

KDF-9 Atlas Autocode Compiler

**Source Listing Ver 4F
Incomplete 16-2-65**

begin

integer i,j,k

integer array cword(1:190),clett(1:800),symbol(300:1250),cc(0:300)

accept m/c instructions

VER 4F

INCOMPLETE

begin

integer cnext,dnext,cnum,dnum,alt,def,asl

integer array kk,dword(201:250),dlett(1:320)

routine spec read string

routine spec record(integer array name word,lett,integer name num,next)

routine spec look up(integer array name word,lett,integer first,last)

asl=300

cnext=1

dnext=1

cnum=0

dnum=200

1:read symbol(1)

if i=35 then ->2

if i=36 then ->3

if i=48 then ->4

if i=37 then ->5

->1

2:read symbol(1)

read string

record(cword,clett,cnum,cnext)

->1

3:read symbol(1)

read string

record(dword,dlett,dnum,dnext)

->1

4:read symbol(i)

read string

look up(dword,dlett,208,dnum)

kk(i)=asl

def=asl

alt=asl+1

asl=asl+2

6:read symbol(i)

if i=81 then ->7

if i=10398 then ->8

if i=10 then ->9

if i=10122 then ->10

if i=14223 then ->11

->6

7:read string

look up(dword,dlett,201,dnum)

12:symbol(asl)=i

asl=asl+1

->6

8:read string

look up(cword,clott,1,cnum)

->12

9:symbol(alt)=as1

alt=as1

as1=as1+1

->6

10:symbol(alt)=as1

symbol(def)=as1

->1

11:i=200

->12

5:cycle i=300,1,as1-1

if 208<symbol(i)<299 then symbol(i)=kk(symbol(i))

repeat

k=kk(dnum)

newline

print(as1,5,0)

print(cnum,5,0)

print(cnext,5,0)

```
print(dnum,5,0)
```

```
print(dnext,5,0)
```

```
newline
```

routine read string

j=0

1:read symbol(1)

if i=86 or i=65 then ->1

if i=10526 or i=82 then ->2

j=j+1

cc(j)=i

->1

2:cc(0)=j

end

routine record(integer array name word,lett,integer name num,next)

num=num+1

word(num)=next

cycle i=0,1,cc(0)

lett(next+i)=cc(i)

repeat

next=next+cc(0)+1

end

routine look up(integer array name word,lett,integer first,last)

cycle i=first,1,last

j=word(i)

cycle k=0,1,lett(j)

if lett(j+k)≠cc(k) then →1

repeat

return

1:repeat

caption ≠ phrase ≠ not ≠ in ≠ dictionary ≠ ≠ ≠

cycle i=1,1,cc(0)

print symbol(cc(i))

repeat

stop

end

end

```

begin
integer line,length,length',hit,p,q,r,lvl,no,num,next,s,t,u,v,faulty,item,n,nr
integer R,qu,plabel,level,mc switch, perm,aflag,dflag,overflag,VO,ppstart,pp curr
integer pt block,pch sum, ptdev,pl start,pl curr, ps start, ps curr,compiler
integer time limit
integer array ST(-3:2500),L,A(1:400),word,tags,SL(0:255),lett(1:650)
integer array cycle,name,label,jump,flag,RAL,SET(0:15)
integer array RA(-3:127),pp buf, pl buf, ps buf(-3:99),T(1:6)
real a,b
routine spec compare
routine spec cSS
routine spec splash(integer warning,integer name from)
routine spec initialise
integer in spec ca
T(1)=5
T(2)=6
T(3)=8
T(4)=9
T(5)=15
T(6)=8

time limit = 120

```

overflag = 1

dflag = 0

faulty=0

n0=4

n = 2

nr = 0

mc switch = 0

perm = 1

compiler = 1

qu = 0

plabel = 50000

level = 0

cycle j = 2500,-1,1001

ST(j) = j-1

repeat

ST(1000) = 0

j = -1

cycle i = 0,1,15

cycle (i) = j

name (i) = j

label (i) = j

jump (i) = j

repeat

cycle i = 0,1,255

SL(i) = 0

tags (i) = j

repeat

comment initialisation

320: * SET 2500

 * DUP

 * = I9

 * = C9

 ** αST(0)

 * = M9

pp curr = 9830400 ; comment 0/150/0

initialise

-> 342

routine initialise

 * SET 419

 * = M13

 * MOM13

 * SET 4

 * OUT

 * DUP

 ** = pt dev

 * = C15

 * SET 5

 * = M15

 * PMDQ15 ; comment rewind

 * PMAQ15 ; comment skip label, call, and possibly compiler

300 * 32768

replace

done

*

Vo = 65535

pt block = 0

p ch sum = 0

pl curr = 0

ps curr = 65536

** α pp buf (0)

** = pp start

** α pl buf (0)

** = pl start

** α ps buf (0)

** = ps start

cycle i = -3,1,99

pp buf (i) = 0

pl buf (i) = 0

ps buf (i) = 0

repeat

end

50P:

** pp start

* = M14

** pp curr

* = Q15

304:

* C15

* NOT

* NEG

* DUP

* SET -8

* xD

* CONT

* = C15

* REV

* SET 255

* AND

* SHC C15

* M14 M15

* OR

* = M14 M15

* SET 6

* J301*

* ERASE

* CO TO Q15

54P:

* SET 1

* = + M15

* SET 100

* M15

* J 302*

* MO TO Q 15

* = + I 15

* Q14

* Q15

splash (0,pp buf(0))

* = Q15

* = Q14

-> 303

301:

* = C15

-> 303

302:

* ERASE

303:

* DC 14

* J304 C14 NZ

* Q15

** = pp curr

* EXIT 1

52P: ** ps start
 ** ps curr
 -> 316
51P: ** pl start
 ** pl curr
316: * = Q15
 * = M14
 * SET 2
 * = C14
314: * C15
 * SET 1
 * DUP
 * = + C15
 * -
 * DUP
 * J310 = Z
 * J311 > Z
 * SHL 32
 * = M14M15
 -> 312
310: * ERASE
 ** VO
 * AND
 * SHL 16
 * M14M15
 * OR
 * = M14M15
 -> 312

311:

** VO
* AND
* M14M15
* OR
* = M14M15
* CO TO Q15
* SET 1
* = + M15
* SET 100
* M15
* -
* J312 ≠ Z
* MO TO Q15
* Q14
* Q15
* I15
* J313 ≠ Z
splash (1, pl buf (0))
-> 317

313:

splash (2, ps buf (0))

317:

* = Q15
* = Q14

312:

* DC14
* J314 C14 NZ
* Q15
* I15
* J315 ≠ Z

** = pl curr

* EXIT 1

315:

** = ps curr
* EXIT 1

routine splash (integer warning, integer name from)

return if faulty ≠ 0

```

    ** α from
    * DUP
    * = RM14
    * SET 3
    * -
    * DUP
    * = I 15
    * SET 100
    * = C14
    * ZERO
    * VR
2:   * MOM14 Q
    * +
    * J1 NV
    * NOT
    * NEG
1:   * J2 C14 NZ
    * M-I14
    * M14 TO Q15
    * REV
    * = M14
    ** pt block
    * DUP
    * = MOM14 Q
    * NOT
```

* NEG
** = pt block
** warning
* = MOM14 Q
* DUP

* = MOM14 Q
** p ch sum
* +

* J3 NV
* NOT
* NEG

3:

** = p ch sum
** pt dev
* = C15

* POAQ 15; comment MWQ 15
* SET 100
* = C14

5:

* ZERO
* = MOM14 Q
* J5 C14 NZ

end

integer in ca

integer i

** pp curr

* DUP

* = Q14

* M+I 14

* SHL-19

* SET B 160000

* AND

* M14

* +

** = i

result = i

end

routine dump stack and routines

integer 1

-> 8 if faulty = 0

caption # PROGRAM # FAULTY

stop

```
8:      ST(0) = n0
        ST(1) = nr
        ** pp curr
        *   = Q15
        *   J1C15 NZ
        *   M15
        *   J2 = Z

1:      splash (0, pp buf (0))
2:      ** pl curr
        *   J3 = Z
        splash (1, pl buf (0))
3:      ** ps curr
        *   = Q15
        *   J4 C15 NZ
        *   M15
        *   J5 = Z

4:      splash (2, ps buf (0))
5:      i=0
6:      splash (3, ST(i))
        i = i + 100
        -> 6 if i < n0
        i = 0

7:      splash (4, RA(i))
        i = i + 100
        -> 7 if i < nr
```


ST(0) = p ch sum

*p///lash (7,ST(0))

caption # PROGRAM # DUMPED # COMPILING # TIME # #

* SET 9

* OUT

* SHL-24

* DUP

* SET 91P:

* =M13

* MOM13

* REV

* =MOM13

* -

* JS 2P

caption # / # #

* SET 3

* OUT

* SHL-24

* JS 2P

** time limit

** pt dev

* SET 58P: ; comment entry sequence

* DUP

* = RM5

* = RM6

* SET 10

* = I1

* SET 13

```
* = RM8
* SET 11
* = RM7
* SET 5
* = M1
* GO TO Q11
* = C1 ; comment device
* PMDQ1 ; comment rewind
* PMAQ1 ; comment skip label, call, compiler
* SET 112
* = M1
* ZERO ; comment for checksum
* SET 35P: ; comment reader
* = M13
* MOM13
* = Q13
* C13
* SET 6
* OUT
* SET 38P: ; comment punch
* =M13
* MOM13
* =Q13
* C13
* SET 6
* OUT
* SET 65P:
* = M13
```

