Meeting 13: Data Types and Polymorphism

Announcements

HW4 due Friday 10/13

Start early! Write turning hours!

Questions

1. printing and explore
2. run free
3. what needs to be explained?
4. Go over the type checking
5. Get help vs. in run tree
6. input(s) - \& PL - just for ret
7. flattening if-expressions?
\[ x = \text{if } \text{not}(\text{not}(\text{not}(\text{not} x))) \text{ else } \text{not} x \]

5) Get Tag 73.

Rule HW4 for
what is ok to do at the runtime
and what to put in the compiled code:

- \text{not} \text{ and } \text{not} \text{ should not} (aka, don't go into runtime)
- Big things like lists
- can go into runtime

Is \text{not}(\cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot) in the runtime?

\text{projected type: (tagged)} \text{unit} \text{to: (tagged) \text{unit}}

\text{can't call from your complex}

Why?

\text{meta of the actual object}
\text{Db} \text{list is values in Pl}

unnecessarily (tagged)
calls are expensive
Rule for simplifying $P_1$ is $P_1 \equiv \text{conclude}$ (Exercise $\cdots \cdots$)

If $\text{Expr}(e_1, e_2, e_3)$

atom $\equiv$ variable
constant

$\text{flatten} \text{ expr} : \text{expression} \rightarrow (\text{statements, atoms})$

$e_2 \text{ if } e_1 \text{ else } e_3$

$P_1 \text{ input} (f_3, f_2) = \text{flatten expr}(f_3, f_2)$

$\text{flatten}$

$f_3$

if $f_3$

if $f_2$
	$\text{tmp} = f_2$
else:
	$\text{tmp} = f_2$

else:
	$\text{tmp} = f_3$

$\text{flatten } P_1 \text{ output}$

$\text{tmp}$

$\text{tmp}$

$\text{tmp}$
\text{If} \ g \neq 0 \ \text{then} \ \{ \ a, \ b, \ c, \ d \}