



Lecture 5:

- Let
- Booleans
- Induction

Declarations

val x: int = 5

val y: int = x + 2

fun f(x) = 2 * x + 3

Expressions

~~(val x: int = 5)~~

~~+~~

~~2~~

→ let

let
 val x:int=5
 val y = x+1
in
 x+y
end

these variables exist in

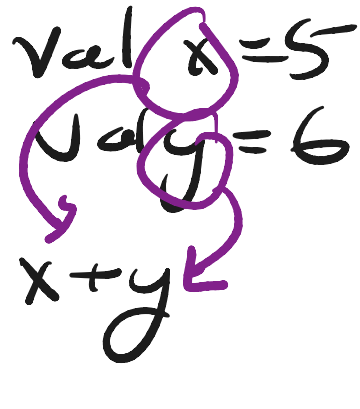
this expression (local variable)

↳ let val x=5
 in
 val y=5+1
 end
 5+y

↳ let val x=5
 in
 val y=6
 end
 5+y

↳ 5+6 ↳ 11

Val z = let val x = 5
 in val y = 6
 x + y
 → end



Val w = z + x error: no x in scope

"nearest enclosing binder"

let val $x^{:A} = e_1^{:A}$
in
 $e_2^{:T}$ Stand for expressions
end

$^{:T}$

let val $x = 5$
in
let val $y = 7$
in
 $x + y$
end
end
 $+ x$

To run `let val x = e1 in e2 end`

① run `e1` to produce a value

② substitute that value in for `x` in `e2`

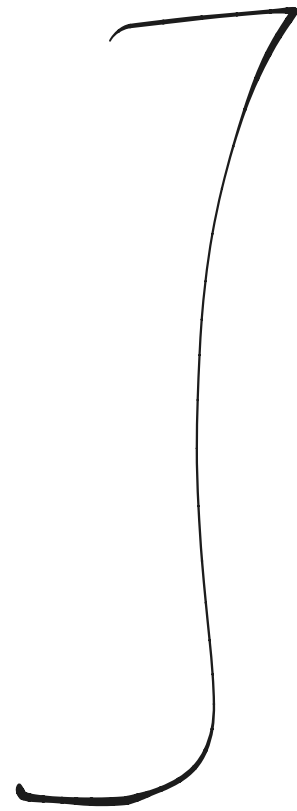
$(\text{let val } x = 5 \text{ in } x + x \text{ end}) \mapsto 5 + 5$
 $(\text{let val } x = 2 + 3 \text{ in } x + x \text{ end}) \mapsto \text{let val } x = 5 \text{ in } x + x \text{ end}$

fun f(x) =

let val y = x + y

in

end



"multiple
Statements"

let val x^{:int} = 4

in

Int.toString(x)^{:string}

end

: String

→ "4"

let val x = 4
in
let val y = x + 7
in
x + y
end
end

same
as

let val x = 4
val y = x + 7
in
x + y
end

val x = 4

val y = let val z = x + 1 in ... end

fun f ((x,y) : int * int) : int =

$2 * x + y$

$f(3,2) \mapsto 2 * 3 + 2$

same as

fun f (p : int * int) : int =

let val (x,y) = p

pair
pattern
in
val

end $(2 * x) + y$

$f(3,2)$
 \mapsto let val (x,y) = (2,3)
in $2 * x + 3$ end

SML is "garbage-collected"

let val x=5 in x+x end

↳ 5+5

memory for
x
gets
freed

Representing a time

12-hour clock

3:16 pm

type time12 = int * int * string
(* (h, m, p) where

$1 \leq h \leq 12$

$0 \leq m < 60$

$p = \text{"am"}$ or $p = \text{"pm"}$ *)

24-hour clock

15:16

type time24 =
int * int

(* $0 \leq h < 24$

$0 \leq m < 60$

*)

(* Purpose: write a function to
convert a 12-hour time to
a 24-hour time

E.g. $\text{to24}(3, 20, \text{"pm"}) = (\underline{15}, \underline{20})$

$$\text{to24}(3, 20, \text{"am"}) = (3, 20)$$

$$\text{to24}(12, 0, \text{"pm"}) = (12, 0)$$

$$\text{to24}(12, 0, \text{"am"}) = (0, 0)$$

*)

fun to24 ((h, m, p) : time12): time24 =

(case p of

"am" \Rightarrow

(case h of

12 \Rightarrow (0, m)

| any \Rightarrow (h, m))

| "pm" \Rightarrow

parens
around

(case h of

12 \Rightarrow (12, m)

| any \Rightarrow (h+12, m))

let val x = 5

in

blah

end

Booleans

a boolean is either

- true, or

- false

→ and that's it!