- * SET 64P:
- * = M14
- * J61P

end

342:

comment set up π and $\frac{1}{2}$

* SET +2
* = M13
* SET B 401
* SHL +38
* = M13M9

* SET +3
* = M13
* SET B 40544
* = C14
* SET B 103755
* = I14
* SET B 450421
* = M14
* Q14
* = M13M9

num=0

next=1

line=-1

8:length=1

length'=1

1:length=length-1

2:read symbol(i)

if i=65 or i=86 then ->2

if i ≠ 88 then -> 10

length = length +2

cc(length-1) = 11150

cc (length) = 18

-> 2

10: if i ≠ 92 then -> 3

length = length +2

cc(length-1) = 31

cc(length) = 21

-> 2

3: length=length+1

cc(length)=1

if i≠4 then ->2

if length=length' then ->1

~~if~~ if cc(length-1)≠12758 then ->4

length=length-2

length'=length+1

->2

4: line = line + 1

q = 1

5: lvl=0

p=k

r=1

compare

if r > 380 then -> 11

if hit=0 then->6

comment call for cSS here

c88

if A(1)=16 then ->7

if q=length then ->8

->5

11: print symbol (73)

newlines(2)

print (line,3,0)

caption : * LONG * ANALYSIS * RECORD *

print symbol (73)

stop

6:newline

newline

p\\rint symbol (73)

print(line,3,0)

caption : * INSTRUCTION * NOT * RECOGNISED *

faulty = 1

cycle i=q,1,length-1

if cc(i)=10122 and cc(i+1)*4 then ->9

repeat

i=length

9:print symbol (73)

cycle j=q,1,i

print symbol(cc(j))

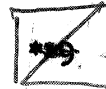
repeat

if i=length then ->8

q=i+1

->5

7:dump stack and routines



NOT ON TAPE 78.

```
routine compare
integer ra,rp,rq,rr,rs
switch bip(200:207),z(1:3),y(0:4)
rp=symbol(p)
rq=q
rr=r
lv1=lv1+1
L(r)=lv1
A(r)=1
p=p+1
ra=symbol(p)
rs=p
bip(200):1:r=r+1
2:rs=rs+1
if rs=ra then ->8
item=symbol(rs)
if item>300 then ->4
if item>200 then ->6
i=cword(item)
j=clett(i)+i
5:i=i+1
if clett(i)≠cc(q) then ->3
q=q+1
if i<j then ->5
L(r)=lv1+1
A(r)=item
->1
```

4: p=item

compare

if hit=0 then ->2

3: q=rq

r=rr

if ra=rp then ->7

rs=ra

A(r)=A(r)+1

ra=symbol(ra)

->1

8: hit=1

->9

7: hit=0

9: lvl=lvl-1

return

6: i=cc(q)

L(r)=lvl+1

->bip(item)

comment phrase NAME

bip(201):if i<33 or 58<i<97 or i>122 then ->3

j=q

u=0

v=0

s=1

t=1

q=q+1

i=cc(q)

z(1):if i<33 or 58<i<97 or i>122 then ->10

11:u=u+1

** v

* SHL +8

** i

* +

** = v

if u ≠ 6 then -> 12

lett(next+s)=v

s=s+1

u=0

v=0

12:q=q+1

i=cc(q)

->z(t)

10:t=2

z(2):if i<16 then ->13

if i>25 then ->14

->11

14:t=3

z(3):if i≠32 then ->13

->11

13: if v=0 then ->15

lett(next+s)=v

->16

15: s=s-1

16: **q

 **j

 * -

 * SHL +8

 **cc(j)

 * +

 ** = lett(next)

 u = -1

17: u=u+1

if u=num then ->18

j=word(u)

cycle i=0,1,s

if lett(next+i)≠lett(j+i) then ->17

repeat

A(r)=u

->1

18: word(num)=next

next=next+s+1

A(r)=num

num=num+1

->1

comment phrase CONST

bip(202): if i $\neq$ 11 then -> 60

A(r) = 2

r = r+1

L(r) = lv1 +1

A(r) = 3

q = q +1

-> 1

60: if i $\neq$ 32 then -> 20

q=q+3

if cc(q-1) $\neq$ 32 then -> 3

u= cc(q-2)

if u<128 then -> 61

if u = 14735 or u = 14807 then u = 65

if u = 1895183 or u = 1895382 then u = 86

if u = 14095 or u = 14167 then u = 4

if u = 1296266 or u = 1435530 then u = 10122

if u = 12687 or u = 12759 then u = 73

-> 61

20:

* SET 41P:

* = M14

* SET + 1000

* = C14

* ZERO

** i

* DUP

* SET + 16

* -

* J 25 < Z

* DUP

* SET + 25

* -

* J23 $\leq$ Z

* SET + 31

* J 25 $\neq$

* ERASE

* ZERO

* = C14

22:

q = q +1
** cc(q)
* DUP
* SET +16
* -
* J 42 < Z
* DUP
* SET + 25
* -
* J 26 > Z
* REV
* MOM14
* xF
* J 8P V
* REV
* DC14

23:

* SET + 16
* -
* SET +47
* FLOAT
* +F
* J22

26:

* SET +31
* J 27 =
* SET + 90
* J 28 =

42:

* ERASE
* C14
* DUP
* J 44 < Z

33:

- * ERASE
- * J 34
- * DUP
- * J 30 = Z
- * DUP
- * J 29 > Z

44:

- * NEG
- * JS 14P
- * J 34

29:

- * JS 15 P
- * J 34

21:

- * ERASE
- * ERASE

25:

- * ERASE
- * ERASE
- * J3

27:

- * ERASE
- * C14
- * DUP
- * J 33 $\leq$ Z
- * ERASE
- * ZERO
- * = C14
- * J22

28:

- * ERASE
- * ZERO
- * ZERO
- q = q +1
- ** cc(q)
- * DUP
- * SET + 16

32:

* -
* J 21 < Z
* DUP
* SET + 25
* -
* J 31 > Z
* REV
* SET + 10
* xD
* CONT
* +
* SET + 16
* -

35:

q = q +1
** cc(q)
* DUP
* SET + 16
* -
* J 24 < Z
* DUP
* SET + 25
* -
* J 32 ≤ Z

24:

* ERASE
* REV
* J 36 = Z
* NEG
* C14
* J 33 > Z
* C14

36:

31:

- * +
- * J 33
- * SET + 29
- * J 37 =
- * SET + 30
- * J 21 *

- * CAB
- * ERASE
- * SET + 1

37:

- * PERM

- * ERASE

- * J35

30:

- * ERASE

34:

- * DUP

- ** = a

if fractp (a) = 0 then -> 43

A(r) = 2

r = r +1

L(r) = lvl +1

- ** no

- * = RM13

- * M-I13

- * M9M13

- * J52=

- * M+I13

- * = M9M13

- * M+I13

- * M13

- ** = no

- * ZERO

52:

* ERASE

A(r) = n0 -1

-> 1

43:

* ERASE

u = int (a)

61:

if u >= 32768 then -> 45

A(r) = 1

r = r +1

L(r) = lvl +1

A(r) = u

-> 1

45:

A(r) = 3

r = r +1

L(r) = lvl +1

if ST(n0-1) = u then -> 46

ST(n0) = u

n0 = n0 +1

46:

A(r) = n0 -1

-> 1

comment phrase N

bip(203): if i<16 ori>25 then ->3

u=0

38:u=10*u+i-16

q=q+1

i=cc(q)

if 16<i<25 then ->38

A(r)=u

->1

comment phrase S

bip(204): if i=4 then ->1

if i≠10122 then ->3

q=q+1

->1

comment phrase TEXT

bip(205):39: if i=4 or i=10122 then ->1

q=q+1

i=cc(q)

->39

comment phrase CAPTION TEXT

bip(206): A(r) = no

j = 0

** no

* DUP

* = M15

* = RM14

* M+I14

* SET + 40

* = C14

* ZERO

40: if i = 4 or i = 10122 then -> 47

if i < 128 then -> 62

if i = 14735 or i = 14807 then i = 65

if i = 14095 or i = 14167 then i = 4

if i = 1895183 or i = 1895382 then i = 86

if i = 12687 or i = 12759 then i = 73

if i > 128 then -> 48

62: ** i

49: * SHLC14

* +

* C14

* SET +8

* -

* DUP

* J 50 < Z

* = C14

-> y(j)

50: * ERASE

* = M9M14

* M+I14

```

* SET +40
* = C14
* ZERO
-> y(j)
48:  if i = 1296266 or i = 1435530 then i = 10122
** i
* SET B 177
* AND
j = 1
-> 49
y(1):y(3): * SET +3
j = j +1
-> 49
y(2): ** i
* SET B 37600
* AND
* SHL-7
j = 0
if i > 16384 then j = 3
-> 49
y(4): ** i
* SHL-14
* SET B 177
* AND
j = 0
-> 49
y(0): q = q +1
i = cc(q)
-> 40

```

47:

- * DUP
- * J 51 = Z
- * = M9M14
- * M+I14

51:

- * ZERO
- * ERASE
- * M-I14
- * M14
- * M15
- * -
- * = M9M15
- * M+I14
- * M14
- ** = nO
- > 1

comment phrase OCTAL

bip(207):if $i < 16$ or $i > 23$ then $\rightarrow 3$

$u = 0$

$41: u = 8 * u + i - 16$

$q = q + 1$

$i = cc(q)$

if $16 \leq i \leq 23$ then $\rightarrow 41$

$A(r) = u$

$\rightarrow 1$

end

***T

routine cSS
routine spec cCOND
routine spec cUI
routine spec cSEXP (integer z)
routine spec cNAME (integer z)
routine spec fetch tag (integer kk)
routine spec find label
routine spec pushdown 2 (integer name cell, integer s1, s2)
routine spec store tag (integer kk)
routine spec copy tag (integer kk)
routine spec from list 2 (integer name cell, s1, s2)
routine spec popup 2 (integer name cell, s1, s2)
routine spec newcell (integer name stad)
routine spec insert after 2 (integer name stad, integer s1, s2)
routine spec replace tag (integer kk)
routine spec link (integer name stad)
routine spec return cell (integer name stad)
routine spec more space
routine spec store cycle
routine spec fetch cycle
routine spec store label
routine spec store jump
routine spec store name
routine spec fetch jump
routine spec fetch labels
routine spec fetch name
routine spec copy jump

routine spec testnst
routine spec fault (integer n)
routine spec cMOD (integer n,x)
routine spec cUCI
routine spec pJ(integer a,s,b)
routine spec pSET(integer n)
routine spec pQ(integer a,q,qq)
routine spec pSH (integer a,n)
routine spec pN (integer a)
routine spec pMS(integer a,m,q)
routine spec fill label (integer at, equals)
routine spec fill set (integer at, equals)
routine spec print name (integer i)

switch sw(1:30), rfpd (1:9), RT(6:10), END (0:10)

integer i,j,k,p,type,q,kk,kkk,qq,jj,jjj,type'

routine spec cRSPEC (integer m)

-> sw(A(1))

sw(1): comment [UI] [S]

p = 2

cUI

->1

```
sw(2):      comment [cycle]
            p = 3
            cNAME(3)
            unless type = 2 then fault (25)
                pMS(1,n,level)

            p = p + 1
            cSEXP(2)

                pN(34)

            p = p + 1
            cSEXP(2)

                pN(34)
                pMS(1,n+1,level)
                pN(34)

            p = p + 1
            cSEXP(2)

                pN(34)
                pN(54)
                pN(46)
                pMS(1,n+2,level)
                pJ(8,SL(5),13)
                pMS(0,n,level)
                pQ(120,13,2)

            k = n
            store cycle

                pQ(65,13,0)

            n = n + 3
            -> 1
```

sw(3): comment [repeat] [8]
 fetch cycle
 if j ≠ -1 then → 3
 fault (1)
 → 1

3:

pms(0,k,level)
pq(120,13,2)
pq(64,13,0)
pms(0,k+1,level)
pn(46)
pms(0,k+2,level)
pj(8,j,1)
pn(42)

→ 1

```
sw(4):      comment [N] [:]  
            k = A(2)  
            find label  
            if j = -1 then -> 2  
            fault (2)  
            -> 1  
2:          store label  
            -> 1
```

```
sw(5):      comment [iu] [COND] [then] [UI] [S]  
            q = 2  
            -> 4
```

```

sw(6):      comment [UI] [iu] [COND] [S]
            q = 2
90:         q = q+1
            -> 90 unless L(q) =2
4:         p = q+2
            oCOND
            plabel = plabel -1
            k = plabel
            store jump
            k = A(q)
                                pJ(7+k,0,6)
            p = p +1
            if A(1) = 6 then p = 2
            q = plabel
            oUI
            k = q
            store label
            -> 1

```

```
sw(7):      comment [TYPE] [NAME LIST] [S]  
            type = 3-A(2)  
            j = 0  
            p = 1  
5:          p = p + 3  
            k = A(p+1)  
            testnst  
            store tag (k)  
            store name  
            n = n+1  
            if A(p) = 1 then -> 5  
            -> 1
```

```

sw(8):      comment [end] [8]
            kkk = flag(level)

50:         comment complete labels jumps
            fetch jump
            if j = -1 then -> 51
            i = j
            find label
            if j ≠ -1 then -> 52
            fault(11)
            print (k,1,0)
            -> 50

52:                                     fill label (i,j)
            ->50

51:         fetch labels

56:         comment clear the decs
            fetch name
            if j = -1 then -> 57
            fetch tag(k)
            if j = 15 then fault(28)
            -> 56 unless 6 ≤ type and type ≤ 10
            jj = k

58:         popup 2 (ST(jj),j,k)
            if j ≠ -1 then -> 58
            return cell (jj)
            -> 56

```

```

57:      comment check cycle - repeat
      fetch cycle
      if j = -1 then -> 59
      fault(13)
      -> 57

59:      comment plant allocation nest up instructions
      if kkk = 1 then                                pJ(8,SL(6),11)
      if 10 > kkk and kkk > 7 then                    pJ(8,SL(11),11)
      fill set (SET(level),n)
      unless kkk = 0 or kkk = 6 then -> 60

                                          pQ(105,level,12)
                                          pQ(64,12,0)
                                          pQ(120,level,2)

      if kkk = 6 then                                pJ(8,SL(4),11)

60:      if kkk = 2 then -> 1
      newline; print (line,4,0); spaces (3*level-3)
      caption END # OF # ; -> END (kkk)
      END(0):  caption BLOCK ; -> 401
      END(1):  caption PROGRAM ; -> 401
      END(6):  caption ROUTINE ; -> 401
      END(7):  caption REAL # FN ; -> 401
      END(8):  caption INTEGER # FN ; -> 401
      END (9):  caption REAL # MAP ; -> 401
      END(10): caption INTEGER # MAP

401:      if level > 2 or perm = 1 then -> 61
      if kkk = 1 then -> 62
      fault(14)
      A(1) = 16
      -> 1

```


61: if kkk $\neq$ 1 then -> 63
 fault(15)
 -> 1

62: kkk = 2

63: level = level - 1
 n = RAL(level)
 if kkk = 0 then -> 1
 if kkk = 2 then -> 50
 copy jump
 store label
 -> 1

sw(9): comment [RT] [NAME] [FPP] [S]

41: kkk = A(2) + 5
 kk = A(4)
 copy tag(kk)
 -> 39 unless i = level
 -> 40 if j = 15 and type = kkk
 fault(7) unless type = 0

39:

CRSPEC(1)

-> 41

40:

j = 0

replace tag(kk)

jj = k

p label = p label -1

k= p label

store jump

pJ(8,0,11)

RAL(level)=n

newline ; print(line,4,0) ; spaces(3*level)

-> RT(type)

RT(6): caption ROUTINE # ; -> 400

RT(7): caption REAL # FN # ; -> 400

RT(8): caption INTEGER # FN # ; -> 400

RT(9): caption REAL # MAP # ; -> 400

RT(10): caption INTEGER # MAP #

400: print name (kk)

if level >= 9 then fault(34)

if level < 15 then -> 411

fault(35) ; stop

411:

level = level +1

R = ca

pQ(121,level,2)

pQ(65,12,0)

pQ(105,12,level)

flag(level) = type

SET(level)=ca

pSET(0)

pQ(122,12,2)

from list 2 (ST(jj),j,k)

n = 2

RA(j) = R

if A(5) = 2 then -> 42

p = 7

type = 1

43:

kkk = A(p)

-> rfpd(A(p+1))

rfpd(1) : type = A(p+4)+5; p = p+1; -> 44

rfpd(2) : type = 4; -> 44

rfpd(3) : type = 12; -> 44

rfpd(4) : type = 2; -> 44

rfpd(5) : type = 3; p = p+2; -> 44

rfpd(6) : type = 11; -> 44

rfpd(7) : type = 1; -> 44

rfpd(8) : type=14; ->44

rfpd(9) : p = p-2

44:

p = p +5

link(jj)

if jj ≠ 0 then -> 45

fault(8)

-> 46

45:

from list 2 (ST(jj),j,jjj)

if j = type then -> 46

fault (9)

```

46:      k = A(p)
        testnst
        if 6 < type and type < 10 then -> 47
        j = 0
        type'=type
        if type=14 then type=2
        store tag(k)
        type=type'
        n = n+1
        store name
        -> 99
47:      newcell(kk)
        j=0
        R = n
        n = kk
        store tag(k)
        store name
        n = R
        pushdown 2(ST(kk),jjj,1000)
99:      p = p+1
        if kkk = 1 then -> 43
42:      link(jj)
        if jj ≠ 0 then fault(10)
        ->1

```

sw(10): comment [RT] [spec] [NAME] [FPP] [S]
 ORSPEC(0)
 ->1

sw(11): comment [spec] [NAME] [FPP] [S]
 ORSPEC(2)
 ->1

sw(12): comment [comment] [TEXT] [S]
 ->1

sw(13):

comment [TYPE'] [array] [ARRAY LIST] [S]

type' = 4

if A(2) $\neq$ 1 then type' = 3

q = 5

15: p = q + 1

10: if A(p) = 2 then -> 9

p = p+3

-> 10

9: kk = 0

p = p+2

11: p = p+2

cSEXP(2)

pQ(67,12,0)

p = p+1

cSEXP(2)

pQ(67,12,0)

kk = kk + 1

if A(p) = 8 then -> 11

if kk $\neq$ 1 then ->700

pQ(97,12,0)

pQ(97,12,0)

pQ(64,12,0)

pQ(72,12,0)

pN(53)
pN(34)
pN(23)
pN(10)
pN(30)
pSET(1)
pN(46)
pN(53)

