Chapter 1. Rudiments of Logic

bool := PN	sort	
p ...	bool	
⊢ := PN	sort	Notation: $\vdash(p)$ will be shortened to $\vdash p$
q ...	bool	
and := PN	bool	
ass1 ...	$\vdash p$	
ass2 ...	$\vdash q$	
conjax := PN	$\vdash$ and	
ass3 ...	$\vdash$ and	
first := PN	$\vdash p$	
second := PN	$\vdash q$	
ξ ...	sort	
p ...	$[x\ \xi]$ bool	
$\forall$:= PN	bool	Notation: $\forall(\xi, P)$ will be abbreviated $\forall P$ since ξ is
ass1 ...	$[x\ \xi] \vdash \{x\} P$	implied by P. Example: We write $\forall [x\ \xi] \mathcal{E}(x)$ instead of $\forall(\xi, [x\ \xi] \mathcal{E}(x))$
generalize := PN	$\vdash \forall$	
ass2 ...	$\vdash \forall$	
x ...	ξ	
specialize := PN	$\vdash \{x\} P$	
$\exists$:= PN	bool	$\exists(\xi, P)$ will be abbreviated $\exists P$.
x ...	ξ	
ass1 ...	$\vdash \{x\} P$	
exist := PN	$\vdash \exists$	
ass2 ...	$\vdash \exists$	
choose := PN	ξ	
thus := PN	$\vdash \{\text{choose}\} P$	

Chapter 2 Sets

elt := PN	sort	element
set := PN	sort (elt)	A set is a special sort of element
x ...	elt	
S ...	set	
in := PN	bool	"x is in S"
S ...	set	
ϵ := PN	sort (elt)	Notation: $\epsilon(S)$ is written ϵS
$\subseteq$:= PN	sort (set)	Notation : $\subseteq(S)$ is written $\subseteq S$
x ...	ϵS	"x is of sort ϵS"
defϵ := PN	$\vdash$ in(x,S)	
x ...	elt	
S ...	set	
ass1 ...	$\vdash$ in(x,S)	
θ := PN	ϵS	Representative of x whose sort is ϵS
P ...	[x elt] bool	
ass2 ...	$\vdash$ {x} P	θ satisfies the same predicates as x
sameθ := PN	$\vdash$ {θ} P	
ass 2 ...	$\vdash$ {θ} P	
same x := PN	$\vdash$ {x} P	
S ...	set	
T ...	set	
incl := $\forall$[xϵS] in(x,T)	bool	Definition of set inclusion
equal := and (incl (S,T), incl (T,S))	bool	Definition of set equality
refl$\subseteq$:= generalize(ϵS, [xϵS] in(x,S), [xϵS] defϵ(x))	$\vdash$ incl(S,S)	$S \subseteq S$
refl= := conjax (incl (S,S), incl(S,S), refl$\subseteq$, refl$\subseteq$)	$\vdash$ equal(S,S)	$S = S$
ass1 ...	$\vdash$ equal (S,T)	
s1 := first (incl(S,T), incl (T,S), ass1)	$\vdash$ incl (S,T)	
s2 := second (incl(S,T), incl(T,S), ass1)	$\vdash$ incl (T,S)	
symm= := conjax(incl(T,S), incl(S,T), s2, s1)	$\vdash$ equal (T,S)	$S=T \Rightarrow T=S$

```
| x ...                                                         elt
|  | S ...                                                      set
|  |  | ass1 ...                                              ⊢ in(x,S)
|  |  |  | T ...                                              set
|  |  |  |  | ass2 ...                                         ⊢ incl(S,T)
|  |  |  |  | s1 := θ(x, S, ass1)                              ∈S
|  |  |  |  | s2 := specialize(∈S, [x∈S] in(x,T), ass2, s1)    ⊢ in(s1, T)
|  |  |  |  | lemma1 := samex(x, S, ass1, [x elt] in(x,T), s2) ⊢ in(x, T)      x∈S⊆T ⇒ x∈T

| S ...                                                         set
|  | T ...
|  |  | ass1 ...                                              ⊢ incl(S,T)
|  |  |  | U ...                                              set
|  |  |  |  | ass2 ...                                         ⊢ incl(T,U)
|  |  |  |  |  | x ...                                         ∈S
|  |  |  |  |  | s1 := specialize(∈S, [x∈S]in(x,T), ass1, x)  ⊢ in(x, T)      (By convention I reuse s1, s2, ... instead of making up
|  |  |  |  |  | s2 := lemma1(x, T, s1, U, ass2)              ⊢ in(x, U)        unique names)
|  |  |  |  | trans⊆ := generalize(∈S, [x∈S] in(x,U), [x∈S] s2(x))  ⊢ incl(S,U)   S⊆T⊆U ⇒ S⊆U
|  | ass1 ...                                                  ⊢ equal(S,T)
|  |  | U ...                                                  set
|  |  |  | ass2 ...                                             ⊢ equal(T,U)
|  |  |  | s1 := trans⊆(first(incl(S,T), incl(T,S), ass1), U, first(incl(T,U), incl(U,T), ass2))   ⊢ incl(S,U)
|  |  |  | s2 := trans⊆(second(incl(S,T), incl(T,S), ass1), U, second(incl(T,U), incl(U,T), ass2)) ⊢ incl(U,S)
|  |  |  | trans= := conjax(incl(S,U), incl(U,S), s1, s2)       ⊢ equal(S,U)     S=T=U ⇒ S=U

|  | T ...                                                      ⊆S
|  |  ∈ := PN                                                   sort(∈S)       ∈T is special sort of ∈S.
|  |  def⊆ := generalize(∈T, [x∈T] in(x,S), [x∈T] def∈(x))     ⊢ incl(T,S)    Note use of double definition of ∈
```

