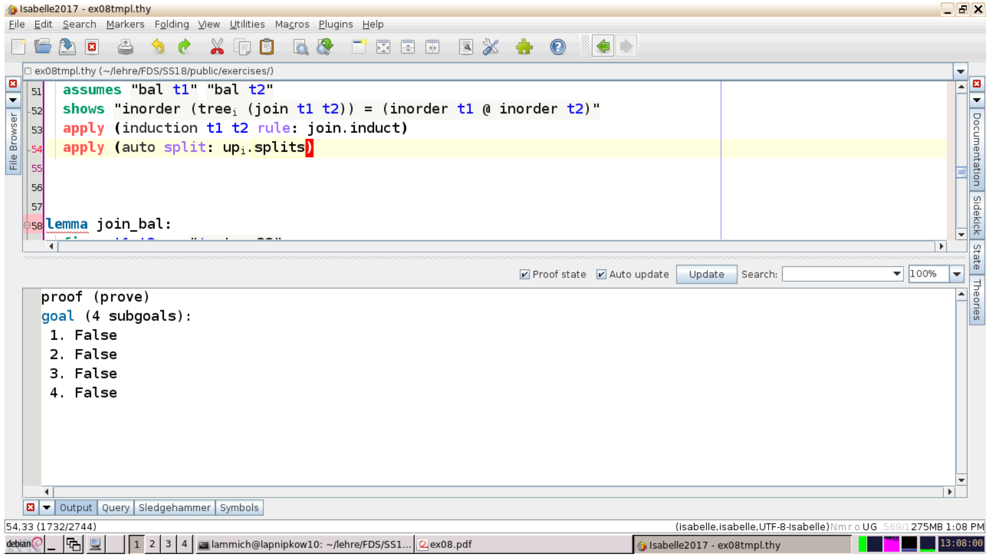

Isabelle2017 - ex08tmpl.thy
| "join _ _ = T_i Leaf"
lemma join_inorder:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (tree_i (join t1 t2)) = (inorder t1 @ inorder t2)"
quickcheck
oops
lemma join_bal:
fixes t1 t2 :: "'a tree23"
Testing conjecture with Quickcheck-exhaustive...


Isabelle2017 - ex08tmpl.thy
| "join _ _ = T₁ Leaf"
lemma join_inorder:
fixes t1 t2 :: "nat tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (tree₁ (join t1 t2)) = (inorder t1 @ inorder t2)"
quickcheck
oops
lemma join_bal:
fixes t1 t2 :: "'a tree23"
proof (prove)
goal (1 subgoal):
1. inorder (tree₁ (join t1 t2)) = inorder t1 @ inorder t2

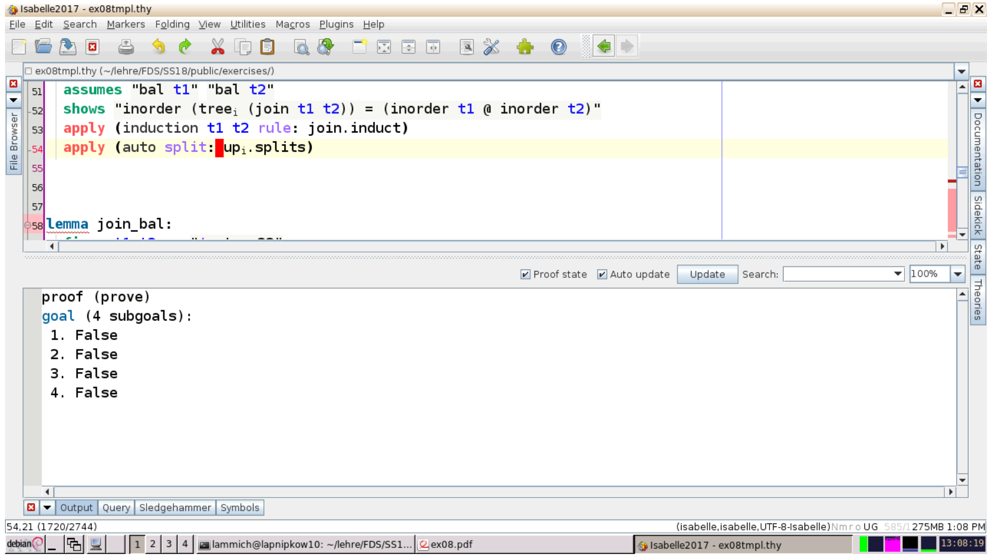

Isabelle2017 - ex08tmpl.thy
lemma join_inorder:
fixes t1 t2 :: "nat tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (tree₁ (join t1 t2)) = (inorder t1 @ inorder t2)"
quickcheck
oops
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
proof (prove)
goal (1 subgoal):
1. inorder (tree₁ (join t1 t2)) = inorder t1 @ inorder t2


Isabelle2017 - ex08tmpl.thy (modified)
assumes "bal t1" "bal t2"
shows "inorder (tree₁ (join t1 t2)) = (inorder t1 @ inorder t2)"
apply (induction t1 t2 rule: join.induct)
apply (auto spl)
lemma join_bal:
proof (prove)
goal (8 subgoals):
1. ⋀t1 a t2 t3 b t4.
inorder (tree₁ (join t2 t3)) = inorder t2 @ inorder t3 ⟹
inorder
(tree₁
(case join t2 t3 of T₁ t23 ⇒ T₁ ⟨t1, a, t23, b, t4⟩
| Up₁ t2' x t3' ⇒ Up₁ ⟨t1, a, t2'⟩ x ⟨t3', b, t4⟩)) =
inorder t1 @ a # inorder t2 @ inorder t3 @ b # inorder t4
2. ⋀t1 a t2 t3 b t4 c t5.

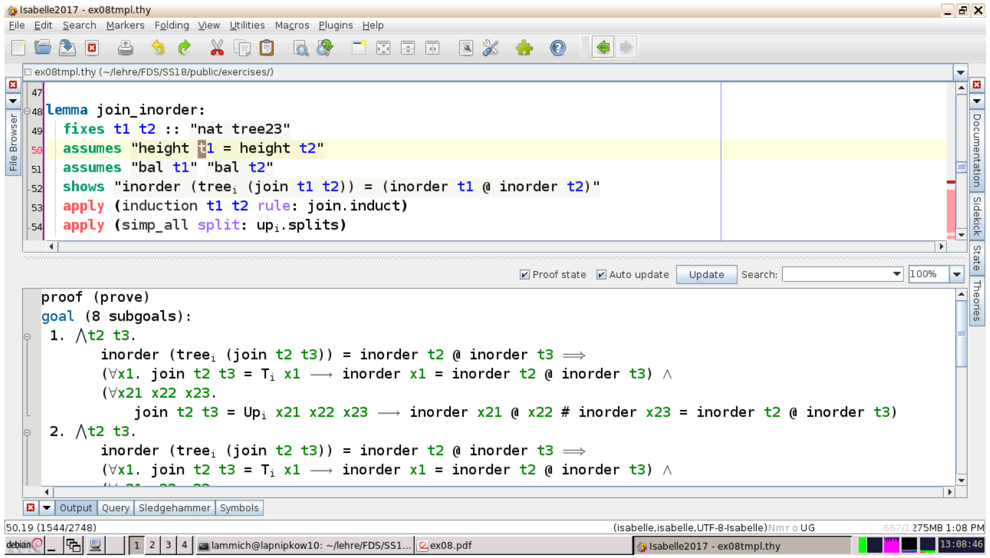

Isabelle2017 - ex08tmpl.thy (modified)
assumes "bal t1" "bal t2"
shows "inorder (tree₁ (join t1 t2)) = (inorder t1 @ inorder t2)"
apply (induction t1 t2 rule: join.induct)
apply (auto split: uzp)
lemma join_bal:
proof (prove)
goal (8 subgoals):
1. ⋀t1 a t2 t3 b t4.
inorder (tree₁ (join t2 t3)) = inorder t2 @ inorder t3 ⟹
inorder
(tree₁
(case join t2 t3 of T₁ t23 ⇒ T₁ ⟨t1, a, t23, b, t4⟩
| Up₁ t2' x t3' ⇒ Up₁ ⟨t1, a, t2'⟩ x ⟨t3', b, t4⟩)) =
inorder t1 @ a # inorder t2 @ inorder t3 @ b # inorder t4
2. ⋀t1 a t2 t3 b t4 c t5.



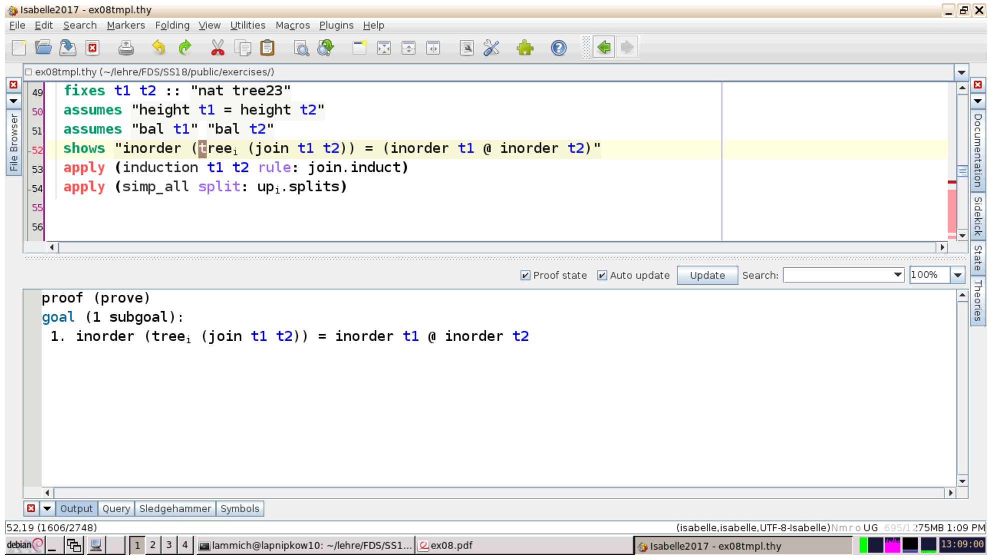
Isabelle2017 - ex08tmpl.thy
assumes "bal t1" "bal t2"
shows "inorder (tree_i (join t1 t2)) = (inorder t1 @ inorder t2)"
apply (induction t1 t2 rule: join.induct)
apply (auto split: up_i.splits)
lemma join_bal:
proof (prove)
goal (4 subgoals):
1. False
2. False
3. False
4. False



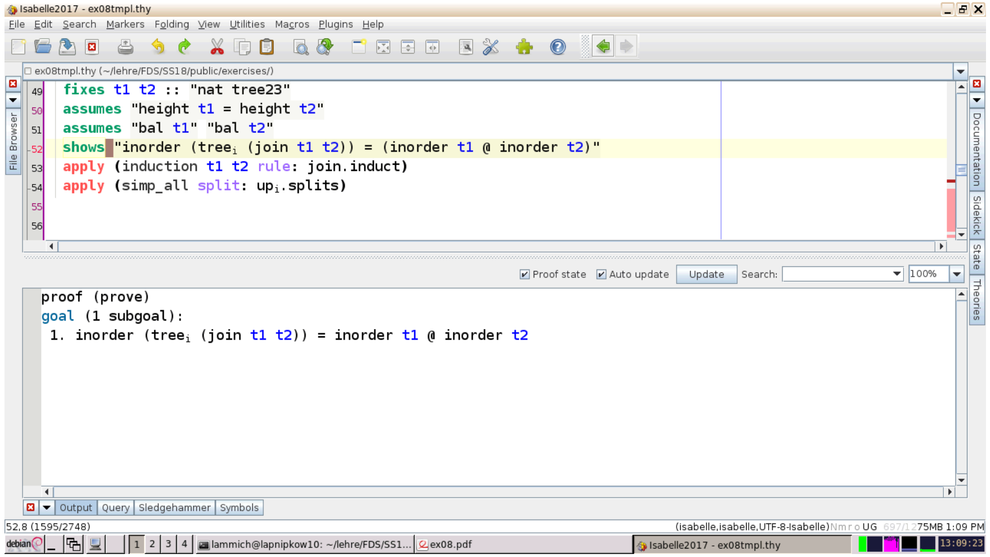
Isabelle2017 - ex08tmpl.thy
assumes "bal t1" "bal t2"
shows "inorder (tree_i (join t1 t2)) = (inorder t1 @ inorder t2)"
apply (induction t1 t2 rule: join.induct)
apply (auto split: up_i.splits)
lemma join_bal:
proof (prove)
goal (4 subgoals):
1. False
2. False
3. False
4. False