->701

700:

pSET(kk)
PJ(8,SL(67),13) }

701:

qq = q+2

13:

k = A(qq)

testnst

type = type'

j = kk

store tag(k)

store name

if kk=1 then ->702

PJ(8,SL(68),13)

->703

702:

pN(35)

pQ(121,12,2)

pN(46)

pMS(1,n,level)

pQ(122,12,2)

703:

n = n+1

if A(qq-1) = 2 then -> 12

qq = qq +3

-> 13

12: if kk=1 then ->704

pN(42)

pN(42)

704: if A(q) = 2 then -> 1

q = p+2

-> 15

sw(14): comment [1] [TEXT] [S]

->1

sw(15): comment [begin] [S]

newline; print(line,4,0); spaces(3*level)

caption BEGIN

RAL(level)=n

level = level +1

if level ≥ 10 then fault(34)

if level < 16 then -> 412

fault(35) ; stop

412:

pQ(121,level,2)

pQ(65,12,0)

pQ(105,12,level)

flag(level) = 0

SET(level) = ca

pSET(0)

pQ(122,12,2)

n = 2
->1

sw(16): comment [end of program]

kkk = 1

-> 50

sw(17): comment [NAME] [() [+] [N] [)] [:]

copy tag(A(2))

if type = 5 and i = level then -> 29

fault(4)

->1

29:

kk = k

kkk = A(6)

if A(4) = 2 then kkk = -kkk

if ST(kk) < kkk and kkk < ST(kk+1) then -> 30

fault(5)

->1

30:

if ST(kk+kkk-ST(kk)+2) = 0 then -> 31

fault(6)

->1

31:

ST(kk+kkk-ST(kk)+2) = ca

->1

```

sw(18):      comment [switch] [SWITCH LIST] [s]
              q = 3
28:          p = q +1
25:          if A(p) = 2 then -> 24
              p = p +3
              -> 25
24:          p = p +3
              kkk = A(p +2)
              if A(p) = 2 then kkk = - kkk
              kk = A(p +6)
              if A(p+4) = 2 then kk = - kk
              if kkk < kk then -> 35
              fault(27)
              kk = kkk
35:          qq = q +2
27:          k = A(qq)
              testnst
              R=n
              type = 5
              j = 0
              n = n0
              store tag(k)
              store name
              n = R
              cycle i = n0 + 2,1,n0+kk-kkk+2
              ST(i) = 0
              repeat
              ST(n0) = kkk
              ST(n0+1) = kk

```

$n0 = n0 + kk - kkk + 3$

if $A(qq-1) = 2$ then $\rightarrow 26$

$qq = qq + 3$

$\rightarrow 27$

26:

if $A(q) = 2$ then $\rightarrow 1$

$q = p + 9$

$\rightarrow 28$

sw(19): comment [compile queries] [S]

$qu = 1$

$\rightarrow 1$

sw(20): comment [ignore queries] [5]
 qu = 0
 ->1

sw(21): comment [accept m/c instructions]
 mc switch = 1
 overflag = 0
 ->1

sw(22): comment [N][P][:]
 if mc switch = 1 then ->20
 fault(32)
 ->1
20: if SL(A(2)) = 0 then ->19
 fault(33)
 ->1
19: SL(A(2)) = ca
 ->1

sw(23): comment [*][UCI][S]
P = 3
if mc switch = 1 then ->21
fault(32)

21: ->1
oUCI
->1

sw(24): comment [end of perm][S]
mc switch = 0
perm = 0
line = -1
overflag = 1
-> 1

sw(25): comment [define compiler][S]
line=line-1
* JS 70
* JS 46 P
initialise
** pp curr
* = Q13
* M13
* = +I 13
* MO TO Q13
* Q13

if the 127) on tape

```

** = pp curr
-> 1
70:  ** pt dev; * SET 6; * OUT; * SET 102 ; * = RM13 ; * J71 EN
72:  * = MO M13Q ; * J72 NEN
71:  * SET 101 ; * = M13 ; * C13 ; * NEG ; * = MOM13
    * SET 119 ; * = RM13 ; * J73EJ
74:  * LINK ; * = MOM13Q ; * J74NEJ
73:  * SET 118 ; * = M13 ; * C13 ; * NEG ; * = MOM13
    * SET 135 ; * = RM13 ; * SET 15 ; * = RC14
    * Q15 ; * Q14 ; * Q13 ; * Q12 ; * Q11
    * Q10 ; * Q9 ; * Q8 ; * Q7 ; * Q6
    * Q5 ; * Q4 ; * Q3 ; * Q2 ; * Q1
75:  * = M13M14Q ; * J75C14NZ
    * SET 101 ; * DUP ; * = M13
    * SET 1 ; * = M14 ; * MOM14 ; * SHL 24 ; * SHL-24
    * DUP ; * SET 103 ; * - ; * = RC14 ; * ZERO ; * ZERO
    *
76:  * M13M14Q ; * ; * STR ; * +D ; * J76C14NZS
    * = M13M14Q ; * = M13M14 ; * NEG ; * NOT ; * = M14 ; * = I14
    * SET 418; * = M13; * MO M13
    * SET 4 ; * OUT ; * DUP ; * = C14
    * = C15 ; * SET 4 ; * = M15
    * PMDQ 15 ; comment rewind
    * PMAQ 15 ; comment skip label, call
    * POAQ 14 ; comment dump E101 to top of store
    * PMEQ 15 ; comment backward skip to check parity
    * PARQ 14; * J77NTR
    caption A TAPE A PARITY A FAILURE A ; ->78
77:  * C14 ; * SET 6 ; * OUT ; comment deallocate tape

```

78:

caption A COMPILER A DEFINED A
* SET 38P: ; * = M13; * MO M13
* = Q13; * C13; * SET 116; * OUT
* ZERO; * OUT

sw(26):

comment [*][*][*][cc][s]

if A(5)=1 then ->501

500:

* SET 35P:

* =M13

* MOM13

* =Q13

* MANUAL Q13

line = line-1

-> 1

501:

comment [*] [*] [*] [A] [S]

begin

integer i,j,k,n,no',tape

real t1

integer array W(1:5)

switch S(0:6)

routine spec read key word

routine spec read sym

i=32768

W(1)=1*193105588 + 9710

W(2)=1*538510757 +20658

W(3)=1*116834604 + 5697

W(4)=1*351338496

W(5)=1*672666624

->1 if line=0 ; fault(38) ; stop

cond.

1: ->2 if line<20 ; caption ~~xx~~ FAULTY ~~xx~~ JOB ~~xx~~ HEAD ; stop

2: read key word

 ->S(k)

n. Page

S(0): comment key word unknown

3: ->1 if i=4 ; read symbol(i) ; ->3

S(1): comment EXECUTION

 read(t1)

 read sym

 (III) ->3 unless i=45 or i=51

if i=45 then t1=60*t1

 time limit=int(t1)

 ->3

S(2): comment PARAMETER

 read sym

 ->3 unless i=14

 read(n)

 * SET 8qp:

 * =M13

 **n

 * =MOM13

->3

```
S(3):      comment COMPILER AA or AB
31:        read sym ; ->31 unless i=4
           line=-1
           ->99 ; comment to end of job head block

S(4):      comment JOB
           ** n0
           * SET 92P:
           * =M13
           * =NOM13 ; comment store address of title
           n0'=n0
           n0=n0+1
           newlines(2)
43:        ->44 unless next symbol=4 ; skip symbol ; -> 43
44:        cycle j=1,1,20
           * ZERO
           cycle k=40,-8,0
           read symbol(1)
           print symbol(1)
           ->41 unless i=4 ; ST(n0')=j ; j=20 ; ->42
41:        **i
           * SET +127
           * AND
           **k
           * =C13
           * SHLC13
```

* OR
repeat

42:

** =ST(n0)
n0=n0+1
repeat

->3

S(5):

comment TAPE
tape=1

51:

read key word
->S(k) unless tape=1

* ZERO

cycle j=42,-6,0

read sym

->3 unless 16<1<58

** 1

** j

* =C13

* SHLC13

* OR

repeat

read sym

* SET 89P:

**n

* DUP

* +

* +

* -M12

```
* =MOM13
->53 unless i=14 ; comment write permit
*SET +1
*=MOM13N
53: ->51 if i=4 ; read symbol(1) ; ->53
```

```
routine read sym
1: read symbol(1) ; ->1 if i=65
end
```

routine read key word

```
1: read sym ; ->1 if i=4
line=line+1
->8 if 17<=i<21
tape=0
```

```
2: * ZERO
cycle j=40,-5,0
**i
* SET 32
* -
* DUP
* SHL-5
* J4#Z

**j
* =C13
* SHLC13
* OR
```

****W(k)**
***J6=**
repeat
read sym
repeat
*** ZERO**
*** ERASE**
k=0
*** ERASE**
->10

8: n=1-16
9: k=0

10: end

99: end ; comment end of job head block

->1

routine CRSPEC (integer m)

integer kk,jj,q,jjj,kkk

switch fpd(1:9)

p = A(2) +5

kk = A(5-m)

copy tag(kk)

if m $\neq$ 2 then ->11

if i=level and 6 $\leq$ type $\leq$ 10 then ->33

fault(3)

->1

11: -> 32 if i $\neq$ level

-> 33 if type = p

34: fault(7) unless type = 0

->32

33: from list 2(ST(k),jj,q)

if q $\neq$ 1000 then -> 34

popup 2 (ST(k),jj,q)

