AUT II M1

Definition of LIMITS in AUTOMATH

by N.G. de Bruijn. *

1. Introduction to mathematics.

1.1	0	bool	:=	PN	sort
1.2	0	x	:=	- - - - -	bool
1.3	x	TRUE	:=	PN	sort
1.4	0	ksi	:=	- - - - -	sort
1.5	ksi	nonempty	:=	PN	bool
1.6	ksi	a	:=	- - - - -	ksi
1.7	a	then 2	:=	PN	TRUE(nonempty)
1.8	ksi	P	:=	- - - - -	[u ksi] bool
1.9	P	exists	:=	PN	bool
1.10	P	v	:=	- - - - -	ksi
1.11	v	ass 1	:=	- - - - -	TRUE({v} P)
1.12	ass 1	then 13*	:=	PN	TRUE(exists)
1.13	P	ALL	:=	[u ksi] TRUE({u} P)	sort
1.14	P	all	:=	nonempty(ALL)	bool
1.15	0	a	:=	- - - - -	bool
1.16	a	b	:=	- - - - -	bool
1.17	b	IMPL	:=	[u TRUE(a)] TRUE(b)	sort
1.18	b	impl	:=	nonempty(IMPL)	bool

* Internal Report, 1969
Department of Mathematics,
Technological University
Eindhoven, The Netherlands.

2. Definition of limit.

2.1	0	real	:=	PN	sort
2.2	0	|r1	:=	- - - - -	real
2.3	r1	||r2	:=	- - - - -	real
2.4	r2	||distance	:=	PN	real
2.5	r2	||less	:=	PN	bool
2.6	0	null	:=	PN	real
2.7	0	nat	:=	PN	sort
2.8	0	sequence	:=	[n nat]real	sort
2.9	0	1	:=	PN	nat
2.10	0	|k	:=	- - - - -	nat
2.11	k	||l	:=	- - - - -	nat
2.12	l	||lessnat	:=	PN	bool
2.13	0	|a	:=	- - - - -	sequence
2.14	a	||l	:=	- - - - -	real
2.15	l	|||eps	:=	- - - - -	real
2.16	eps	|||m_o	:=	- - - - -	nat
2.17	m_o	||||m	:=	- - - - -	nat
2.18	m	||||w	:=	impl(lessnat(m_o,m), less(distance({m}a,l),eps))	bool
2.19	m_o	||||z	:=	all(nat,[s nat] w(s))	bool
2.20	eps	|||y	:=	exists(nat,[t nat] z(t))	bool
2.21	eps	|||q	:=	impl(less(null,eps),y)	bool
2.22	l	||lim	:=	all(real,[eps,real] q(eps))	bool
2.23	a	|conv	:=	exists(real,[t real] lim(t))	bool

3. A lemma without proof.

3.1	0	x	:=	- - - - -	real
3.2	x	y	:=	- - - - -	real
3.3	y	if	:=	- - - - -	TRUE(less(null,y))
3.4	if	lemma	:=	PN	TRUE(less(distance(x,x),y))

4. Constant sequences are convergent.

4.1	0	c	:=	- - - - -	real
4.2	c	p	:=	[m nat] c	sequence
4.3	c	delta	:=	- - - - -	real
4.4	delta	ass	:=	- - - - -	TRUE(less(null,delta)
4.5	ass	n_o	:=	- - - - -	nat
4.6	n_o	n	:=	- - - - -	nat
4.7	n	then	:=	lemma({n}p,delta,ass)	TRUE(less(distance ({n}p,c),delta))
4.8	n	a	:=	lessnat(n_o,n)	bool
4.9	n	aa	:=	w(p,c,delta,n_o,n)	bool
4.10	n	b	:=	then 2([u TRUE(a)] sort(then), [u TRUE(a)] then)	TRUE(aa)
4.11	n_o	h	:=	then 2([s nat] TRUE(aa(s)), [s nat]b(s))	TRUE(z(p,c,delta,n_o))
4.12	ass	d	:=	then 13*(nat,[x nat]z(p,c,delta,1),1,h(1))	TRUE(y(p,c,delta))
4.13	delta	e	:=	then 2([c sort(ass)] sort((c)d), [c sort(ass)]d(c))	TRUE(q(p,c,delta))
4.14	c	f	:=	then 2([s real] TRUE(q,(p,c,s)), [s real] e(s))	TRUE(lim(p,c))
4.15	c	g	:=	then 13*(real,[s real] lim(p,s),c,f)	TRUE(conv(p))

Remark. If with a given indicator a line a := Σ Λ is acceptable, then we take the liberty to write sort(Σ) instead of Λ . Example: (see b above) : sort(then) stands for TRUE(less(distance({n}p,c),delta).