P ...	[x ∈ S] bool	construction of
brace := PN	⊆ S	$\{x \in S \mid P(x)\}$
x ...	∈ S	
ass1 ...	⊢ in (x, brace)	
unbrace := PN	⊢ {x} P	
ass2	⊢ {x} P	
embrace := PN	⊢ in (x, brace)	
x ...	∈ brace	
def b := unbrace(x, def∈(brace, x))	⊢ {x} P	
power := PN	sort (set)	The power set.
℘ := PN	power (S)	
U ...	power (S)	
∈ := ⊆ S	sort (set)	

One can prove now that $A \in \wp S$ implies $incl(A, S)$ and $A \subseteq S$ implies $in(A, \wp S)$, without further axioms; but these facts aren't needed directly in chapter 3

Chapter 3 Equivalence relations

S ...	set	A fixed equivalence relation, "equiv", is assumed. It is not difficult to change the assumption to a variable equivalence relation by adding extra hypotheses surrounding chapter 3 and replacing some PN's by proofs.
x ...	∈ S	
y ...	∈ S	
equiv := PN	bool	
refl := PN	⊢ equiv(x, x)	
ass 1 ...	⊢ equiv(x, y)	
sym := PN	⊢ equiv(y, x)	
z ...	∈ S	
ass2 ...	⊢ equiv(y, z)	
trans := PN	⊢ equiv(x, z)	
eclass := brace(S, [y∈S] equiv(x, y))	⊆ S	Equivalence class containing x
z ...	∈ eclass(x)	
s1 := defb(S, [y∈S] equiv(x, y), z)	⊢ equiv(x, z)	
s2 := trans(y, x, sym, s1)	⊢ equiv(y, z)	
s3 := embrace(S, [z∈S] equiv(y, z), z, s2)	⊢ in(z, eclass(y))	
hence := generalize(∈ eclass(x), [z∈eclass(x)] in(z, eclass(y)), [z∈eclass(x)] s3(z))	⊢ incl(eclass(x), eclass(y))	If x ≡ y then eclass(x) ⊆ eclass(y)
similarly := hence(y, x, sym)	⊢ incl(eclass(y), eclass(x))	similarly eclass(y) ⊆ eclass(x)
Proposition 1 := conjax(incl(eclass(x), eclass(y)), incl(eclass(y), eclass(x)), hence, similarly)	⊢ equal(eclass(x), eclass(y))	<u>First result</u>: x ≡ y ⟹ eclass(x) = eclass(y)
ass 3 ...	⊢ in(y, eclass(x))	
defeclass := unbrace(S, [z∈S] equiv(x, z), y, ass3)	⊢ equiv(x, y)	y ∈ eclass(x) ⟹ x ≡ y.
A ...	∈ ℘S	
P := ∃([x∈S] equal(A, eclass(x)))	~~EA∈℘S~~ bool	
Π := brace(℘S, [A∈℘S] P(A))	⊆ ℘S	The set of equivalence classes

```
x ...                                                          ∈ S
  A ...                                                        ∈ Π
    ass 1 ...                                                  ⊢ in(x,A)
      s1 := defb(PS, [A∈PS] P(A), A)                           ⊢ P(A)
      z := choose(∈S, [x∈S] equal(A, eclass(x)), s1)           ∈ S
      p1 := thus(∈S, [x∈S] equal(A, eclass(x)), s1)            ⊢ equal(A, eclass(z))