-> 35

32: jj = nr

nr = nr +1

newcell(k)

j = 15

type = p

R = n

n = k

store tag(kk)

n = R

R = k

k = kk

store name

```

35:      k = R
      jjj = k
      q = 0
      if A(6-m) = 2 then -> 36
      p = 8-m
      type = 0
37:      kkk = A(p)
      -> fpd(A(p+1))
      fpd(1) : type = A(p+4)+5; p = p+7; -> 101
      fpd(2) : type = 4; -> 38
      fpd(3) : type = 12; -> 38
      fpd(4) : type = 2; -> 38
      fpd(5) : type = 3; p = p+2; -> 38
      fpd(6) : type = 11; -> 38
      fpd(7) : type = 1; -> 38
      fpd(8) : type = 14; -> 38
      fpd(9) : p = p-2
      if type  $\neq$  0 then -> 38
      type = 1
      fault(12)
38:      p = p + 6
      insert after 2 (k,type,0)
102:      q = q+1
      if kkk = 1 then -> 37
36:      pushdown 2 (ST(jjj),jj,q)
      -> 1
101:      insert after 2 (k,type,nr)
      nr = nr + 1
      -> 102
      1: end

```

routine cUI

integer kk,q

switch sw(1:10)

-> sw(A(p))

sw(1): comment [NAME] [APP] [=] [+] [EXPR] [QUERY']

q = p +1

copy tag (A(q))

p = p +3

3:

p = p +1

if L(p) $\neq$ 3 then $\rightarrow$ 3

p = p +1

kk = parity (type)

if kk = 1 then cSEXP(2)

if kk = -1 then cSEXP(1)

unless A(p) = 1 and qu = 1 then $\rightarrow$ 2

if kk = 1 then

pJ(8,SL(20),13)

if kk = -1 then

pJ(8,SL(21),13)

2:

kk = p

p = q

oNAME(1)

p = kk +2

$\rightarrow$ 1

sw(2): comment [NAME] [APP]
 p = p +1
 oNAME(o)
 ->1

sw(3): comment [->] [N]
 k = A(p+2)
 store jump

 p = p +3
 -> 1

pJ(8,0,11)

sw(4): comment [caption] [CAPTION TEXT]

pSET(A(p+2))

pJ(8,SL(3),13)

p = p +3

-> 1

sw(5): comment [return]

unless flag(level) = 6 or perm = 1 then fault(30)

p = p +2

6:

pQ(105,level,12)

pQ(64,12,0)

pQ(120,level,2)

pJ(8,SL(4),11)

->1

```

sw(6):      comment [result] [=] [+] [EXPR]
           unless 7< flag(level)<10 then fault(31)
           p = p +3
           if flag(level) = 7 then -> 5
           cSEXP(2)
           ->6
5:          cSEXP(1)
           ->6

```

```

sw(7):      comment [stop]

           p = p +2
           -> 1

```

pJ(8,SL(6),11)

```

sw(8):      comment [->] [NAME] [(] [+] [EXPR] []]
           copy tag(A(p+2))
           kk = k
           unless type = 5 and i = level then fault(4)
           p = p +4
           cSEXP(2)

```

pSET(kk)

pJ(8,SL(7),11)

p = p +1

1: end

***T

routine cSEXP(integer z) ; comment dated 20/4/65 - FOR VER 4F

comment z=1 for real, z=2 for integer, z=3 for integer if possible

integer c,d,j,m,q',n0',lc,tsf,z'

integer array op(0:5)

switch si(1:4)

routine spec print orders(integer n)

->55 if A(p+5)=0 and A(p+4)=1 and A(p+3)=2 and A(p+2)=2 ; comment single zero

dflag=dflag+1

if dflag=1 then aflag=0

n0'=n0 ; q'=0 ; d=0 ; m=0 ; lc=0 ; tsf=2

op(0)=0

->1 unless A(p)=2

op(1)=13 ; q'=1

1:

p=p+4 ; q'=q'+1 ; comment p to operand + 1

c=A(p-2) ; comment expr alternative

->si(A(p-1))

```

s1(1):      comment p to [NAME]
            ->2 if A(p+1)=1
            copy tag(A(p))
            ->2 unless type=1 or type=2 or type=11 or type=12

            ST(n0)=type ; ST(n0+1)=i ; ST(n0+2)=k
            n0=n0+3
            p=p+3 ; comment APP-CODE + 1 ; ->20

2:          cNAME(2) ; ->6

s1(2):      comment p to [CONST]
            type=A(p)-1 ; if type=0 and A(p+1)=2 and op(q'-1)=15 then type=-1
            ST(n0)= type ; ST(n0+2)=A(p+1)
            ->5 if type ≤ 0
            ST(n0+1)=13 ; lc=1

5:          n0=n0+3 ; p=p+2 ; ->20 ; comment p = CONST-VALUE + 1

```

s1(4): m=1 ; comment | EXPR |

s1(3): p=p+1 ; comment p to ['+']
 z'=3 ; z'=2 if z=2 or op(q'-1) = 15
 cSEXP(z') ; p=p+1 ; comment p to) + 1
 if type=0 then type=2 ; comment nut might write (0)
 ->6 unless m=1

pN(26-2*type)

m=0

6: ->35 if c=2 and q' = 1 and (z≠2 or type=2)
 d=d+1
 ST(n0)=type+2 ; n0=n0+3

pQ(67,12,0)

20: ->21 unless type = 1 or type =11

 tsf=1

if z=2 then fault(24)

if op(q'-1)=15 then fault(39)

21: ->30 if c=2

 op(q')=T(A(p))

if (op(q')=9 or op(q')=15) and z≠2 then tsf=1

22: if op(q') > op(q'-1) +1 then ->1

 ST(n0)=op(q'-1) ; n0=n0+3

 op(q'-1)=op(q') ; q'=q'-1

 ->22

30: ->31 if d=0

 pSET(-d); pQ(122,12,2)

31: ->32 if lc=0

 pQ(121,11,4); pQ(120,13,2)

32: cycle j=q'-1,-1,0

 ST(n0)=op(j)

 n0=n0+3

repeat

cycle j=n0',3,n0-6

 print orders(ST(j))

repeat

->33 if d=0

pSET(-d)

pQ(122,12,2)

33:

type=tsf

35:

->50 unless z=1 and type=2 ;

pQ(121,1,4)

pN(20)

type=1

50:

if n0>n0'+6 then aflag=1

if aflag*dflag*overflag=1 then pJ(9,SL(8),8)

dflag=dflag-1

n0=n0'

->60 ; comment end

55:

p=p+6

pN(33)

type=0

comment to end

routine print orders (integer n)
switch S(-1:15)

-->S(n)

S(-1): ST(j+3)=8 ;

pN(34) ; -->12

S(0): -->1 unless ST(j+3)=13
ST(j+3)=14 ; ST(j+2)=-ST(j+2)

1: n=2

pSET(ST(j+2)) ; -->9

S(1):S(2):S(11):S(12):
n=n-10 ;

pMS(0,ST(j+2),ST(j+1)) ; -->9 if n<3
pQ(120,10,2);pQ(64,10,0) ; -->9

S(3):S(4): n=n-2 ;

pQ(66,12,0) ; -->9

S(5):

pN(74-14*tsf) ; -->12

S(6):

pN(92-31*tsf) ; -->12

S(8):

pN(24*tsf-20)
if tsf=2 then pN(40) ; -->12

S(9):

pN(64-14*tsf)
if tsf=2 then pJ(8,SL(9),6)
-->12

S(13):

pN(27-2*tsf) ; ->12

S(15):

pJ(8,SL(80+tsf),13) ; ->12

9:

->12 unless tsf=1 and n=2 and ST(j+3)≠15

pQ(121,1,4) ; pN(20)

12:S(14): end

60:

end ; comment end of cSEXP

routine cCOND ; comment DATED 17/3/65 - FOR VER 2F

integer c,t

routine spec cCC

routine spec cSC

routine spec cCOMP (integer c)

dflag=dflag+1

if dflag=1 then aflag=0

t=A(p) - 1

c=1 ; if t=2 then c=2 ; comment c=1 for and/or, c=2 for simple cond

p=p+2 ; comment p to ['+'] of SC

cCC

if aflag*dflag*overflag=1 then

pJ(9,SL(8),8)

dflag=dflag-1

->99

routine cCC

integer line

cSC

if c=2 then ->1

line=ca

c=A(p+1) ; p=p+3 ; comment p to ['] of SC

cCC

pN(34)

pJ(8+t,0,6) ; pN(42)

fill label(line,ca)

1:

end

routine cSC

integer line1, line2

switch S(1:3)

->S(A(p-1)) ; comment choice of SC

S(2): cSEXP(3)
 cCOMP(1)
 ->1

S(3): p=p+1
 cCOND
 p=p+1
 ->1

S(1): cSEXP(3)
 cCOMP(2)
 line1=ca

 cCOMP(1)
 line2=ca

1: end

pJ(8,0,6)

pJ(8,0,11)

fill label(line1,ca)

pN(42) ; pQ(121,12,4)

fill label(line2,ca)

routine cCOMP (integer c)
comment c=1 for single cond, c=2 for 2-sided cond
integer t1, comp, ctype
switch S(1:7) 8
 t1=type
 comp=A(p) ; p=p+2
 cSEXP(3)
 ->1 if c=1

1: ctype=31 ; if t1=1 or type=1 then ctype=63
 ->3 unless t1*type=2
 ->2 if t1=1
 c 9
 comp=8-comp

2:

3:

->S(comp)

S(4): S(5):

S(3):

S(6): (7)

S(2):

S(5): (6)

S(1): S(7): 4: end

(8)