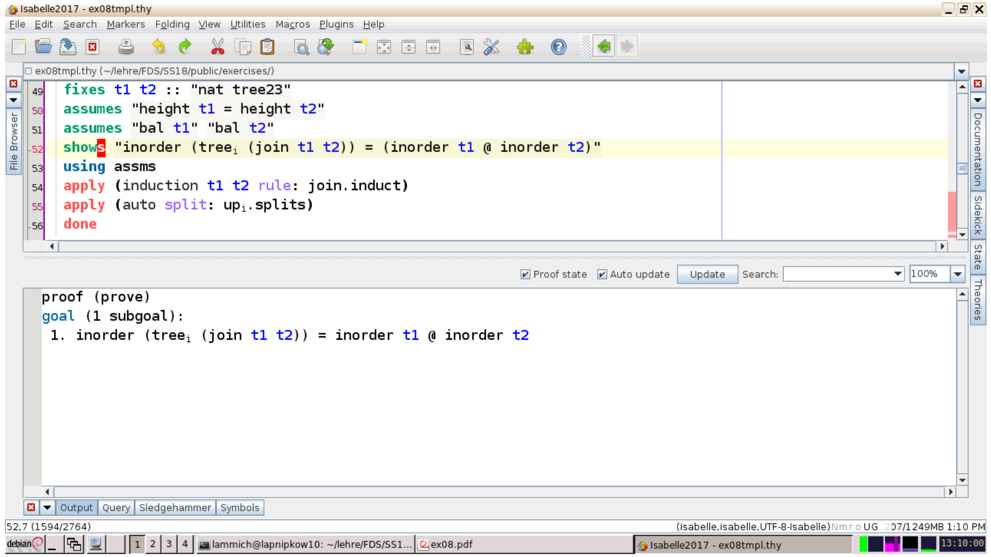


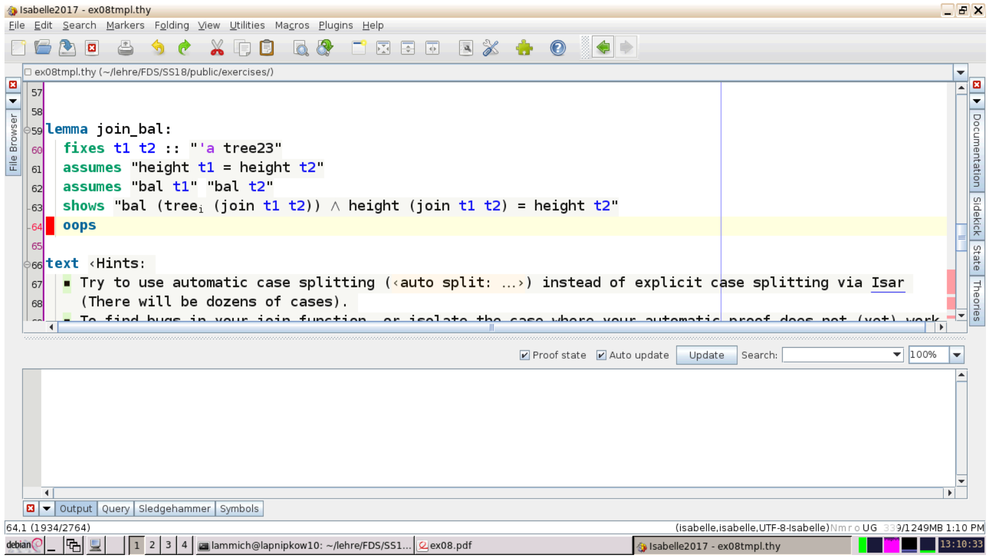
Isabelle2017 - ex08tmpl.thy
lemma join_inorder:
fixes t1 t2 :: "nat tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (tree_i (join t1 t2)) = (inorder t1 @ inorder t2)"
apply (induction t1 t2 rule: join.induct)
apply (simp_all split: up_i.splits)
proof (prove)
goal (8 subgoals):
1. ⋀t2 t3.
inorder (tree_i (join t2 t3)) = inorder t2 @ inorder t3 ⟹
(∀x1. join t2 t3 = T_i x1 ⟶ inorder x1 = inorder t2 @ inorder t3) ∧
(∀x21 x22 x23.
join t2 t3 = Up_i x21 x22 x23 ⟶ inorder x21 @ x22 # inorder x23 = inorder t2 @ inorder t3)
2. ⋀t2 t3.
inorder (tree_i (join t2 t3)) = inorder t2 @ inorder t3 ⟹
(∀x1. join t2 t3 = T_i x1 ⟶ inorder x1 = inorder t2 @ inorder t3) ∧

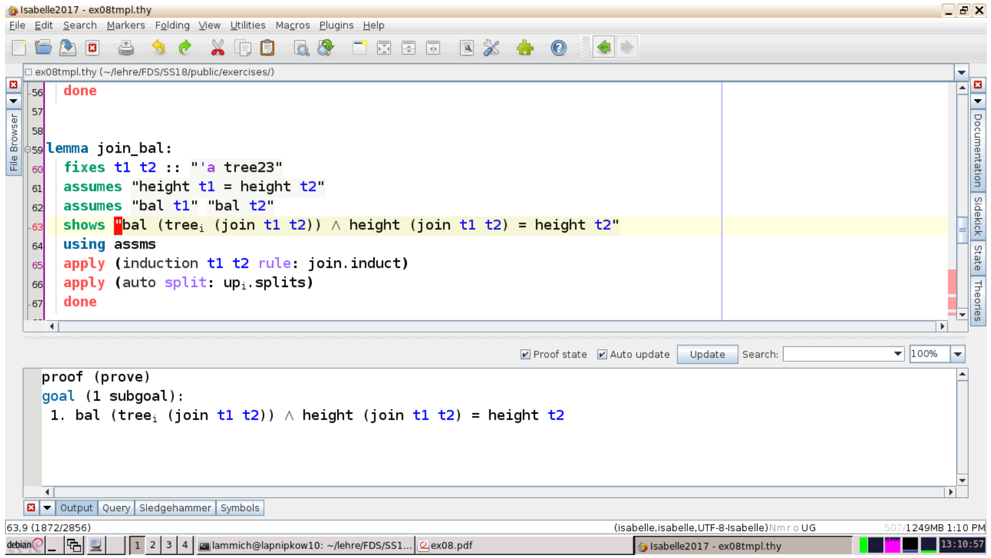


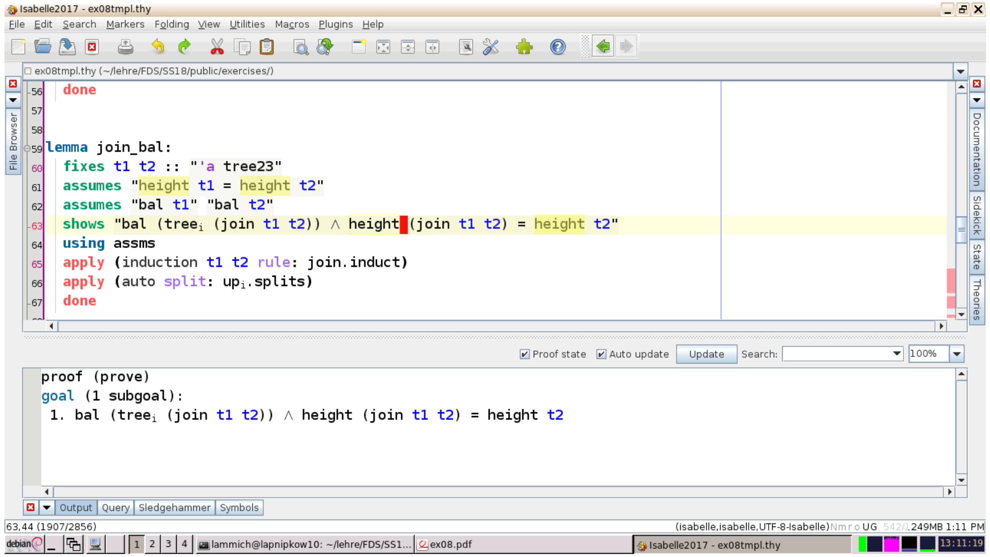
Isabelle2017 - ex08tmpl.thy
fixes t1 t2 :: "nat tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (tree_i (join t1 t2)) = (inorder t1 @ inorder t2)"
apply (induction t1 t2 rule: join.induct)
apply (simp_all split: up_i.splits)
proof (prove)
goal (1 subgoal):
1. inorder (tree_i (join t1 t2)) = inorder t1 @ inorder t2

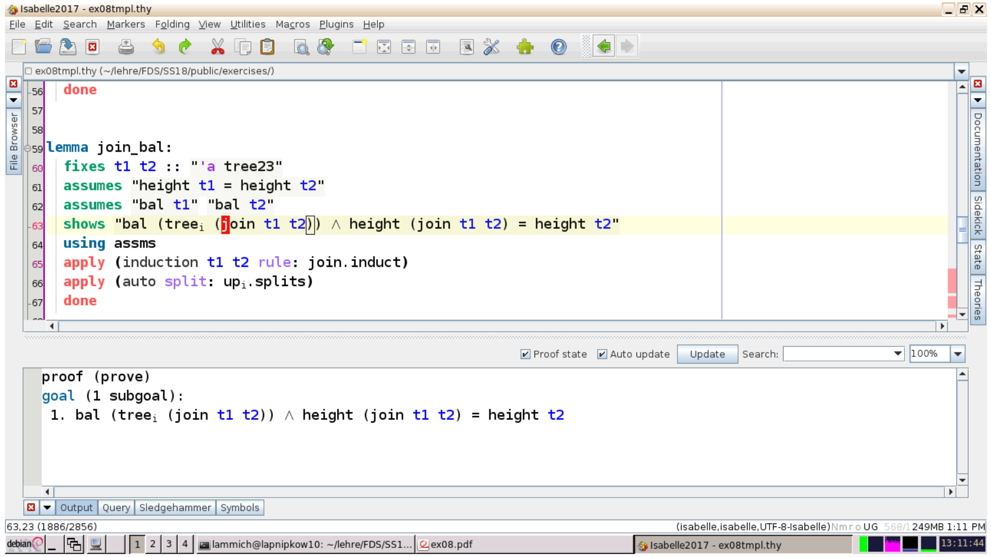

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
fixes t1 t2 :: "nat tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (tree₁ (join t1 t2)) = (inorder t1 @ inorder t2)"
apply (induction t1 t2 rule: join.induct)
apply (simp_all split: up₁.splits)
proof (prove)
goal (1 subgoal):
1. inorder (tree₁ (join t1 t2)) = inorder t1 @ inorder t2

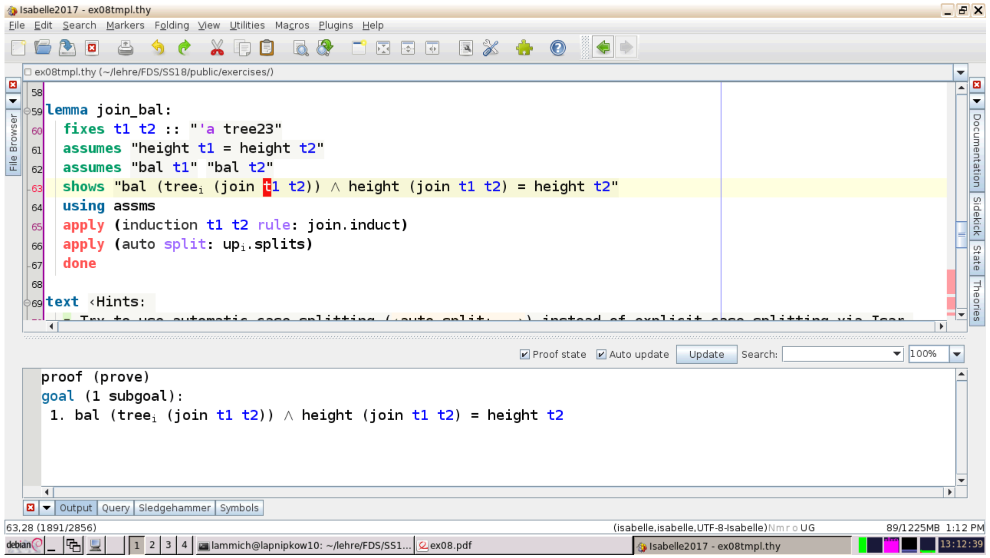

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
fixes t1 t2 :: "nat tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "inorder (tree₁ (join t1 t2)) = (inorder t1 @ inorder t2)"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: up₁.splits)
done
proof (prove)
goal (1 subgoal):
1. inorder (tree₁ (join t1 t2)) = inorder t1 @ inorder t2

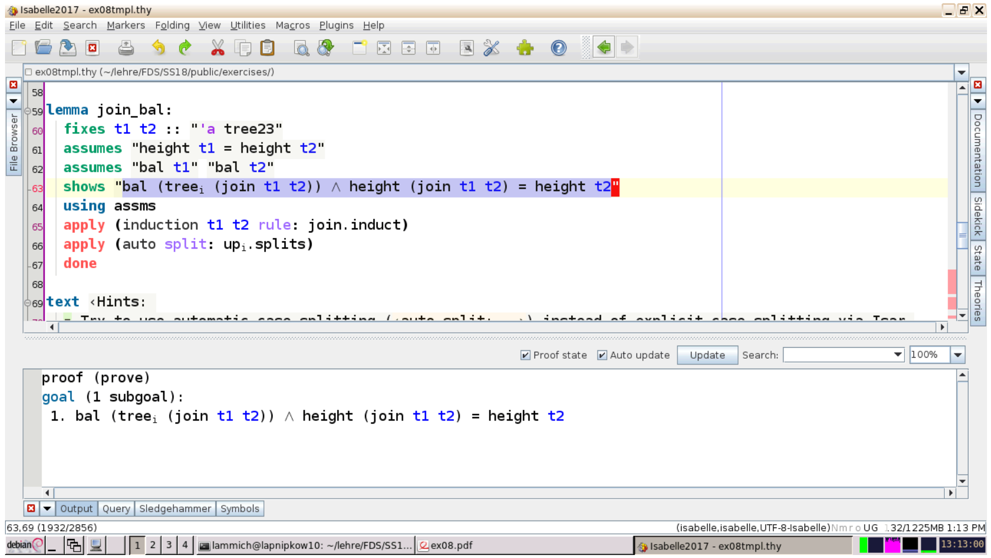

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (tree₁ (join t1 t2)) ∧ height (join t1 t2) = height t2"
oops
text ‹Hints:
▪ Try to use automatic case splitting (‹auto split: ...›) instead of explicit case splitting via Isar
(There will be dozens of cases).