Operation: case analysis

Case b : bool of

true $\Rightarrow e_1$

| false $\Rightarrow e_2$

Case true of

true $\Rightarrow 5$ $\mapsto 5$

| false $\Rightarrow 6$

Case false of

true $\Rightarrow 5$

| false $\Rightarrow 6$

$\mapsto 6$

$x = y$ $x, y : \text{int}$

$x < y$ $x, y : \text{int}$

$x > y$ $x, y : \text{int}$

$b_1 \text{ or else } b_2 :=$
case b_1 of
 $\text{true} \Rightarrow \text{true}$
 $\text{false} \Rightarrow b_2$

$(b_1 : \text{bool} \ \&\& \ b_2 : \text{bool}) : \text{bool} :=$
andalso

case b_1 of
 $\text{false} \Rightarrow \text{false}$
 $\text{true} \Rightarrow b_2$

(* Purpose: check if one 12-hour time ^{opening helper} is earlier than another *)

fun earlier (t₁: time12, t₂: time12): bool =

let val (h₁, m₁) = to24(t₁)

val (h₂, m₂) = to24(t₂)

in

(h₁ <sup>= h₂) or else

(h₁ = h₂

and also

m₁ <sup>= m₂)

end

"using
to24
as
a
helper
function"

(* Example: 12:00 am is earlier than 12:00 pm

7:00 am is earlier than 6:00 pm *)

7:15 am is earlier than 7:16 am

earlier $((7, 0, \text{"am"}), (6, 0, \text{"pm"}))$

\mapsto let val $(h_1, m_1) = \text{to24}(7, 0, \text{"am"})$
val $(h_2, m_2) = \text{to24}(6, 0, \text{"pm"})$

in

end

\mapsto let val $(h_1, m_1) = (7, 0)$
val $(h_2, m_2) = (18, 0)$

in
end $h_1 < h_2$ or else -----

$\mapsto 7 < 18$ or else ----- \mapsto true or else ...
 \mapsto true

Val x = 5
↑ ↑

Case b of

true \Rightarrow
↑

~17

5-17

Laughs

$\text{laughs}(n) = \underbrace{\text{"hahaha...ha"}}_{n \text{ letters}}$

$\text{laughs}(n) = \text{"ahaaha"}$ if n is odd

fun laughs(n : int) : string =

case n of

0 \Rightarrow ""

1 \Rightarrow "a"

! any $\Rightarrow \text{laughs}(n-2) \wedge \text{"ha"}$

laughs(5)

↳ laughs(3) ~ "ha"

↳ (laughs(1) ~ ha) ~ "ha"

↳ ("a" ~ "ha") ~ "ha"

↳ "ahaha"

fun altLaughs(n: int): String =

Case n of

0 \Rightarrow ""

| any \Rightarrow

(case evenP(n) of

true \Rightarrow "h" ~ altLaughs(n-1)

| false \Rightarrow "a" ~ altLaughs(n-1)

altLaughs(4) = "haha"

5 = "ahaha"

atlaughs(5)

↳ case evenP(5) of false \Rightarrow "a" \wedge atlaughs(5-1)

↳ "a" \wedge atlaughs(4)

↳ "a" \wedge case evenP(4) of true \Rightarrow "h" \wedge atlaughs(3)

↳ "a" \wedge "h" \wedge atlaughs(3)

↳ "ah" \wedge atlaughs(3)

↳ . . .

Theorem

For all nats n ,

$$\underline{\text{laughs}(n)} = \underline{\text{altlaughs}(n)}$$

Helper
Theorem :

If n is odd,

laughs

atlaughs

then $\underbrace{\text{laughs}(n)}^{\text{laughs}} \wedge \text{"ha"} = \text{"ah"} \wedge \text{laughs}(n)$

"ahaha ... ha" \wedge "ha"
n letters

\downarrow
"ah" \wedge "ahaha ... ha"
 \uparrow

Proof case for 1:

To show $\text{lengths}(1)^a h a = a h^a \text{lengths}(1)$

But $\text{lengths}(1) = "a"$

Both = "aha" ✓

Case for $k+2$, where k is odd

Inductive hypothesis: $\text{lengths}(k)^a h a = a h^a \text{lengths}(k)$

To show $\text{lengths}(k+2)^a h a = a h^a \text{lengths}(k+2)$

$$\boxed{\text{laughs}(k+2) \wedge ha}$$

$$\mapsto (\text{laughs}(k) \wedge ha) \wedge ha$$

$$= \underline{ah} \wedge \underline{\text{laughs}(k)} \wedge \underline{ha}$$

$$= ah \wedge (\text{laughs}(k) \wedge ha)$$

by inductive hypothesis associativity

$$\Leftarrow \boxed{ah \wedge (\text{laughs}(k+2))}$$

$$\boxed{\text{Using IH}} \quad \text{laughs}(k) \wedge ha = ah \wedge \text{laughs}(k)$$