99:

end ; comment end of cCOND

0 true 1 false

1 2 3 4 5 6
 = > > ≠ < ≤

= > > ≠ ≠ < ≤
 1 2 3 4 5 6 7

pN(34) ; pN(10)

pN(53)

pQ(121,1,4) ; pN(20)

pN(ctype)

pN(22)

pN(23) ; pN(27) ; ->4

pN(23)

pSH(116,-1) ; ->4

pN(27) ; pN(23)

for ≠

```

routine cNAME(integer z)
integer jj, kk, type', rr, qq, jjj, ii, q, qq, pp
switch sw(0:12), fpt(1:14)
copy tag(A(p))
if i  $\neq$  -1 then -> 3
fault(16)
i = level
j = 0
k = A(p)
type = 0
store tag(k)
store name
k = n
n = n+1
jj=j
if jj = 15 then jj = 0
if z  $\neq$  0 then -> sw(type)
if type = 6 then ->sw(7)
fault(17) unless type = 0
->sw(0)

```

3:


```
sw(5):      comment switch  
            fault(20)  
  
sw(0):      comment NAME NOT SET  
            p = p+1  
  
11:         type = 2  
            if A(p) = 2 then -> 6  
            -> 102
```

```

sw(3):      comment real array
sw(4):      comment integer array
            type' = type
            kk = k
            rr = i
            pp = p
            p = p+1
            if A(p) = 1 then -> 16
            fault(19)
            p = p+2
            -> 10
16:         p = p+1
            q = 0
17:         p = p+2
            q = q+1
            cSEXP(2)
            if A(p) = 8 then -> 17
            if jj  $\neq$  0 then -> 2
            jj = q
            copy tag(A(pp))
            j = jj
            replace tag(A(pp))
2:          p = p +1
            if q = jj then -> 18
            fault(19)
            -> 10

```

18:

if q=1 then
if q≠1 then
if z = 3 then ->10

pMS(0,kk,rr)

pN(46)

pJ(8,SL(66),13)

pQ(120,13,2)

pQ(66-z,13,0)

10:

type = type' -2
-> 1

```

sw(6):  comment routine in expression
        fault(23)
        ->sw(0)

sw(7):  comment real fn and routine
sw(8):  comment integer fn
sw(9):  comment real map
sw(10): comment integer map
        type '= type

20:     jj = k
        from list 2 (ST(jj),rr,k)
        p = p + 1
        if i ≠ 1 or rr > 2 then -> 400
        if A(p) = 2 then -> 402
        p = p + 1
        q = L(p)

411:    if A(p + 2) = 3 and A(p + 4) = 2 and A(p + 5) = 1 then -> 412
        fault (22)

415:    p = p + 2
413:    if L(\\p) = q then -> 414
        p = p + 1
        -> 413

412:    p =p + 6
        cNAME (3)
        if rr = 1 then -> 401

                                pSET(type)
                                pJ(8,SL(12),13)

        if A(p) = 8 then -> 411
414:    p = p + 1
        -> 1

```

401: ~~type~~ = 2
if A(p) ≠ 8 then -> 414
fault (19)
-> 415
400: qq = L(p+1)
if k ≠ 1000 then -> 21
fault(21)
-> 11
21: if A(p) = 1 then -> 22
if k = 0 then -> 23
402: fault(19)
p = p + 2
type = 2
-> 1
22: if k ≠ 0 then -> 24
fault(19)
-> 11
24: q = 0

kk = k
qq = 0
p = p + 1

pQ(96,12,0)

pQ(96,12,0)

26:

```
link(jj)
from list 2 (ST(jj), type, jjj)
p = p + 2
-> fpt(type)

fpt(1) : comment real
         cSEXP(1)
         -> 27

fpt(2) : comment integer
         cSEXP(2)
         -> 27

fpt(5) : comment switch
         fault(0)
         stop

fpt(6) : comment routine
fpt(7) : comment real fn
fpt(8) : comment integer fn
fpt(9) : comment real map
fpt(10): comment integer map
         unlessA(p) ≠ 3 or A(p+2) ≠ 2 or A(p+3) ≠ 1 or A(p+5) ≠ 2 then -> 28

33: k = L(p+2)
    p = p+3
```

30: if L(p) ≤ k then -> 29
 p = p+1
 -> 30
29: fault(22)
 -> 32
28: ii = type
 copy tag(A(p+4))
 if i = -1 then fault(16)
 if ii = type then -> 31
 fault(22)
 p = p+7
 -> 32
31: j = k
 from list 2 (ST(j),i,k)

p = p+7
-> 32

pms(0,1,11)
pms(1,jjj,11)

fpt(11): comment real name

fpt(12): comment integer name

fpt(14): comment addr

ii = type -10

if A(p) $\neq$ 3 or A(p+2) $\neq$ 2 or A(p+3) $\neq$ 1 then -> 33

p = p +4

cNAME(3)

if ii = type or ii=4 then ->27

-> 29

fpt(3): comment real array name

fpt(4): comment integer array name

ii = type

if A(p) $\neq$ 3 or A(p+2) $\neq$ 2 or A(p+3) $\neq$ 1 or A(p+5) $\neq$ 2 then -> 33

copy tag(A(p+4))

if i = -1 then fault(16)

unless ii = type then ->33

cMOD(1,2)

p = p +7

27:

pQ(67,12,0)

qq = qq +1

32:

q = q +1

if q = kk and A(p) = 7 then -> 35

if A(p) $\neq$ 7 then ->36


```

    fault(19)
    p = p+1
    -> 41
36:    if q<kk then -> 26
        fault(19)
40:    if L(p) <qqq then -> 41
        p = p +1
        -> 40
23:    p = p +1
        -> 37
35:
37:
        pSET(-qq-2)
        pQ(122,12,2)
        pSET(rr)
        pJ(8,SL(1),13)

    p = p+1
41:    if type' = 6 then ->1
        if type' ≠ 7 and type' ≠ 8 then ->201
        type = type' -6
        if z ≠ 2 then fault (29)
        ->1
201:    type = type' -8
        if z = 3 then ->1

        pQ(120,13,2)
        pQ(66-z,13,0)

    -> 1

```

```

sw(1):      comment real
sw(2):      comment integer
            cMOD(1,z)
            -> 9
sw(11):     comment real name
sw(12):     comment integer name
            type = type -10
            cMOD(0,z)
9:          p = p+1
            if A(p) = 2 then -> 6
            fault(19)
102:        q = L(p+1)
            p = p+3
8:          if L(p) = q then -> 7
            p = p+1
            -> 8
6:          p = p +1
7:          p = p+1
1:          end

```

routine cmdd (integer n,z)

switch sw(1:6)

-> sw(3n+z)

sw(1): sw(2):

-> 1

sw(3): sw(5):

-> 1

sw(4):

-> 1

sw(6):

1: end

routine fault (integer n)

newline; print symbol (73); print (line,4,0)

spaces (3*level)

caption FAULT #

print (n,1,0); print symbol (73)

faulty = 1

end

pms(0,k,1)

pq(120,13,2)

pq(66-z,13,0)

pms(0,k,1)

pms(1,k,1)

pset(k)

pq(121,1,2)

pn(46)

routine print name (integer i)

integer j,k,n

j = word(i); ** lett(j); * ZERO ; * SHLD+40

** =n ; * SHL -40 ; ** =k ; print symbol (k)

1: j = j+1 ; ** lett(j) ; n = n-6 ; if n<0 then -> 2

4: * ZERO ; * SHLD+8 ; * DUP ; * J3 ≠ Z ; * ERASE ; * ERASE ; * J1

3: ** =k ; print symbol (k) ; -> 4

2: * SET+1 ; ** n ; * - ; * SHL+3 ; * =C13 ; * SHLC13

5: * ZERO ; * SHLD+8 ; * DUP ; * J6 ≠ Z

* ERASE ; * ERASE ; return

6: ** =k ; print symbol (k) ; -> 5

end

routine testnst

integer kk,ii,jj,ttype

kk = k

ii = i

jj = j

ttype = type

copy tag(k)

if i =level and type $\neq$ 0 then fault(7)

k = kk

i = ii

j = jj

type = ttype

end

routine cUCI

integer q,i,j

switch e(1:100), f(1:20), g(1:8), e2(1:10), f2(1:12), iom(7:24)

-> e(A(p))

e(1): comment [*] [α] [NAME] [APP]

p = p +3

cNAME(3)

-> 9

e(2): comment [*] [='] [NAME] [APP]

p = p +4

cNAME (A(p-2))

-> 9

e(3): comment [J] [N] [P'] [J-INSTR]

if A(p+3) = 1 then -> 10

k = A(p+2)

store jump

k = 0

11: p = p +5

-> f(A(p))

10: k = SL(A(p+2))

-> 11

f(1): comment JrCqNZ5

fetch jump

pQ(127,A(p+2),0)