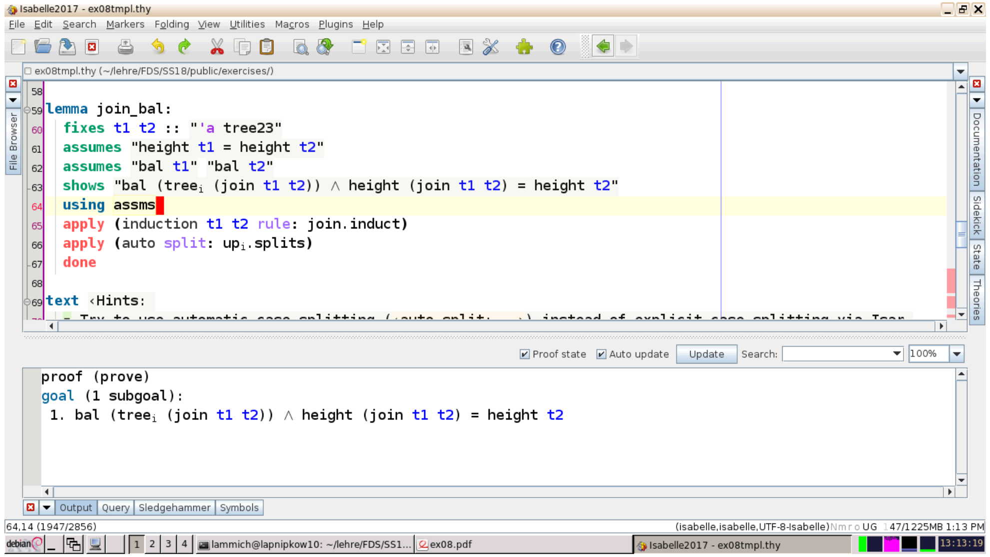

Isabelle2017 - ex08tmpl.thy
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
done
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (tree₁ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: up₁.splits)
done
proof (prove)
goal (1 subgoal):
1. bal (tree₁ (join t1 t2)) ∧ height (join t1 t2) = height t2

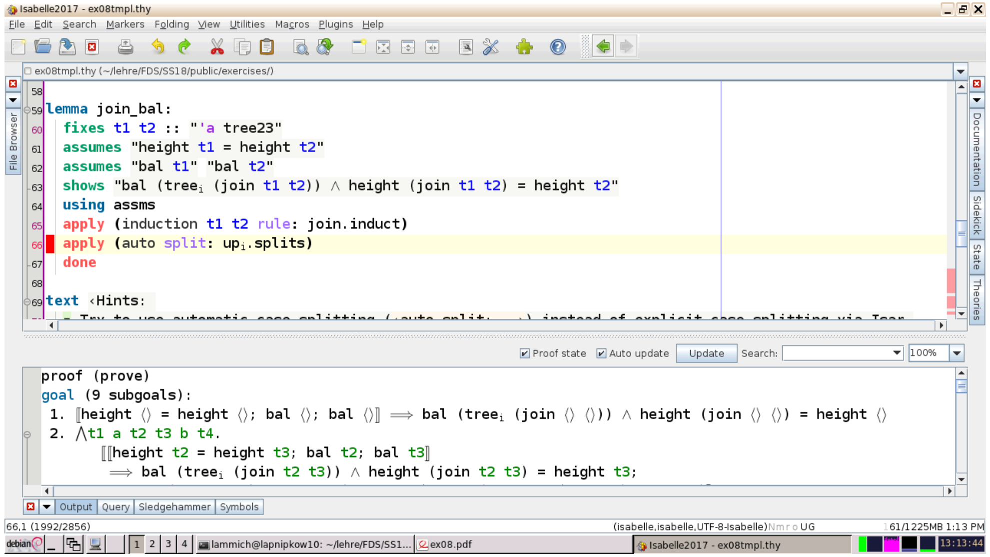

Isabelle2017 - ex08tmpl.thy
done
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: upᵢ.splits)
done
proof (prove)
goal (1 subgoal):
1. bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2

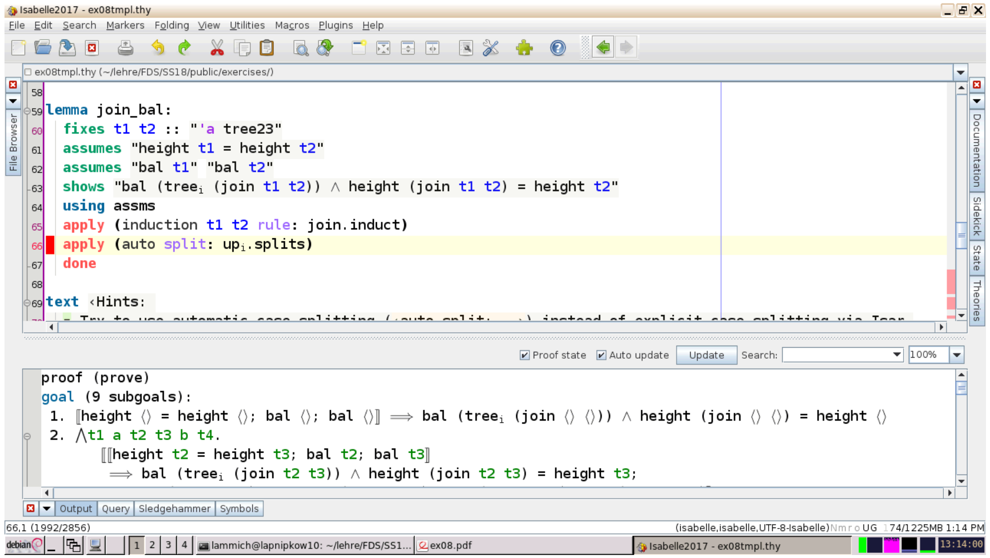

Isabelle2017 - ex08tmpl.thy
done
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: upᵢ.splits)
done
proof (prove)
goal (1 subgoal):
1. bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2

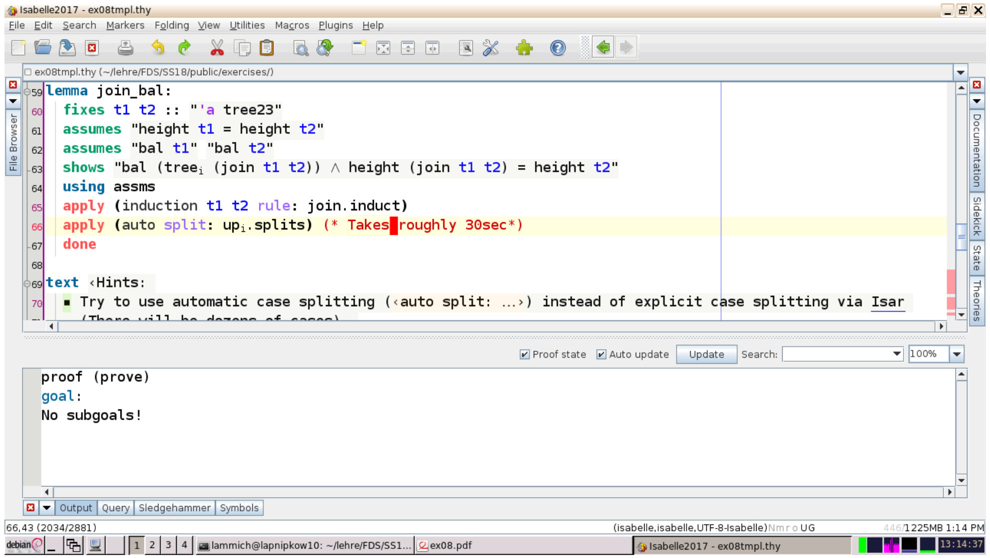

Isabelle2017 - ex08tmpl.thy
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: upᵢ.splits)
done
text ‹Hints:
proof (prove)
goal (1 subgoal):
1. bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2

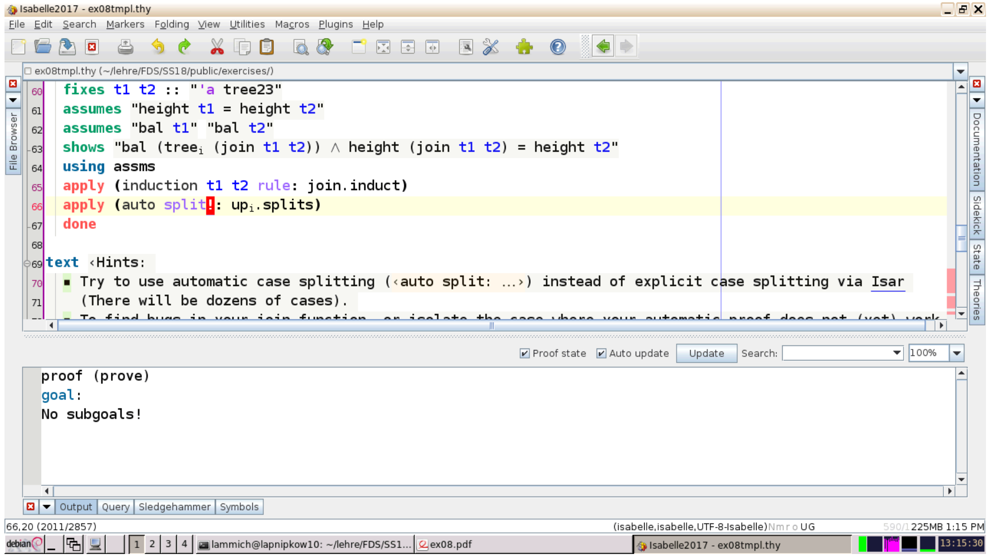

Isabelle2017 - ex08tmpl.thy
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: upᵢ.splits)
done
text ‹Hints:
proof (prove)
goal (1 subgoal):
1. bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2


Isabelle2017 - ex08tmpl.thy
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: upᵢ.splits)
done
text ‹Hints:
proof (prove)
goal (1 subgoal):
1. bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2

