Term variables, x, y, z	variables
Type variables, a, b	type variables
Integer, i	

Monotypes, τ	$::=$			Monotypes
	$\|$	$\mathtt{Int}$		
	$\|$	a		
	$\|$	$\tau_1 \rightarrow \tau_2$		
Rho-types, ρ	$::=$			Rho-types
	$\|$	τ		
	$\|$	(ρ)	S	
Polytypes, σ	$::=$			
	$\|$	ρ		
	$\|$	$\forall a.\sigma$	bind a in σ	
	$\|$	$[a \mapsto \tau]\,\sigma$	M	
Terms, t, u	$::=$			
	$\|$	i		Literal
	$\|$	x		Variable
	$\|$	$\lambda x.t$	bind x in t	Abstraction
	$\|$	$t\ u$		Application
	$\|$	$\lambda(x :: \sigma).t$	bind x in t	Typed abstraction
	$\|$	$\mathtt{let}\ x = u\ \mathtt{in}\ t$	bind x in t	Local binding
	$\|$	$t :: \sigma$		Type annotation
	$\|$	(t)	S	
	$\|$	$t\,[x \rightsquigarrow t']$	M	
value, v	$::=$			
	$\|$	i		
	$\|$	$\lambda x.t$	bind x in t	
	$\|$	$\lambda(x :: \sigma).t$	bind x in t	
Typing contexts, Γ	$::=$			
	$\|$	ϵ		empty context
	$\|$	$\Gamma, x : \sigma$		assumption
fv	$::=$			
	$\|$	$\mathit{fv}(t)$	M	
ftvany	$::=$			
	$\|$	$\mathrm{ftv}(\tau)$	M	
	$\|$	$\mathrm{ftv}(\rho)$	M	
	$\|$	$\mathrm{ftv}(\sigma)$	M	
	$\|$	$\mathrm{ftv}(\Gamma)$	M	
terminals	$::=$			
	$\|$	$\longrightarrow$		
	$\|$	$\rightarrow$		
	$\|$	$::$		
	$\|$	fv		
	$\|$	ftv		

$$
\begin{array}{lcl}
 & | & \in \\
 & | & \notin \\
 & | & \vdash \\
 & & \\
\mathit{formula} & ::= & \\
 & | & \mathit{judgement} \\
 & | & \mathsf{value}\, v \\
 & | & \mathsf{uniq}\, \Gamma \\
 & | & \mathsf{closed}\, \sigma \\
 & | & a \notin \mathit{ftvany} \\
 & | & x : \sigma \ \in \ \Gamma \\
 & & \\
\mathit{JTyping} & ::= & \\
 & | & \Gamma \vdash t : \sigma \\
 & & \\
\mathit{JStep} & ::= & \\
 & | & t \longrightarrow u \\
 & & \\
\mathit{judgement} & ::= & \\
 & | & \mathit{JTyping} \\
 & | & \mathit{JStep} \\
 & & \\
\mathit{user\_syntax} & ::= & \\
 & | & \text{Term variables} \\
 & | & \text{Type variables} \\
 & | & \text{Integer} \\
 & | & \text{Monotypes} \\
 & | & \text{Rho-types} \\
 & | & \text{Polytypes} \\
 & | & \text{Terms} \\
 & | & \mathit{value} \\
 & | & \text{Typing contexts} \\
 & | & \mathit{fv} \\
 & | & \mathit{ftvany} \\
 & | & \mathit{terminals} \\
 & | & \mathit{formula}
\end{array}
$$

$\boxed{\Gamma \vdash t : \sigma}$

$$\frac{}{\Gamma \vdash i : \mathtt{Int}} \quad \text{TYP\_INT}$$

$$\frac{\begin{array}{c}\mathsf{uniq}\, \Gamma \\ x : \sigma \ \in \ \Gamma\end{array}}{\Gamma \vdash i : \sigma} \quad \text{TYP\_VAR}$$

$$\frac{\Gamma, x : \tau_1 \vdash t : \tau_2}{\Gamma \vdash (\lambda x.t) : (\tau_1 \to \tau_2)} \quad \text{TYP\_ABS}$$

$$\frac{\begin{array}{c}\Gamma \vdash t : \tau_1 \to \tau_2 \\ \Gamma \vdash u : \tau_1\end{array}}{\Gamma \vdash t\ u : \tau_2} \quad \text{TYP\_APP}$$

$$\frac{\Gamma \vdash u : \sigma \quad \Gamma, x : \sigma \vdash t : \rho}{\Gamma \vdash \texttt{let}\ x = u\ \texttt{in}\ t : \rho} \quad \textsc{typ\_let}$$

$$\frac{\mathsf{closed}\,\sigma \quad \Gamma \vdash t : \sigma}{\Gamma \vdash (t :: \sigma) : \sigma} \quad \textsc{typ\_annot}$$

$$\frac{\Gamma \vdash t : \rho}{\Gamma \vdash t : \forall a.\rho} \quad \textsc{typ\_gen}$$

$$\frac{\Gamma \vdash t : \forall a.\rho}{\Gamma \vdash t : [a \mapsto \tau]\,\rho} \quad \textsc{typ\_inst}$$

$\boxed{t \longrightarrow u}$

$$\frac{u \longrightarrow u'}{\texttt{let}\ x = u\ \texttt{in}\ t \longrightarrow \texttt{let}\ x = u'\ \texttt{in}\ t} \quad \textsc{step\_let}1$$

$$\frac{}{\texttt{let}\ x = v\ \texttt{in}\ t \longrightarrow t\,[x \rightsquigarrow v]} \quad \textsc{step\_let}$$

$$\frac{t \longrightarrow t'}{t\ u \longrightarrow t'\ u} \quad \textsc{step\_app}1$$

$$\frac{u \longrightarrow u'}{(\lambda x.t)\ u \longrightarrow (\lambda x.t)\ u'} \quad \textsc{step\_app}2$$

$$\frac{}{(\lambda x.t)\ v \longrightarrow t\,[x \rightsquigarrow v]} \quad \textsc{step\_app}$$

$$\frac{u \longrightarrow u'}{(\lambda(x :: \sigma).t)\ u \longrightarrow (\lambda(x :: \sigma).t)\ u'} \quad \textsc{step\_annot\_app}2$$

$$\frac{}{(\lambda(x :: \sigma).t)\ v \longrightarrow t\,[x \rightsquigarrow v]} \quad \textsc{step\_annot\_app}$$

$$\frac{}{t :: \sigma \longrightarrow t} \quad \textsc{step\_erase}$$

```
Definition rules:         16 good    0 bad
Definition rule clauses: 31 good    0 bad
```