-> 9

f(2): comment JrCqNZ ; pJ(11,k,A(p+2)) ; -> 9

f(3): comment JrCqZ ; pJ(10,k,A(p+2)) ; -> 9

f(4):	<u>comment</u> Jr = Z ;	pJ(9,k,6)	;	-> 9
f(5):	<u>comment</u> Jr = ;	pJ(9,k,1)	;	-> 9
f(6):	<u>comment</u> Jr ≠ Z ;	pJ(8,k,6)	;	-> 9
f(7):	<u>comment</u> Jr ≠ ;	pJ(8,k,1)	;	-> 9
f(8):	<u>comment</u> Jr > Z ;	pJ(9,k,4)	;	-> 9
f(9):	<u>comment</u> Jr > Z ;	pJ(8,k,2)	;	-> 9
f(10):	<u>comment</u> Jr < Z ;	pJ(9,k,2)	;	-> 9
f(11):	<u>comment</u> Jr ≤ Z ;	pJ(8,k,4)	;	-> 9
f(12):	<u>comment</u> JrV ;	pJ(9,k,8)	;	-> 9
f(13):	<u>comment</u> JrNV ;	pJ(8,k,8)	;	-> 9
f(14):	<u>comment</u> JrEN ;	pJ(9,k,10)	;	-> 9
f(15):	<u>comment</u> JrNEN ;	pJ(8,k,10)	;	-> 9
f(16):	<u>comment</u> JrEJ ;	pJ(9,k,12)	;	-> 9
f(17):	<u>comment</u> JrNEJ ;	pJ(8,k,12)	;	-> 9
f(18):	<u>comment</u> JrTR ;	pJ(9,k,14)	;	-> 9

f(19): comment JrNTR ; pJ(8,k,14) ; -> 9

f(20): comment Jr ; pJ(8,k,11) ; -> 9

e(4): comment [SETB] [OCTAL]

pSET(A(p+2))

-> 9

e(5): comment [SET][N][P][:]

pSET(SL(A(p+2)))

-> 9

e(6): comment [JS] [N] [P']

if A(p+3) = 1 then -> 12

k = A(p+2)

store jump

pJ(8,0,13)

-> 9

12:

pJ(8,SL(A(p+2)),13)

-> 9

e(7): comment [EXIT] [N]

pJ(8,8192 parity (A(p+2))+intpt (A(p+2)/2)+8192,15)

-> 9

e(8): comment [='] [M] [N] [M] [N] [MS-INSTR]

i = 0

if A(p+1) = 1 then i = 1

q = A(p+6)

k = A(p+4)

p = p+7

-> g(A(p))


```

g(1):      comment MkmQ QHN ;      pQ(78+1,q,k) ;      -> 9

g(2):      comment MkmQ QH ;      pQ(70 + 1,q,k) ;      -> 9

g(3):      comment MkmQ QN ;      pQ(74+1,q,k) ;      -> 9

g(4):      comment MkmQ ;      pQ(66+1,q,k) ;      -> 9

g(5):      comment MkmQ HN ;      pQ(76+1,q,k) ;      -> 9

g(6):      comment MkmQH ;      pQ(68+1,q,k) ;      -> 9

g(7):      comment MkmQN ;      pQ(72+1,q,k) ;      -> 9

g(8):      comment MkmQ ;      pQ(64+1,q,k) ;      -> 9

e(9):      comment [SH] [SH-INSTR]
           p = p +2
           -> e2 (A(p))

e2(1):      comment SHA + n
           k = A(p+4)
           if A(p+2) = 2 then k = -k
                                           pSH(113,k)
           -> 9

e2(2):      comment SHAD + n
           k = A(p+4)

```

if A(p+2) = 2 then k = -k

pSH(114,k)

-> 9

e2(3):

comment SHL + n

k = A(p+4)

if A(p+2) = 2 then k = -k

pSH(116,k)

-> 9

e2(4):

comment SHLD + n

k = A(p+4)

if A(p+2) = 2 then k = -k

pSH(118,k)

-> 9

e2(5):

comment SHC +n

k = A(p+4)

if A(p+2) = 2 then k = -k

pSH(119,k)

-> 9

e2(6):

comment SHAC q

pQ(113,A(p+2),0)

-> 9

e2(7):

comment SHADCq

pQ(114,A(p+2),0)

-> 9

e2(8): comment SHLCq

-> 9

pQ(116,A(p+2),0)

e2(9): comment SHLDCq

-> 9

pQ(118,A(p+2),0)

e2(10): comment SHCCq

-> 9

pQ(119,A(p+2),0)

e(10): comment [=] [QS-INSTR]

p = p+2

-> f2(A(p))

f2(1): comment = LINK

-> 9

pQ(124,0,0)

f2(2): comment = Qq

i = 120

j = 14

2:

-> 9

pQ(i,A(p+2),j)

f2(3): comment = Cq
 i = 120 ; j = 8 ; -> 2

f2(4): comment = Iq
 i = 120 ; j = 4 ; -> 2

f2(5): comment = Mq
 i = 120 ; j = 2 ; -> 2

f2(6): comment = RCq
 i = 120 ; j = 9 ; -> 2

f2(7): comment = RIq
 i = 120 ; j = 5 ; -> 2

f2(8): comment = RMq
 i = 120 ; j = 3 ; -> 2

f2(9): comment = +Qq
 i = 122 ; j = 14; -> 2

f2(10): comment = +Cq
 i = 122 ; j = 8 ; -> 2

f2(11): comment = +Iq
 i = 122 ; j = 4 ; -> 2

f2(12): comment = + Mq
 i = 122 ; j = 2 ; -> 2

e(11): comment Qk TO Qq
i = 111

3:
-> 9

pQ(1,A(p+2),A(p+4))

e(12): comment Ck TO Qq
i = 108 ; -> 3

e(13): comment Ik TO Qq
i = 106 ; -> 3

e(14): comment Mk TO Qq
i = 105 ; -> 3

e(15): comment IMk TO Qq
i = 107 ; -> 3

e(16): comment CMk TO Qq
i = 109 ; -> 3

e(17): comment Clk TO Qq
i = 110 ; -> 3

e(18): comment [I] [N] [=] [±'] [N]
 k = A(p+6) ; i = 0
 if A(p+4)=2 then k=-k
 if k = 1 then i = 4
 if k = -1 then i = 5
 if k = 2 then i = 6
 if k = -2 then i = 7
 if i = 0 then fault(33)

pQ(96+1,A(p+2),0)

->9

e(19): comment Qq
 i = 121 ; j = 14 ; -> 2

e(20): comment Cq
 i = 121 ; j = 8 ; ->2

e(21): comment Iq
 i = 121 ; j = 4 ; -> 2

e(22): comment Mq
 i = 121 ; j = 2 ; -> 2

e(23): comment NCq
 i = 98 ; j = 0 ; -> 2

e(24): comment DCq
 i = 99 ; j = 0 ; -> 2

e(25): comment LINK

pQ(123,0,0)

->9

e(26): comment M + Iq

p = p+3 ; i = 95 + A(p-1) ; j = 0

-> 2

e(27): comment [x+] [+'] [N]

k = A(p+4)

if A(p+2) = 2 then k = -k

pSH(115,k)

->9

e(28): comment x+Cq

i = 115; j = 0 ; ->2

e(29): comment ROUNDHF ; pN(17) ; ->9

e(30): comment ROUNDF ; pN(16) ; ->9

e(31): comment ROUNDH ; pN(12) ; ->9

e(32): comment ROUND ; pN(14) ; ->9

e(33): comment FLOATD ; pN(21) ; ->9

e(34): comment FLOAT ; pN(20) ; -> 9

e(35):	<u>comment</u> ERASE	:	pN(42)	:	->9
e(36):	<u>comment</u> REVD	:	pN(41)	:	->9
e(37):	<u>comment</u> REV	:	pN(53)	:	->9
e(38):	<u>comment</u> ZERO	:	pN(33)	:	->9
e(39):	<u>comment</u> DUPD	:	pN(35)	:	->9
e(40):	<u>comment</u> DUP	:	pN(34)	:	->9
e(41):	<u>comment</u> NEGDF	:	pN(57)	:	->9
e(42):	<u>comment</u> NEGF	:	pN(25)	:	->9
e(43):	<u>comment</u> NEGOD	:	pN(8)	:	->9
e(44):	<u>comment</u> NEG	:	pN(23)	:	->9
e(45):	<u>comment</u> ABSF	:	pN(24)	:	->9
e(46):	<u>comment</u> ABS	:	pN(22)	:	->9
e(47):	<u>comment</u> MAXF	:	pN(58)	:	->9
e(48):	<u>comment</u> MAX	:	pN(26)	:	->9

•(49): comment SIGN ; pN(63) ; ->9
•(50): comment SIGN ; pN(31) ; ->9
•(51): comment CAB ; pN(54) ; ->9
•(52): comment FRB ; pN(55) ; ->9
•(53): comment STAND ; pN(56) ; ->9
•(54): comment + DF ; pN(51) ; ->9
•(55): comment + D ; pN(49) ; ->9
•(56): comment + F ; pN(50) ; ->9
•(57): comment + R ; pN(52) ; ->9
•(58): comment + I ; pN(36) ; ->9
•(59): comment + ; pN(48) ; ->9
•(60): comment +DF ; pN(19) ; ->9
•(61): comment +D ; pN(47) ; ->9
•(62): comment +F ; pN(60) ; ->9
•(63): comment + ; pN(46) ; ->9