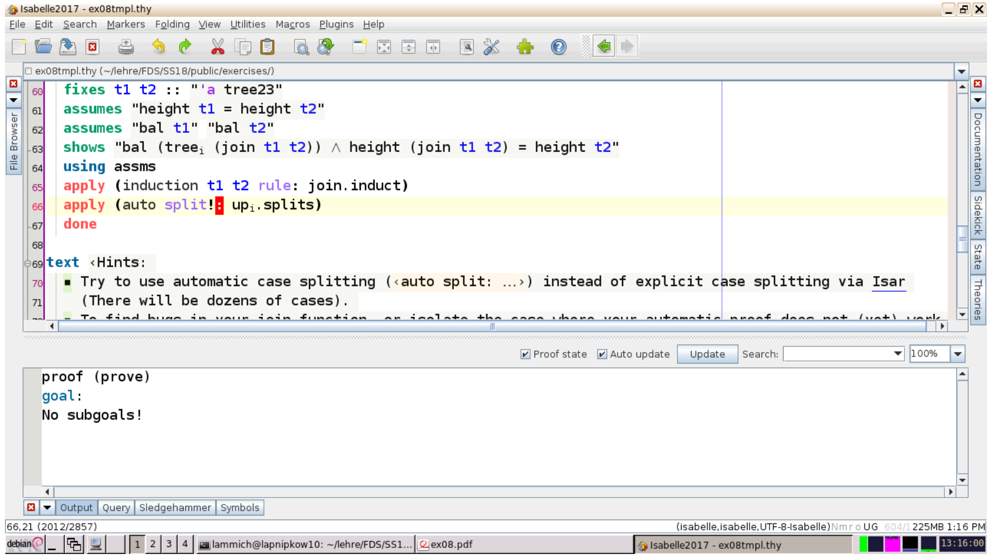

Isabelle2017 - ex08tmpl.thy
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: upᵢ.splits)
done
text ‹Hints:
proof (prove)
goal (9 subgoals):
1. ⟦height ⟨⟩ = height ⟨⟩; bal ⟨⟩; bal ⟨⟩⟧ ⟹ bal (treeᵢ (join ⟨⟩ ⟨⟩)) ∧ height (join ⟨⟩ ⟨⟩) = height ⟨⟩
2. ⋀t1 a t2 t3 b t4.
⟦height t2 = height t3; bal t2; bal t3⟧
⟹ bal (treeᵢ (join t2 t3)) ∧ height (join t2 t3) = height t3;


Isabelle2017 - ex08tmpl.thy
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: upᵢ.splits)
done
text ‹Hints:
proof (prove)
goal (9 subgoals):
1. ⟦height ⟨⟩ = height ⟨⟩; bal ⟨⟩; bal ⟨⟩⟧ ⟹ bal (treeᵢ (join ⟨⟩ ⟨⟩)) ∧ height (join ⟨⟩ ⟨⟩) = height ⟨⟩
2. ⋀t1 a t2 t3 b t4.
⟦height t2 = height t3; bal t2; bal t3⟧
⟹ bal (treeᵢ (join t2 t3)) ∧ height (join t2 t3) = height t3;

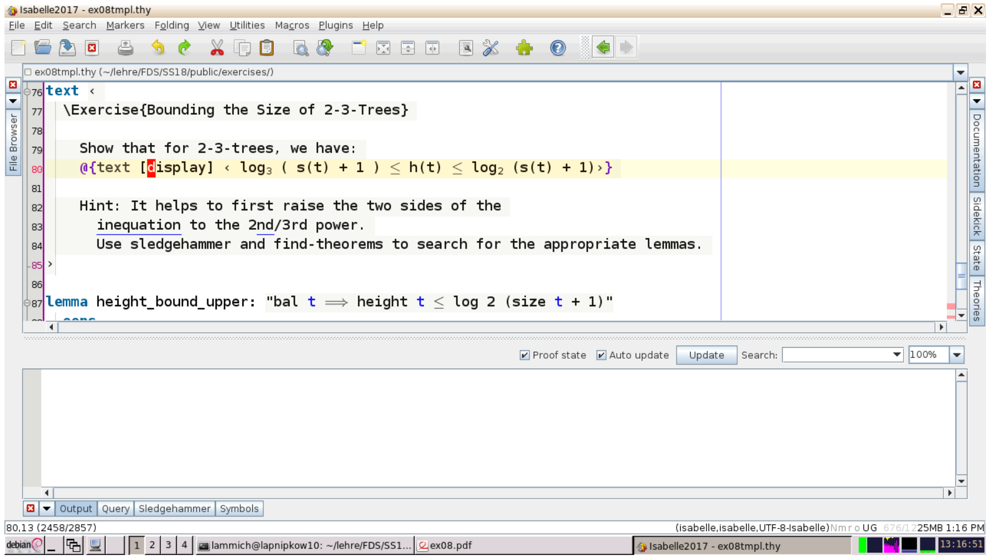

Isabelle2017 - ex08tmpl.thy
lemma join_bal:
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split: upᵢ.splits) (* Takes roughly 30sec*)
done
text ‹Hints:
▪ Try to use automatic case splitting (‹auto split: ...›) instead of explicit case splitting via Isar
proof (prove)
goal:
No subgoals!



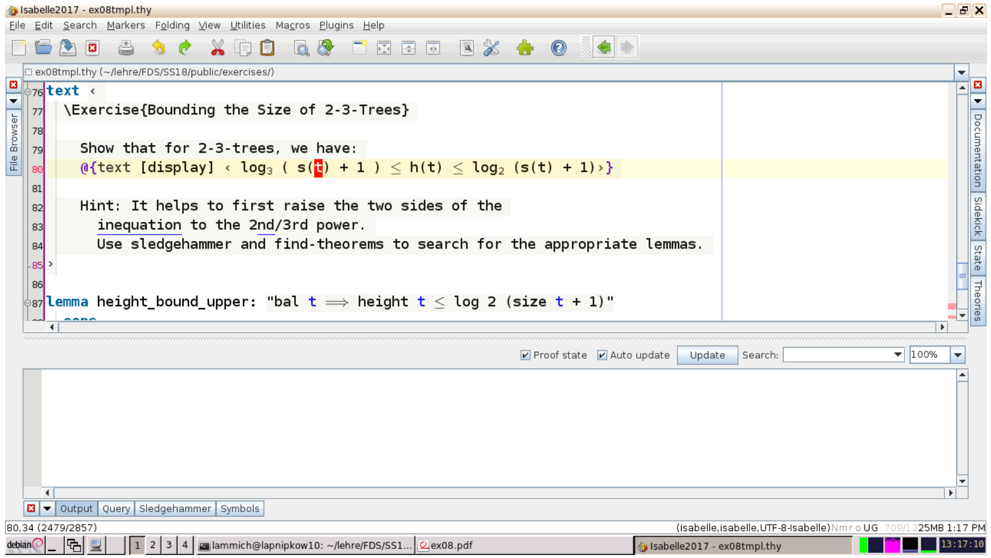
Isabelle2017 - ex08tmpl.thy
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split!: upᵢ.splits)
done
text ‹Hints:
▪ Try to use automatic case splitting (‹auto split: ...›) instead of explicit case splitting via Isar
(There will be dozens of cases).
proof (prove)
goal:
No subgoals!



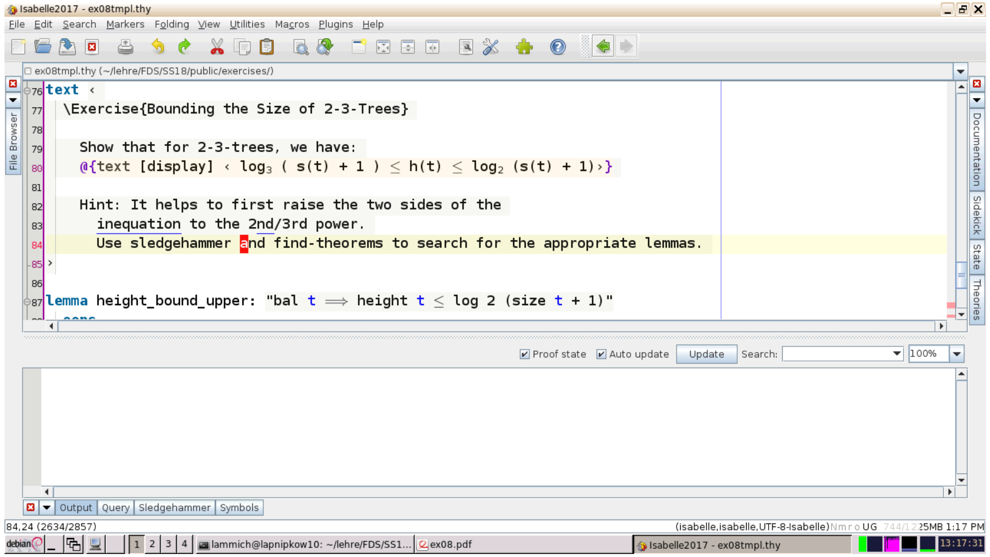
Isabelle2017 - ex08tmpl.thy
fixes t1 t2 :: "'a tree23"
assumes "height t1 = height t2"
assumes "bal t1" "bal t2"
shows "bal (treeᵢ (join t1 t2)) ∧ height (join t1 t2) = height t2"
using assms
apply (induction t1 t2 rule: join.induct)
apply (auto split!: upᵢ.splits)
done
text ‹Hints:
▪ Try to use automatic case splitting (‹auto split: ...›) instead of explicit case splitting via Isar
(There will be dozens of cases).
proof (prove)
goal:
No subgoals!

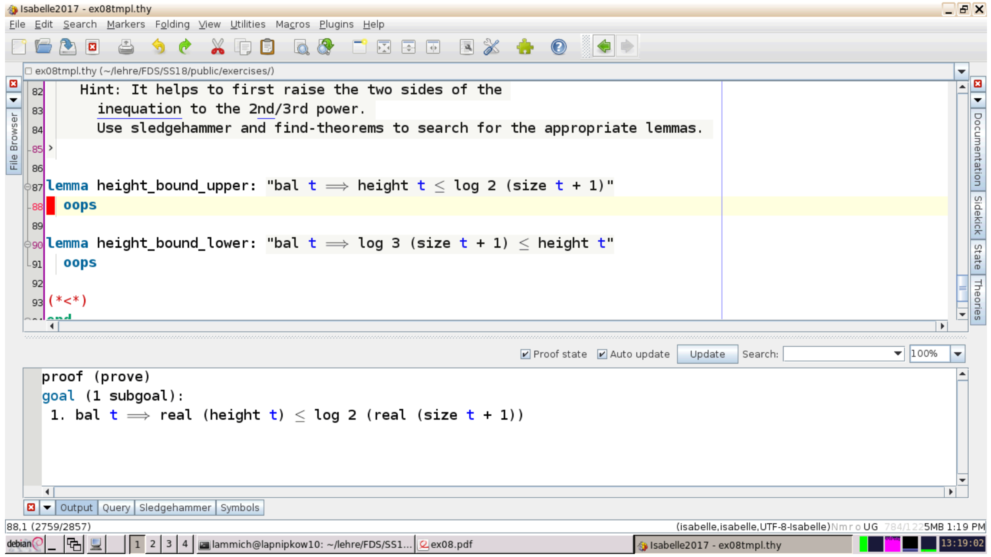


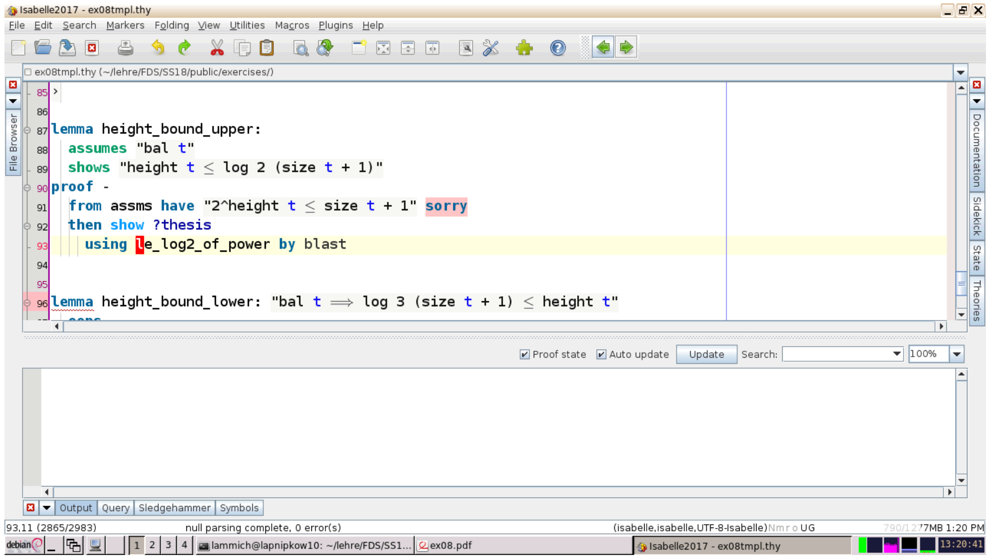
Isabelle2017 - ex08tmpl.thy
text ‹
\Exercise{Bounding the Size of 2-3-Trees}
Show that for 2-3-trees, we have:
@{text [display] ‹ log₃ (s(t) + 1) ≤ h(t) ≤ log₂ (s(t) + 1)›}
Hint: It helps to first raise the two sides of the
inequation to the 2nd/3rd power.
Use sledgehammer and find-theorems to search for the appropriate lemmas.
›
lemma height_bound_upper: "bal t ⟹ height t ≤ log 2 (size t + 1)"