      s2 := lemma 1(x, A, ass1, eclass(z), p1)                 ⊢ in(x, eclass(z))
      s3 := defeclass(z, x, s2)                                ⊢ equiv(z,x)
      p2 := trans=(A, eclass(z), p1, eclass(x), Proposition 1(z,x,s3))      ⊢ equal(A, eclass(x))
      B ...                                                    ∈ Π
        ass 2                                                  ⊢ in(x,B)
          p3 := p2(B, ass2)
          Proposition 2 := trans=(A, eclass(x), p2,
                    B, symm=(B, eclass(x), p3))                ⊢ equal(A,B)
                                                               ] Second result: A,B∈Π, x∈A, x∈B ⇒ A=B.
q1 := embrace(S, [y∈S] equiv(x,y), x, refl(x))                 ⊢ in(x, eclass(x))
q2 := embrace(PS, [A∈PS] P(A), eclass(x),
            exist(∈S, P(eclass(x)), x, refl=(eclass(x))))      ⊢ in(eclass(x), Π)
q3 := ~~θ~~ θ(eclass(x), Π, q2)                                ∈ Π
q4 := same θ(eclass(x), Π, q2, [A elt] in(x,A), q1)            ⊢ in(x, q3)
Proposition 3 := exist(∈Π, [A∈Π] in(x,A), q3, q4)              ⊢ ∃[A∈Π] in(x,A)
                                                               ] Third result: x∈S ⇒ ∃A∈Π, x∈A.
  y ...                                                        ∈ S
    ass1                                                       ⊢ equiv(x,y)
      s1 := lemma 1(x, eclass(x), q1(x), eclass(y), Proposition 1(x,y,ass1))   ⊢ in(x, eclass(y))
      s2 := same θ(eclass(y), Π, q2(y), [A elt] in(x,A), s1)   ⊢ in(x, q3(y))
    Proposition 4 := exist(∈Π, [A∈Π] and(in(x,A), in(y,A)), q3(y), conjax(in(x,A), in(y,A), s2, q4(y)))
                                                               ⊢ ∃[A∈Π] and(in(x,A), in(y,A))
                                                               ] Fourth result: x≡y ⇒ ∃ A∈Π; x∈A, y∈A.
    ~~ass 1~~
    A                                                          ∈ Π
      ass1 ...                                                 ⊢ in(x,A)
        ass2 ...                                               ⊢ in(y,A)
          Proposition 5 := defeclass(x, y, lemma 1(y, ~~eclass~~ A, ass2, eclass(x), p2(A, ass1))
                                                               ⊢ equiv(x,y)
                                                               ] Fifth result: x,y∈A ⇒ x≡y.
```