•(64): comment -DF ; pN(18) ; ->9
•(65): comment -D ; pN(43) ; ->9
•(66): comment -F ; pN(61) ; ->9
•(67): comment - ; pN(30) ; ->9
•(68): comment xDF ; pN(5) ; ->9
•(69): comment xD ; pN(28) ; ->9
•(70): comment xF ; pN(4) ; ->9
•(71): comment x+F ; pN(7) ; ->9
•(72): comment x ; pN(29) ; ->9
•(73): comment = TR ; pN(2) ; ->9
•(74): comment BITS ; pN(3) ; ->9
•(75): commentDUMMY ; pN(15) ; ->9
•(76): comment PERM ; pN(10) ; ->9
•(77): comment TOB ; pN(11) ; ->9
•(78): comment OR ; pN(9) ; ->9

e(79): comment VR ; pN(1) ; ->9
 e(80): comment NEV ; pN(13) ; ->9
 e(81): comment NOT ; pN(27) ; ->9
 e(82): comment FIX ; pN(37) ; ->9
 e(83): comment STR ; pN(39) ; ->9
 e(84): comment CONT ; pN(40) ; ->9
 e(85): comment AND ; pN(44) ; ->9
 e(86): comment EXIT ; pJ(8,16384,15) ; ->9
 e(87): comment OUT ; pJ(8,0,9) ; ->9
 e(88): comment x+ ; pSH(115,0) ; ->9

e(89): comment [P][IOM][A TO F][Q][N]
 -> iom(6A(p+2)+A(p+4))

iom(7): comment PIAQq
 i = 84 ; j = 0

i: pQ(i,A(p+7),j) ; -> 9

iom(8): comment PIBQq
 i = 85 ; j = 0 ; -> 1

iom(9): comment PICQq
 i = 84 ; j = 8 ; -> 1

iom(10): comment PIDQq
i = 85 ; j = 8 ; -> 1

iom(11): comment PIEQq
i = 86 ; j = 0 ; -> 1

iom(12): comment PIFQq
i = 87 ; j = 0 ; -> 1

iom(13): comment POAQq
i = 88 ; j = 0 ; -> 1

iom(14): comment POBQq
i = 89 ; j = 0 ; -> 1

iom(15): comment POCQq
i = 88 ; j = 8 ; -> 1

iom(16): comment PODQq
i = 89 ; j = 8 ; -> 1

iom(17): comment POEQq
i = 88 ; j = 8 ; -> 1

iom(18): comment PCFQq
i = 88 ; j = 4 ; -> 1

iom(19): comment PMAQq
i = 92 ; j = 0 ; -> 1

iom(20): comment PMEQq
i = 80 ; j = 8 ; -> 1

iom(21): comment PMCQq
i = 80 ; j = 4 ; -> 1

iom(22): comment PMDQq
i = 94 ; j = 8 ; -> 1

iom(23): comment PMEQq
i = 94 ; j = 0 ; -> 1

iom(24): comment PMFQq
i = 82 ; j = 0 ; -> 1

e(90): comment TLOQq
i = 84 ; j = 4 ; -> 2

e(91): comment INTQq
i = 92 ; j = 2 ; -> 2

e(92): comment PARQq
i = 81 ; j = 0 ; -> 2

e(93): comment BUSYQq
i = 80 ; j = 2 ; -> 2

e(94): comment MANUALQq
i = 80 ; j = 1 ; -> 2

e(95): comment CTQq
i = 80 ; j = 0 ; -> 2

e(96): comment CLOQq
i = 84 ; j = 2 ; -> 2

e(97): comment [SET][+][N]
k = A(p+4)
if A(p+2)=2 then k = -k

pSET(k)

-> 9
e(98): comment [advca][N]
cycle i = 1,1,2A(p+2)

pSET(o)

repeat
-> 9

e(99):

comment [OCTAL][L/][OCTAL][L/][OCTAL]

cycle i = p+1,2,p+5

j = A(i)

**j

* SHL-8

** = q

pN(q) ; pN(j)

repeat

-> 9

e(100):

comment p

8:

** pp curr

* = Q13

* J9 C13 Z

pN(15)

-> 8

9:

end

***T

routine find label

** k

** label(level)

* = Q15

* M15

* J1<Z

3: * I15

** VO

* AND

* J2=

* M15

* J1<Z

* M9M15

* = Q15

-> 3

2: * C15

** VO

* AND

-> 4

1: * SET -1

4: ** = j

* ERASE

end

routine copy tag (integer kk)

** tags (kk)

* = Q15

* M15

* J1<Z

** VO

* I15

* AND

* C15

* DUP

* SET 15

* AND

* REV

* SHL-4

* DUP

* SET 15

* AND

* REV

* SHL-4

* SET 255

* AND

-> 2

1: * SET-1

* DUP

* DUPD

2: ** = type

** = i

** = j

** = k

end

routine replace tag (integer kk)

 ** tags(kk)

 * = M15

 * M15

 * J1>Z

 * MO TO Q15

1: ** k

 * = I15

 ** type

 * SET 255

 * AND

 * SHL 8

 ** i

 * SET 15

 * AND

 * SHL 4

 ** j

 * SET 15

 * AND

 * OR

 * OR

 * = C15

 * Q15

 ** = tags(kk)

end

routine from list 2 (integer name cell, s1,s2)

** cell

* = Q15

* M15

* J1<Z

** VO

* DUP

* C15

* AND

* REV

* I15

* AND

-> 2

1: * SET -1

* DUP

2: ** = s2

** = s1

end

routine fetch tag (integer kk)

integer z

copy tag (kk)

popup 2 (tags(kk),z,z)

end

routine popup 2 (integer name cell, s1,s2)

** cell

* = Q15

* M15

* DUP

* J1<Z

** VO

* DUP

* C15

* AND

** = s1

* I15

* AND

** = s2

* J2=Z

* MQM15

** = cell

* C9

* = MQM15

* M15

* = C9

-> 3

2: * SET -1

** = cell

-> 3

1: * DUP

** = s1

** = s2

3:end

routine store tag (integer kk)

integer z

**type

* SET 255

* AND

* SHL8

** level

* SET 15

* AND

* SHL 4

** j

* SET 15

* AND

* OR

* OR

** = z

pushdown 2 (tags(kk),z,n)

end

routine pushdown 2 (integer name cell, integer s1,s2)

```
    ** cell
    * DUP
    * = C15
    * C15
    * J1<Z
    * J2 C9 NZ
more space
2:   * C9
    * DUP
    * I9
    * -
    * J3>Z
    * DUP
    * = I9
3:   * = M15
    * MQM15
    * = C9
    * = MQM15
4:   ** s1
    * = C15
    ** s2
    * = I15
    * Q15
    ** = cell
    -> 5
1:   * NO TO Q15
    * ERASE
    -> 4
```

5:end

routine link (integer name stad)

** stad

* DUP

* J1<Z

* = M15

* M9M15

* = C15

* C15

1: ** = stad

end

routine more space

newline

caption AVAILABLE & SPACE & LIST & EMPTY

newline

stop

end

routine new cell (integer name stad)

* J1 C9 NZ

more space

1: * C9

* DUP

** = stad

* DUP

* I9

* -

* J2>Z

* DUP

```
      * = I9
2:    * = M15
      * M9M15
      * = C9
      * SET-1
      * = M9M15
```

end

routine return cell (integer name stad)

```
  ** stad
  * DUP
  * = M15
  * C9
  * = M9M15
  * = C9
```

end

routine insert after 2 (integer name stad, integer s1, s2)

integer i

```
  i = stad
  link (stad)
  if stad = 0 then -> 1
  if stad = -1 then stad = i
  -> 2
```

1: newcell (stad)

```
  **i
```


* = M15

* M9M15

** stad

* OR

 ** = M9M15

2: pushdown 2(ST(stad),s1,s2)

end

routine store cycle
pushdown 2(cycle(level),ca,k)
end

routine fetch cycle
popup 2(cycle(level),j,k)
end

routine store label
pushdown 2(label(level),ca,k)
end

routine store jump
push down2(jump(level),ca,k)
end

routine store name
pushdown 2(name(level),O,k)
end

routine fetch jump
popup 2(jump(level),j,k)
end

routine fetch labels
1: popup 2(label(level),j,k)
if j $\neq$ -1 then -> 1
end

```

routine fetch name
  popup 2(name(level),j,k)
end

```

```

routine copy jump
  from list 2(jump(level),j,k)
end

```

```

routine pSET (integer n)
  !: ** n
    * DUP
    * SHL-8
    * SET 196
    * SET 3
    * = C14
    * JS 50 P

```

10: end

→ 1 ~~7~~ |n| < 1 ; ~~PN~~(33)
 → 10 ~~4~~ n = 0 ; ~~PN~~(27)
 → 10 ~~4~~ n < 1 ; ~~PN~~(23)
 → 10

```

routine pSH (integer a,n)
  ** n
  * NOT
  * SHL 1
  * NOT
  ** a
  * SET 2
  * = C14
  * JS 50 P

```

end

routine pN (integer a)

** a

* SET 1

* = C14

* JS 50 P

end

routine pQ (integer a,q,qq)

** q

* SHL 4

** qq

* SET 15

* AND

* OR

** a

* SET 2

* = C14

* JS 50 P

end

routine fill label (integer at, equals)

** equals

** at

* JS 51 P

end

routine fill set (integer at, equals)

** equals

** at

* JS 52 P

end

routine pJ (integer a,s,b)

** s

* DUP

* SHL-8

* DUP

* SET 15

* AND

** b

* SHL 4

* OR

* REV

* DUP

* SET 16

* AND

* SHL-1

* REV

* SHL-5

* SET 7

* AND

** a

* SHL 4

* OR

* OR

* SET 3

* = C14

* JS 50P

end

routine pMS (integer a,m,q)

** m

* DUP

* DUP

* SHL-8

* SET 15

* AND

** q

* SHL 4

* OR

** a

* SET 7

* AND

* SET 192

* OR

* CAB

* SHL-9

* SET 56

* AND

* OR

* SET 3

* = C14

* JS 50P

end

1: end

end

end of program

***Z

////////////////////