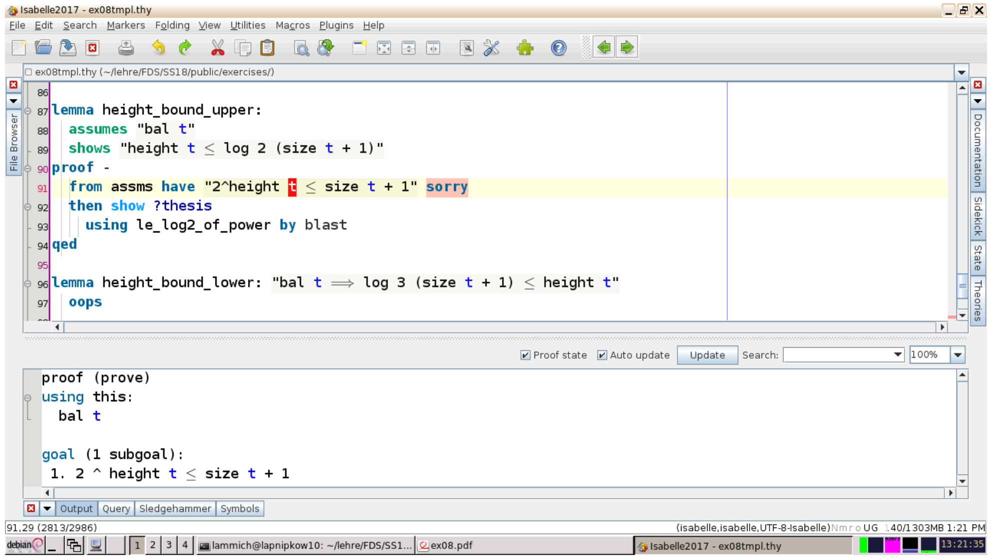


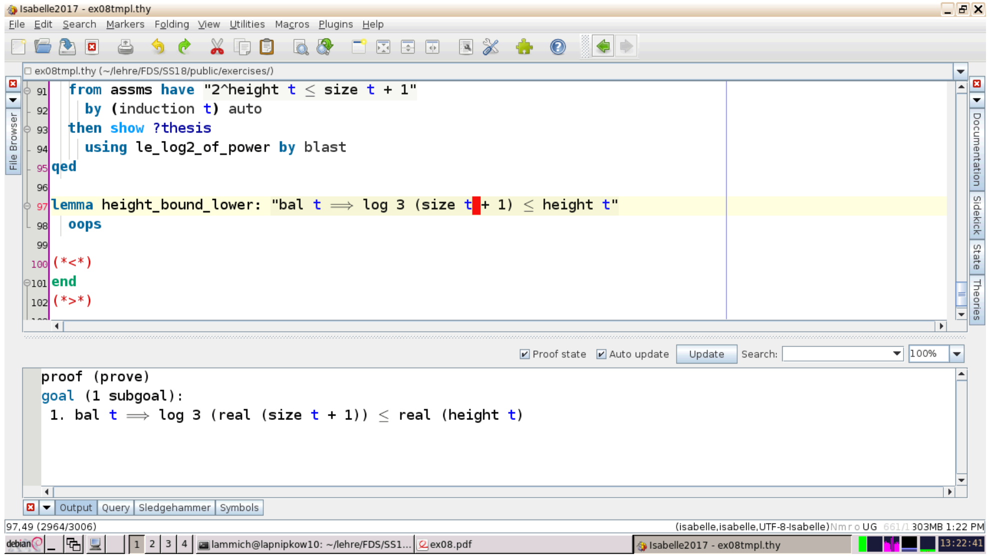
Isabelle2017 - ex08tmpl.thy
text ‹
\Exercise{Bounding the Size of 2-3-Trees}
Show that for 2-3-trees, we have:
@{text [display] ‹ log₃ (s(t) + 1) ≤ h(t) ≤ log₂ (s(t) + 1)›}
Hint: It helps to first raise the two sides of the
inequation to the 2nd/3rd power.
Use sledgehammer and find-theorems to search for the appropriate lemmas.
›
lemma height_bound_upper: "bal t ⟹ height t ≤ log 2 (size t + 1)"

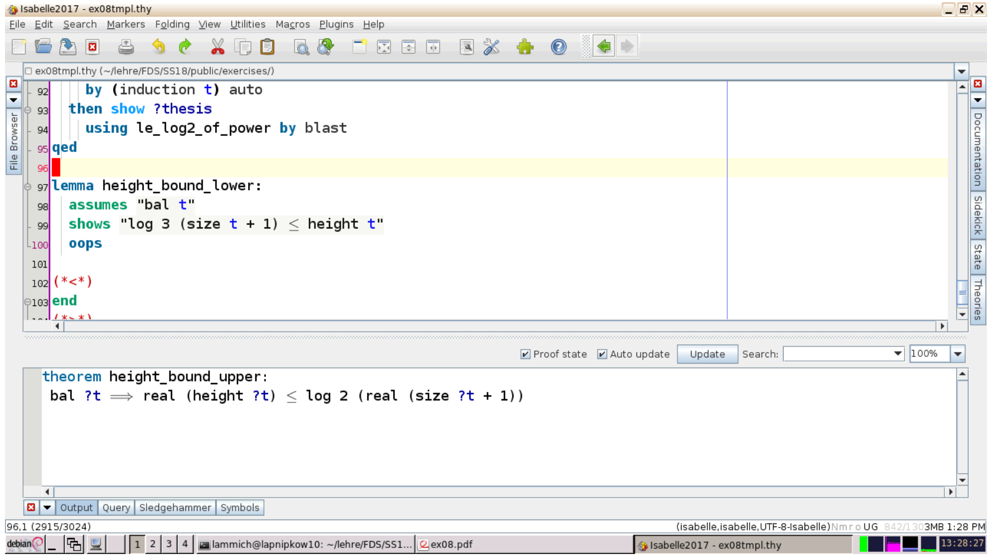

Isabelle2017 - ex08tmpl.thy
text ‹
\Exercise{Bounding the Size of 2-3-Trees}
Show that for 2-3-trees, we have:
@{text [display] ‹ log₃ (s(t) + 1) ≤ h(t) ≤ log₂ (s(t) + 1)›}
Hint: It helps to first raise the two sides of the
inequation to the 2nd/3rd power.
Use sledgehammer and find-theorems to search for the appropriate lemmas.
›
lemma height_bound_upper: "bal t ⟹ height t ≤ log 2 (size t + 1)"
Output Query Sledgehammer Symbols

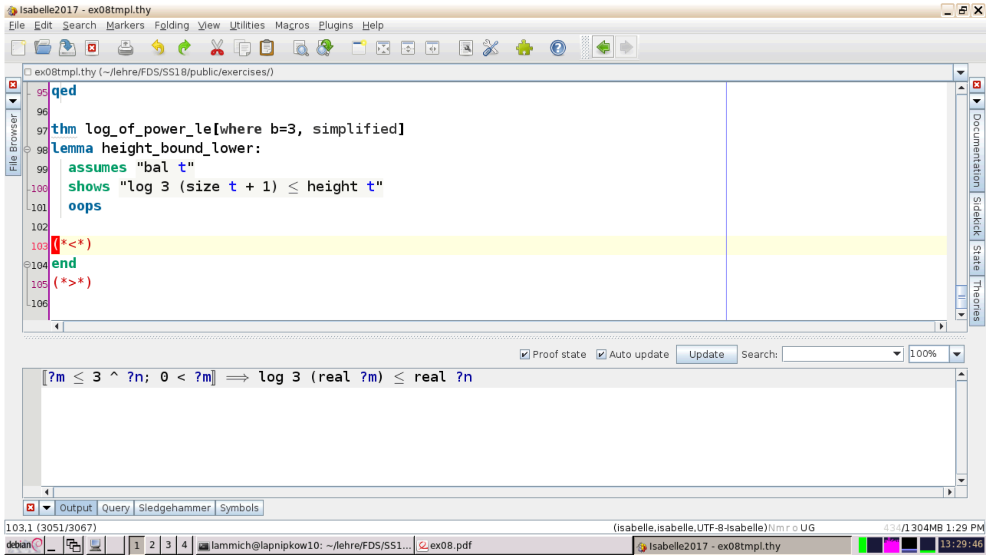

Isabelle2017 - ex08tmpl.thy
Hint: It helps to first raise the two sides of the
inequation to the 2nd/3rd power.
Use sledgehammer and find-theorems to search for the appropriate lemmas.
›
lemma height_bound_upper: "bal t ⟹ height t ≤ log 2 (size t + 1)"
oops
lemma height_bound_lower: "bal t ⟹ log 3 (size t + 1) ≤ height t"
oops
(*<*)
proof (prove)
goal (1 subgoal):
1. bal t ⟹ real (height t) ≤ log 2 (real (size t + 1))
Output Query Sledgehammer Symbols

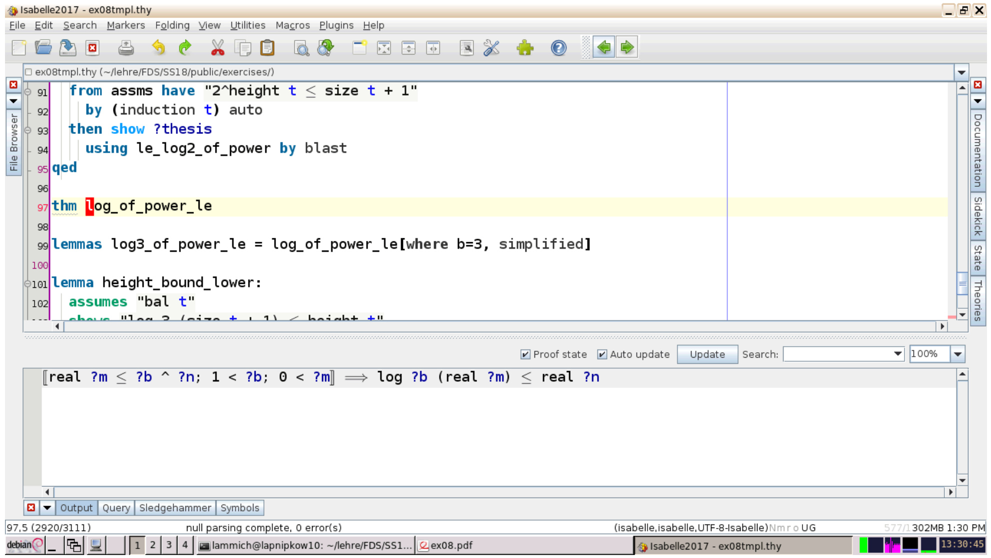

Isabelle2017 - ex08tmpl.thy
›
lemma height_bound_upper:
assumes "bal t"
shows "height t ≤ log 2 (size t + 1)"
proof -
from assms have "2^height t ≤ size t + 1" sorry
then show ?thesis
using le_log2_of_power by blast
lemma height_bound_lower: "bal t ⟹ log 3 (size t + 1) ≤ height t"
Output Query Sledgehammer Symbols

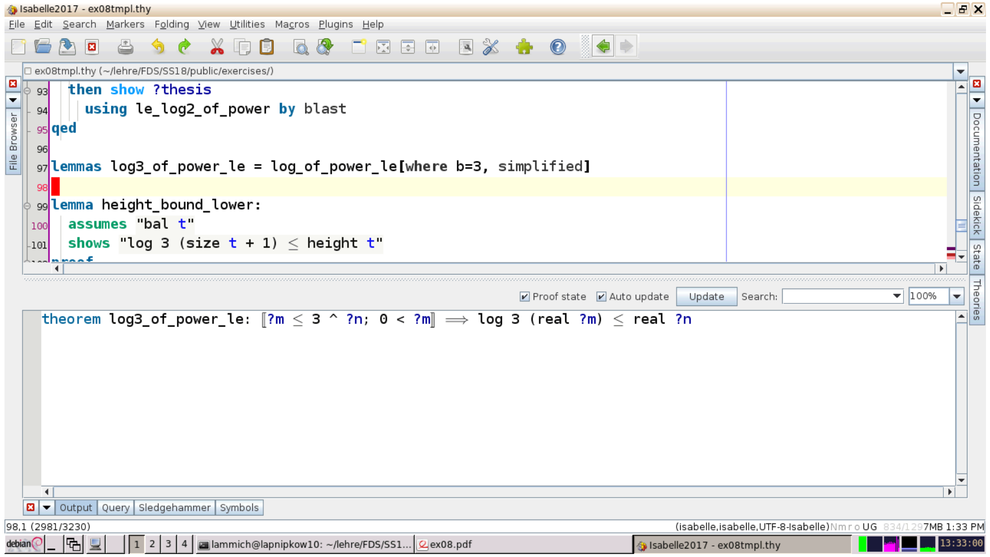

Isabelle2017 - ex08tmpl.thy
lemma height_bound_upper:
assumes "bal t"
shows "height t ≤ log 2 (size t + 1)"
proof -
from assms have "2^height t ≤ size t + 1" sorry
then show ?thesis
using le_log2_of_power by blast
qed
lemma height_bound_lower: "bal t ⟹ log 3 (size t + 1) ≤ height t"
oops
proof (prove)
using this:
bal t
goal (1 subgoal):
1. 2 ^ height t ≤ size t + 1
Output Query Sledgehammer Symbols

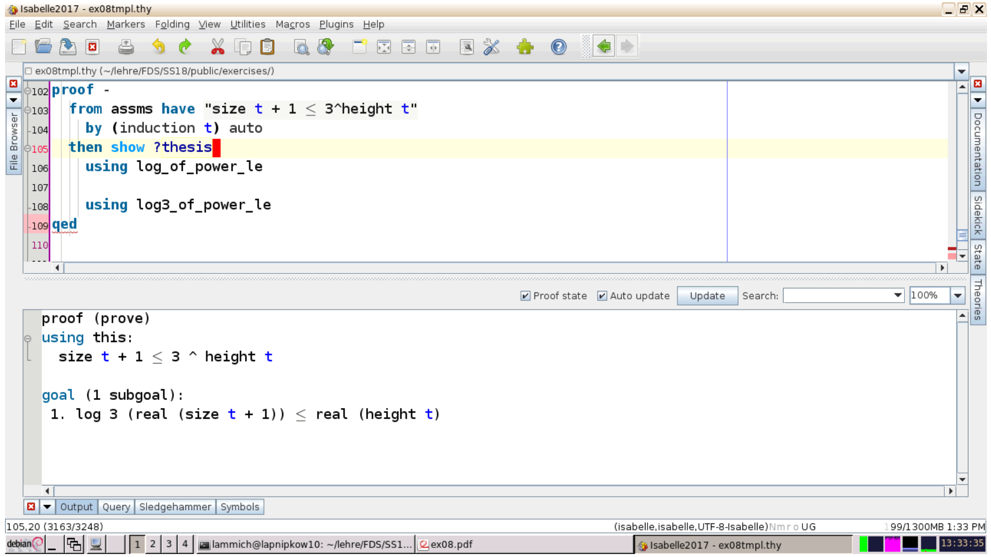

Isabelle2017 - ex08tmpl.thy
from assms have "2^height t ≤ size t + 1"
by (induction t) auto
then show ?thesis
using le_log2_of_power by blast
qed
lemma height_bound_lower: "bal t ⟹ log 3 (size t + 1) ≤ height t"
oops
(*<*)
end
(*>*)
proof (prove)
goal (1 subgoal):
1. bal t ⟹ log 3 (real (size t + 1)) ≤ real (height t)

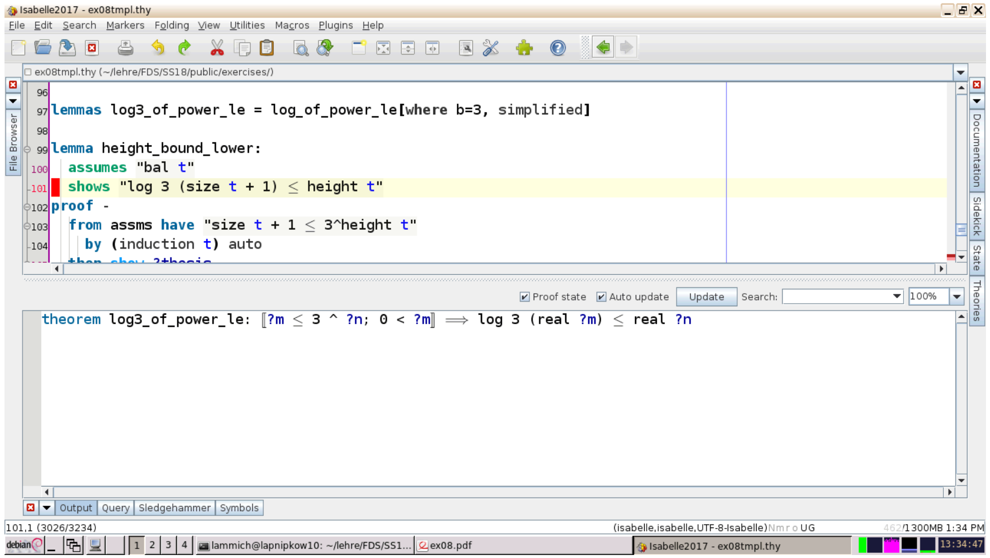

Isabelle2017 - ex08tmpl.thy
by (induction t) auto
then show ?thesis
using le_log2_of_power by blast
qed
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
oops
(*<*)
end
theorem height_bound_upper:
bal ?t ⟹ real (height ?t) ≤ log 2 (real (size ?t + 1))

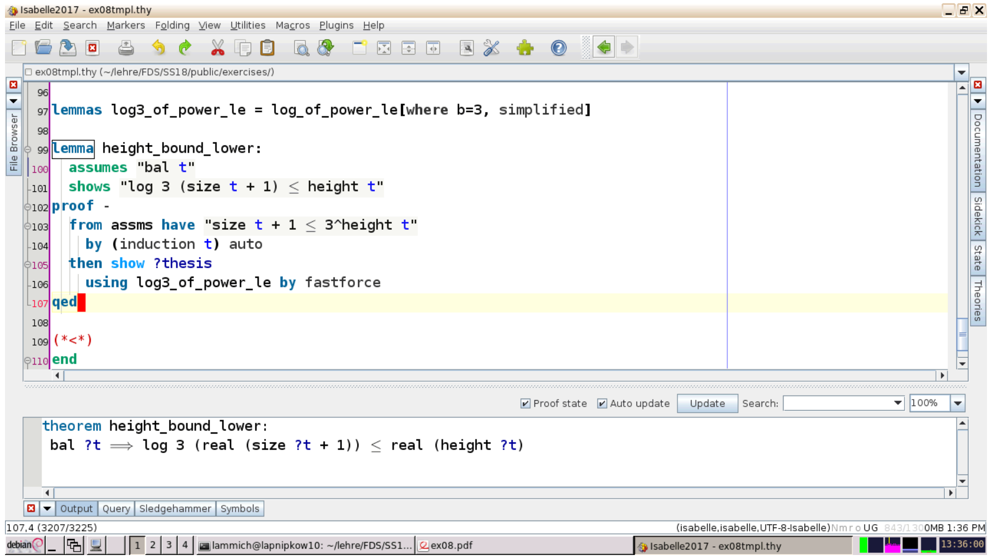

Isabelle2017 - ex08tmpl.thy
qed
thm log_of_power_le[where b=3, simplified]
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
oops
(*<*)
end
(*>*)
⟦?m ≤ 3 ^ ?n; 0 < ?m⟧ ⟹ log 3 (real ?m) ≤ real ?n

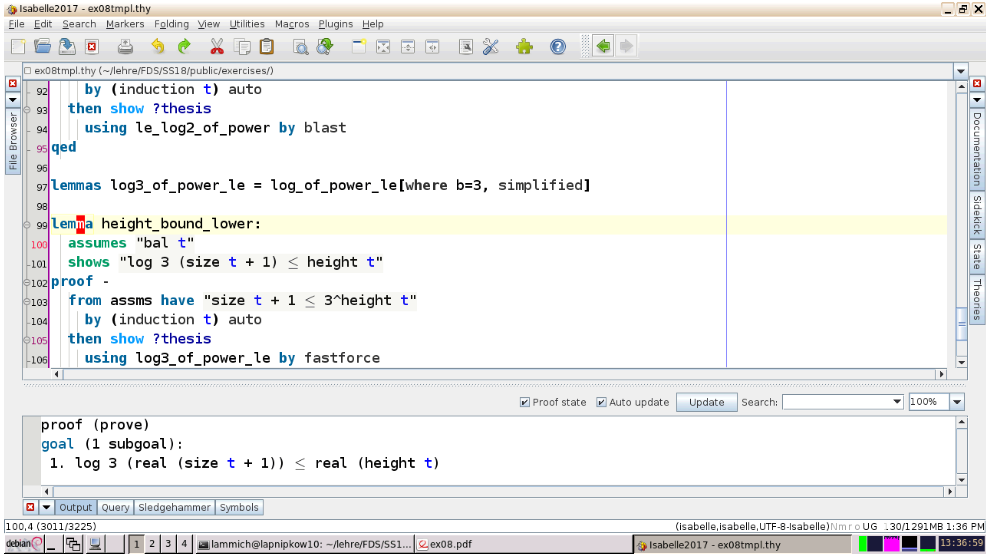

Isabelle2017 - ex08tmpl.thy
from assms have "2^height t ≤ size t + 1"
by (induction t) auto
then show ?thesis
using le_log2_of_power by blast
qed
thm log_of_power_le
lemmas log3_of_power_le = log_of_power_le[where b=3, simplified]
lemma height_bound_lower:
assumes "bal t"
⟦real ?m ≤ ?b ^ ?n; 1 < ?b; 0 < ?m⟧ ⟹ log ?b (real ?m) ≤ real ?n

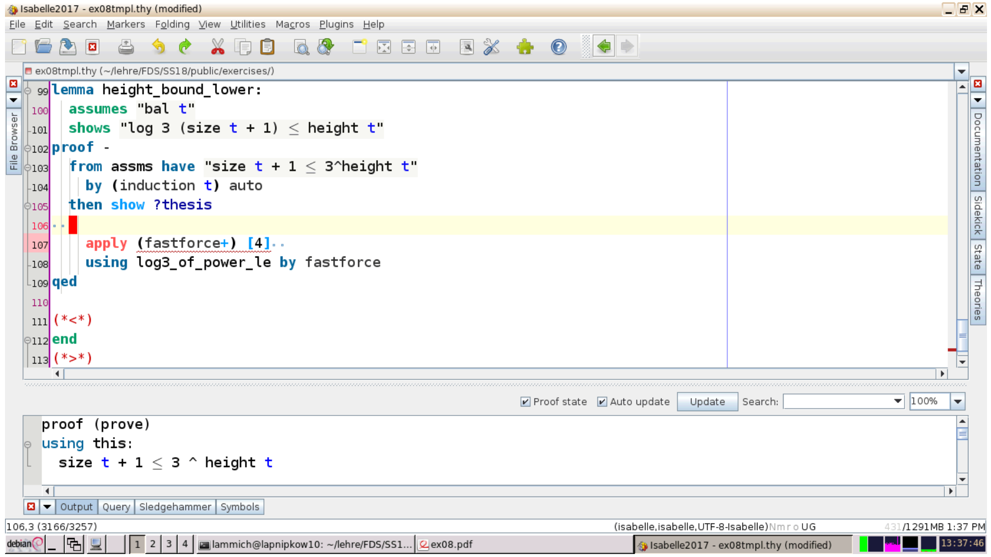

Isabelle2017 - ex08tmpl.thy
then show ?thesis
using le_log2_of_power by blast
qed
lemmas log3_of_power_le = log_of_power_le[where b=3, simplified]
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
theorem log3_of_power_le: ⟦?m ≤ 3 ^ ?n; 0 < ?m⟧ ⟹ log 3 (real ?m) ≤ real ?n

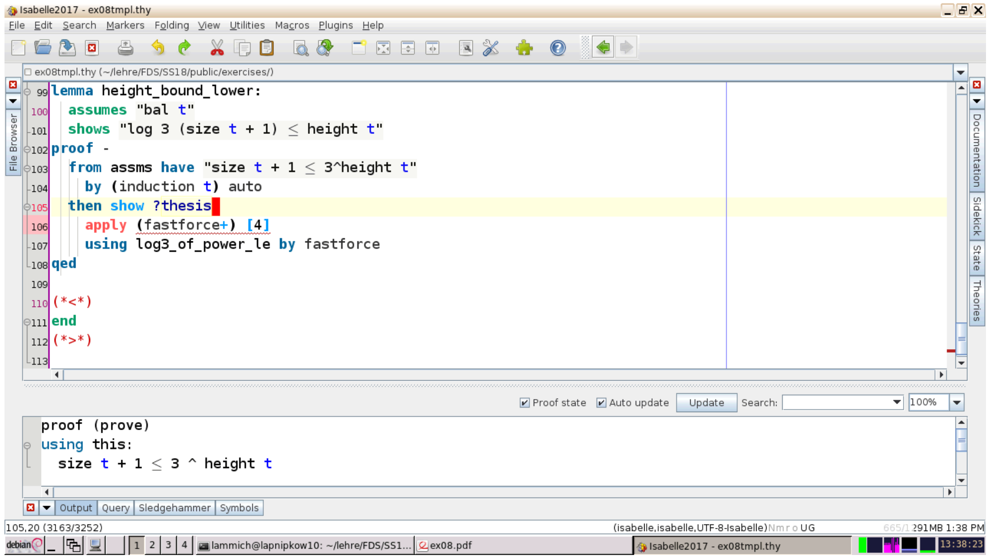

Isabelle2017 - ex08tmpl.thy
proof -
from assms have "size t + 1 ≤ 3^height t"
by (induction t) auto
then show ?thesis
using log_of_power_le
using log3_of_power_le
qed
proof (prove)
using this:
size t + 1 ≤ 3 ^ height t
goal (1 subgoal):
1. log 3 (real (size t + 1)) ≤ real (height t)

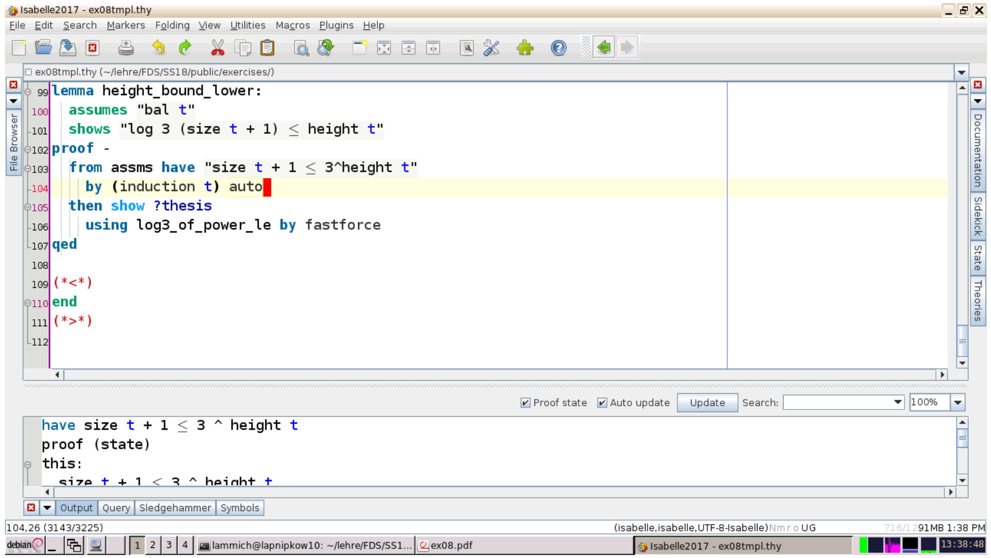

Isabelle2017 - ex08tmpl.thy
lemmas log3_of_power_le = log_of_power_le[where b=3, simplified]
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
by (induction t) auto
theorem log3_of_power_le: ⟦?m ≤ 3 ^ ?n; 0 < ?m⟧ ⟹ log 3 (real ?m) ≤ real ?n

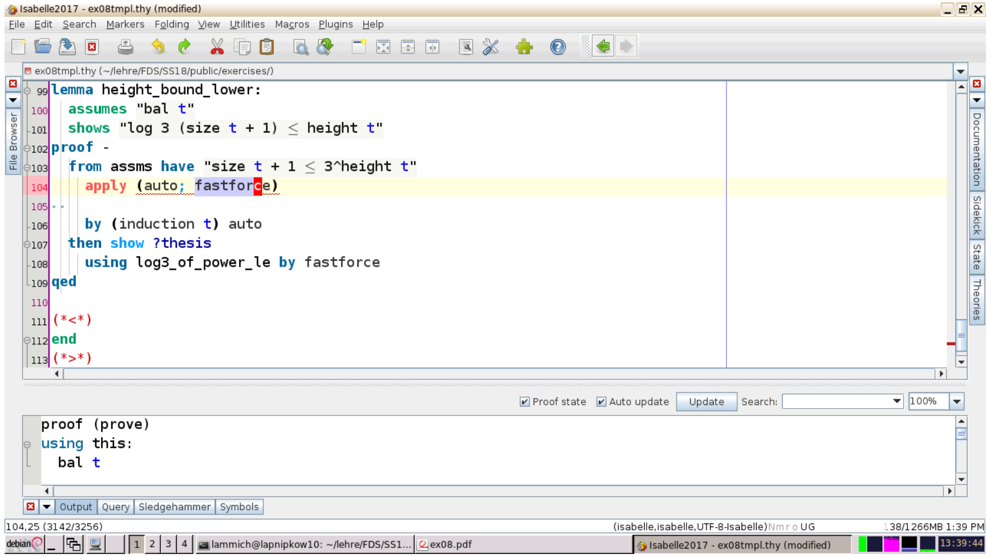

Isabelle2017 - ex08tmpl.thy
lemmas log3_of_power_le = log_of_power_le[where b=3, simplified]
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
qed
(*<*)
end
theorem height_bound_lower:
bal ?t ⟹ log 3 (real (size ?t + 1)) ≤ real (height ?t)

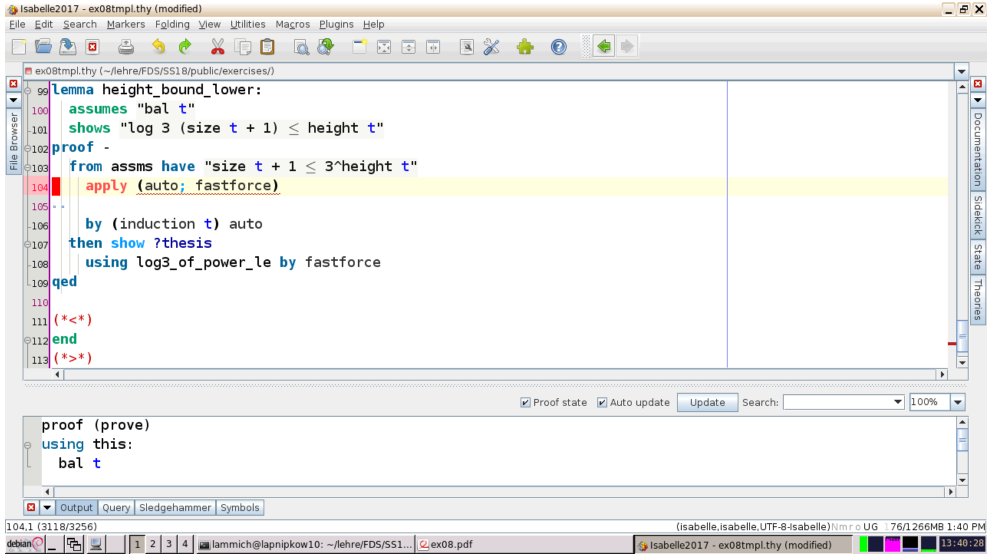

Isabelle2017 - ex08tmpl.thy
by (induction t) auto
then show ?thesis
using le_log2_of_power by blast
qed
lemmas log3_of_power_le = log_of_power_le[where b=3, simplified]
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
proof (prove)
goal (1 subgoal):
1. log 3 (real (size t + 1)) ≤ real (height t)

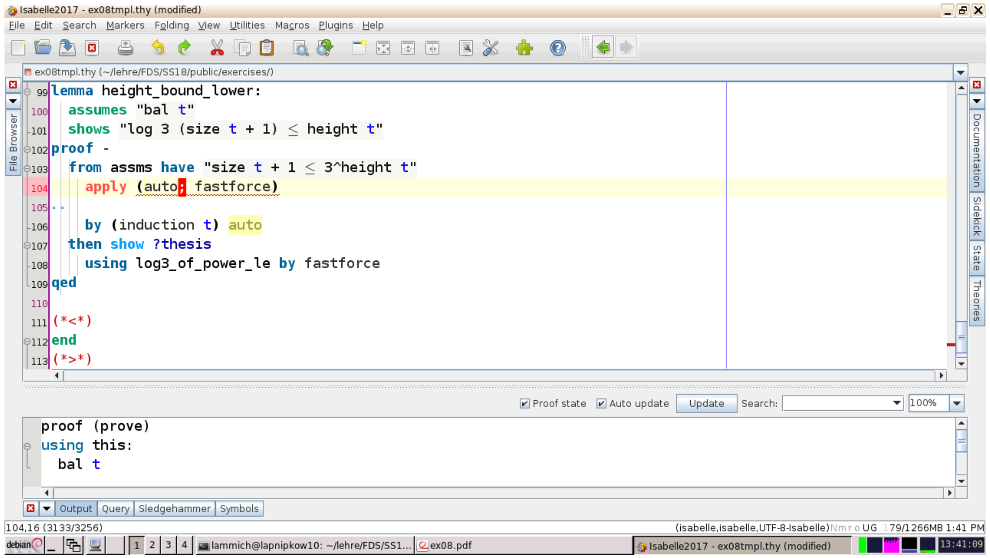

Isabelle2017 - ex08tmpl.thy (modified)
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
by (induction t) auto
then show ?thesis
apply (fastforce+) [4]
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
proof (prove)
using this:
size t + 1 ≤ 3 ^ height t

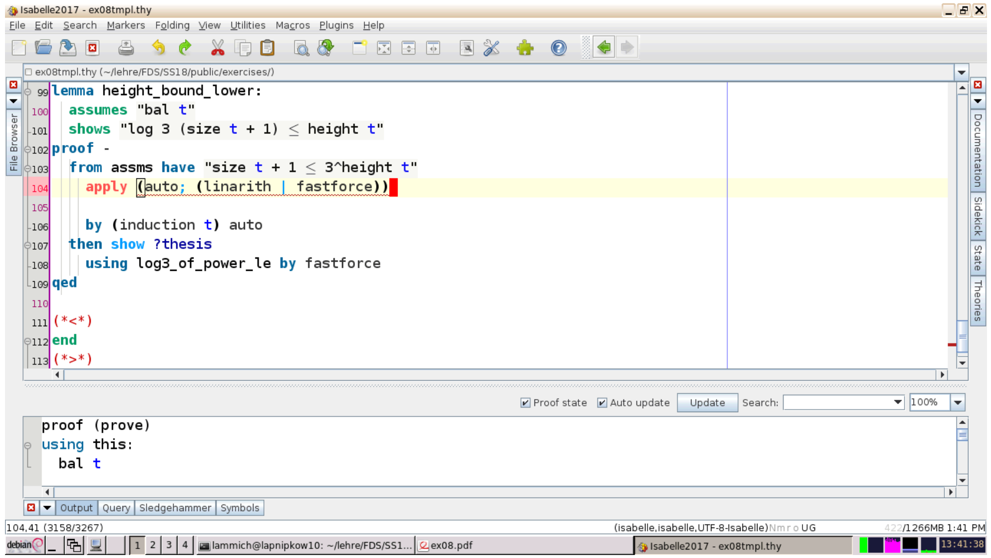

Isabelle2017 - ex08tmpl.thy
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
by (induction t) auto
then show ?thesis
apply (fastforce+) [4]
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
proof (prove)
using this:
size t + 1 ≤ 3 ^ height t

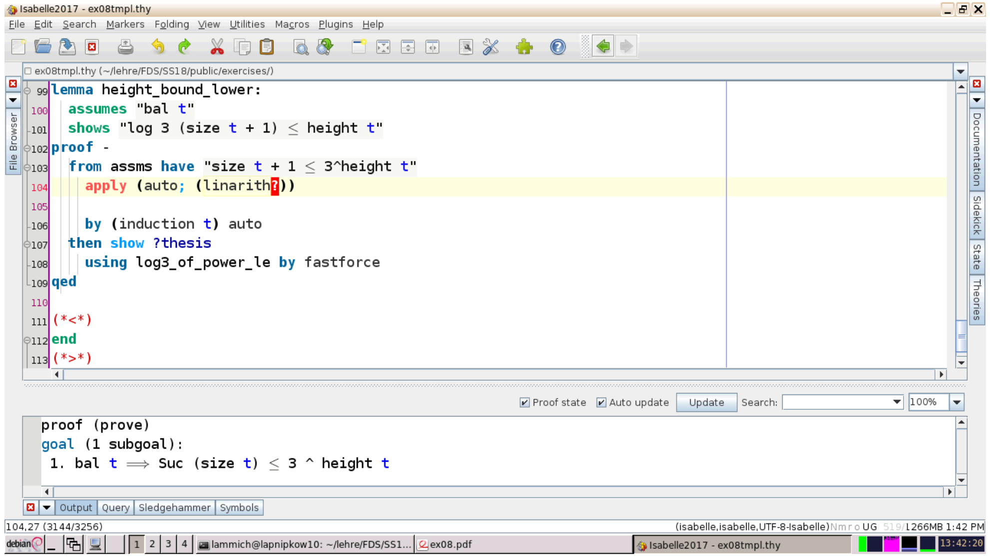

Isabelle2017 - ex08tmpl.thy
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
have size t + 1 ≤ 3 ^ height t
proof (state)
this:

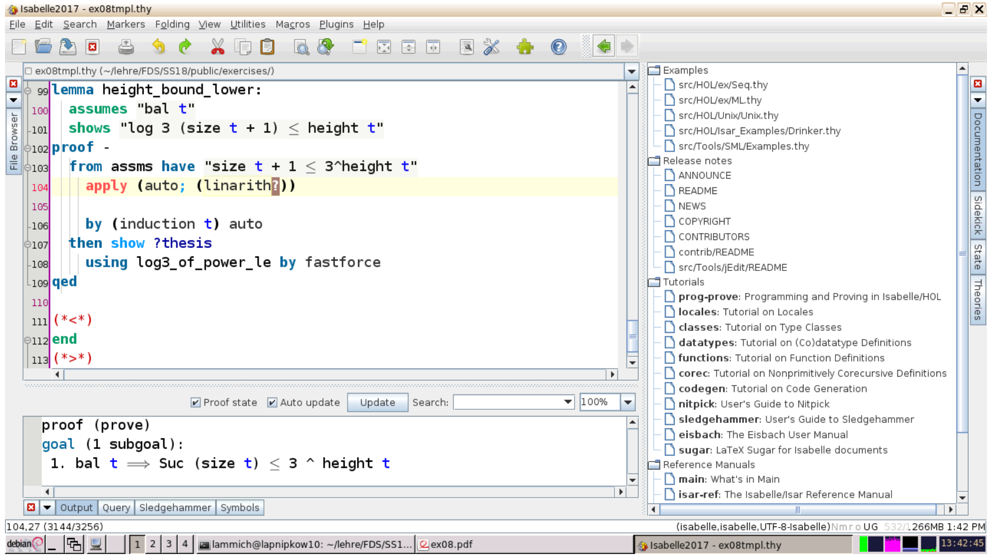

Isabelle2017 - ex08tmpl.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
apply (auto; fastforce)
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
Proof state
Auto update
Update
Search:
100%
proof (prove)
using this:
bal t
Output
Query
Sledgehammer
Symbols
104,25 (3142/3256)

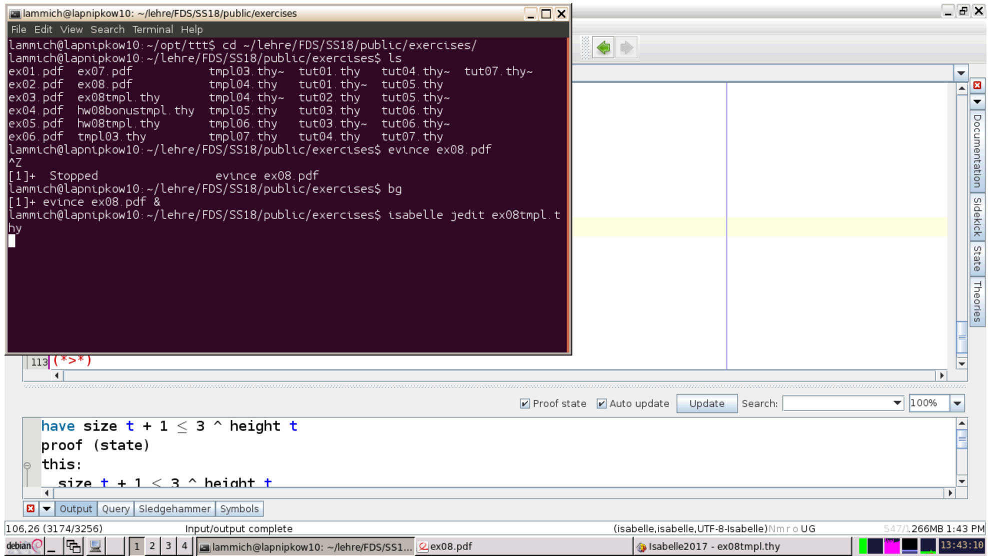

Isabelle2017 - ex08tmpl.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
apply (auto; fastforce)
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
Proof state
Auto update
Update
Search:
100%
proof (prove)
using this:
bal t
Output
Query
Sledgehammer
Symbols
104,1 (3118/3256)

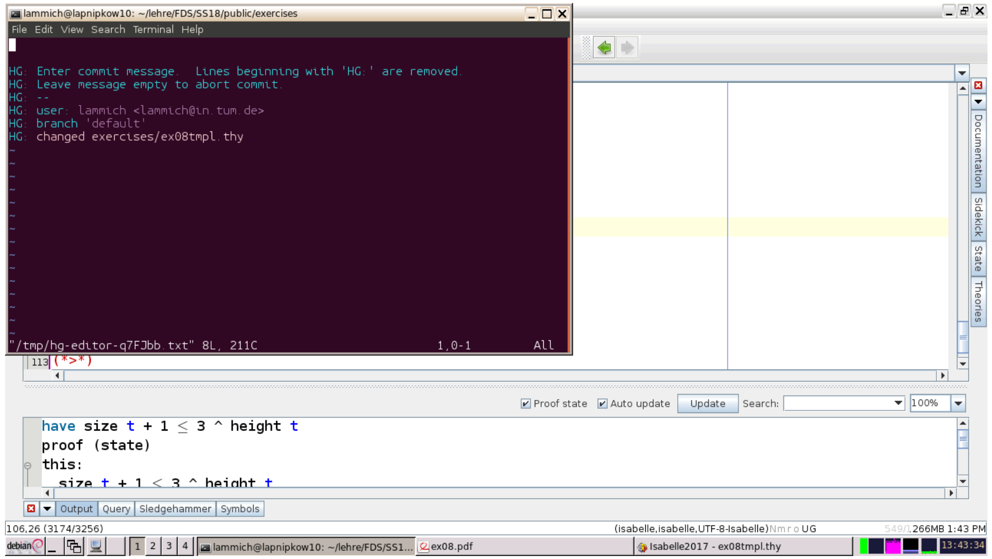

Isabelle2017 - ex08tmpl.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
apply (auto; fastforce)
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
Proof state
Auto update
Update
Search:
100%
proof (prove)
using this:
bal t
Output
Query
Sledgehammer
Symbols
104,16 (3133/3256)

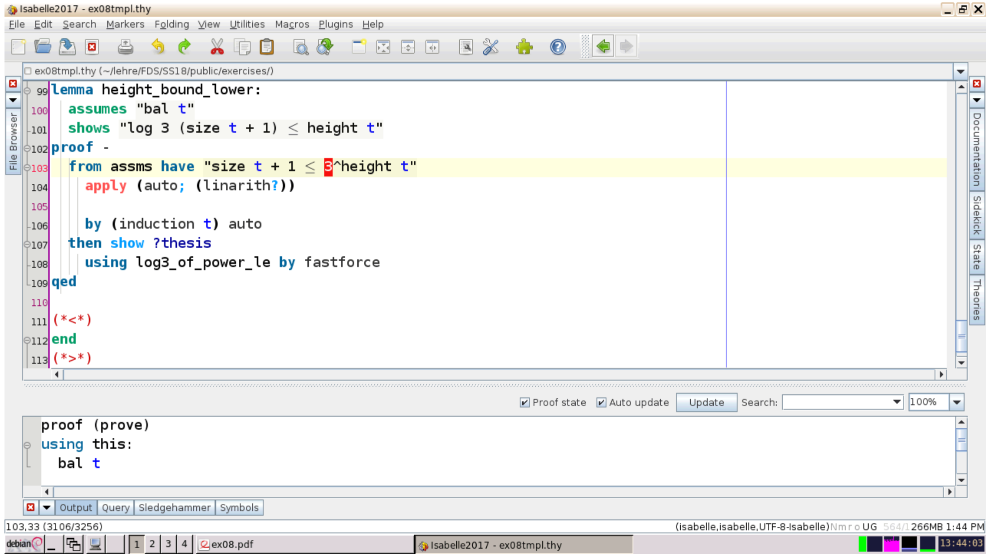

Isabelle2017 - ex08tmpl.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
apply (auto; (linarith | fastforce))
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
Proof state
Auto update
Update
Search:
100%
proof (prove)
using this:
bal t
Output
Query
Sledgehammer
Symbols
104,41 (3158/3267)


Isabelle2017 - ex08tmpl.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
apply (auto; (linarith?))
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
Proof state
Auto update
Update
Search:
100%
proof (prove)
goal (1 subgoal):
1. bal t ⟹ Suc (size t) ≤ 3 ^ height t
Output Query Sledgehammer Symbols
104,27 (3144/3256)
(isabelle,isabelle,UTF-8-Isabelle)
1266MB 1:42 PM
lammich@lapnipkow10: ~/lehre/FDS/SS1...
ex08.pdf
Isabelle2017 - ex08tmpl.thy
13:42:20


Isabelle2017 - ex08tmpl.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
apply (auto; (linarith?))
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
Examples
src/HOL/ex/Seq.thy
src/HOL/ex/ML.thy
src/HOL/Unix/Unix.thy
src/HOL/Isar_Examples/Drinker.thy
src/Tools/SML/Examples.thy
Release notes
ANNOUNCE
README
NEWS
COPYRIGHT
CONTRIBUTORS
contrib/README
src/Tools/jEdit/README
Tutorials
prog-prove: Programming and Proving in Isabelle/HOL
locales: Tutorial on Locales
classes: Tutorial on Type Classes
datatypes: Tutorial on (Co)datatype Definitions
functions: Tutorial on Function Definitions
corec: Tutorial on Nonprimitively Corecursive Definitions
codegen: Tutorial on Code Generation
nitpick: User's Guide to Nitpick
sledgehammer: User's Guide to Sledgehammer
eisbach: The Eisbach User Manual
sugar: LaTeX Sugar for Isabelle documents
Reference Manuals
main: What's in Main
isar-ref: The Isabelle/Isar Reference Manual
proof (prove)
goal (1 subgoal):
1. bal t ⟹ Suc (size t) ≤ 3 ^ height t
Output Query Sledgehammer Symbols
104,27 (3144/3256)
1266MB 1:42 PM
13:42:45


lammich@lapnipkow10: ~/lehre/FDS/SS18/public/exercises
File Edit View Search Terminal Help
lammich@lapnipkow10:~/opt/ttt$ cd ~/lehre/FDS/SS18/public/exercises/
lammich@lapnipkow10:~/lehre/FDS/SS18/public/exercises$ ls
ex01.pdf ex07.pdf tmpl03.thy~ tut01.thy tut04.thy~ tut07.thy~
ex02.pdf ex08.pdf tmpl04.thy tut01.thy~ tut05.thy
ex03.pdf ex08tmpl.thy tmpl04.thy~ tut02.thy tut05.thy~
ex04.pdf hw08bonustmpl.thy tmpl05.thy tut03.thy tut06.thy
ex05.pdf hw08tmpl.thy tmpl06.thy tut03.thy~ tut06.thy~
ex06.pdf tmpl03.thy tmpl07.thy tut04.thy tut07.thy
lammich@lapnipkow10:~/lehre/FDS/SS18/public/exercises$ evince ex08.pdf
^Z
[1]+ Stopped evince ex08.pdf
lammich@lapnipkow10:~/lehre/FDS/SS18/public/exercises$ bg
[1]+ evince ex08.pdf &
lammich@lapnipkow10:~/lehre/FDS/SS18/public/exercises$ isabelle jedit ex08tmpl.thy
have size t + 1 ≤ 3 ^ height t
proof (state)
this:
size t + 1 ≤ 3 ^ height t
106,26 (3174/3256)
Input/output complete
1266MB 1:43 PM
13:43:10


lammich@lapnipkow10: ~/lehre/FDS/SS18/public/exercises
File Edit View Search Terminal Help
HG: Enter commit message. Lines beginning with 'HG:' are removed.
HG: Leave message empty to abort commit.
HG: --
HG: user: Lammich <lammich@in.tum.de>
HG: branch 'default'
HG: changed exercises/ex08tmpl.thy
"/tmp/hg-editor-q7FJbb.txt" 8L, 211C 1,0-1 All
have size t + 1 ≤ 3 ^ height t
proof (state)
this:
size t + 1 ≤ 3 ^ height t
106,26 (3174/3256)
1266MB 1:43 PM
13:43:34


Isabelle2017 - ex08tmpl.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
ex08tmpl.thy (~/lehre/FDS/SS18/public/exercises/)
lemma height_bound_lower:
assumes "bal t"
shows "log 3 (size t + 1) ≤ height t"
proof -
from assms have "size t + 1 ≤ 3^height t"
apply (auto; (linarith?))
by (induction t) auto
then show ?thesis
using log3_of_power_le by fastforce
qed
(*<*)
end
(*>*)
Proof state
Auto update
Update
Search:
100%
proof (prove)
using this:
bal t
Output
Query
Sledgehammer
Symbols
103,33 (3106/3256)
(isabelle,isabelle,UTF-8-Isabelle)
266MB 1:44 PM
ex08.pdf
Isabelle2017 - ex08tmpl.thy
13:44:03