```
" stat

arg = 017777

    lac arg
    tad d1
    dac name

loop;
    lac arg i
    sad d4
    sys exit
    tad dm4
    dac arg i
    lac name
    tad d4
    dac name

    lav statbuf
    sys status; dotdot; name: ..
    spa
    jmp error
    lav obuf-1
    dac 12
    lac ii
    lmq
    -3
    jms octal
    lac iflags
    lmq
    -2
    jms octal
    lac iuid
    lmq
    -2
    jms octal
    -1
    tad iniks
    cma
    lmq
    -2
    jms octal
    lac isize
    lmq
    -5
    jms octal
    lac o12
    dac obuf+18
    lac d1
    sys write; obuf; 19
    jmp loop

error;
    lac name
    dac 1f
    lac d1
    sys write; 1:..; 4
    lac d1
    sys write; 1f; 2
    jmp loop
1: 040077; 012
```

BA19

BA20

```
octal; 0
    dac c
    lav tbuf-1
    dac 8
    lac o40
    dac 8 i
1;
    lacq
    and o7
    tad o60
    dac 8 i
    lrs 3
    isz c
    jmp 1b
    lac 8
    dac c
1;
    lac c i
    dac 12 i
    sad o40
    jmp octal i
    -1
    tad c
    dac c
    jmp 1b

dotdot;
    056056; 040040; 040040; 040040

d1; 1
d2; 2
d3; 3
d8; 8
d14; 14
d12; 12
o12; 012
o60; 060
o40; 040
o7; 7
d4; 4
dm4; -4

c; .=.+1

statbuf:
    iflags; .=.+1
    .=.+7
    iuid; .=.+1
    inlks; .=.+1
    isize; .=.+1
    .=.+1
    ii; .=.+1

obuf; .=.+19
tbuf; .=.+10
```

8A19

8A20

```
" tm

    sys time
    cll
    div
    216000
    dac t1
    lacq
    cll
    idiv
    10
    tad o60
    dac buf+1
    lacq
    cll; idiv; 10
    tad o60
    dac buf
    lacq
    sna
    jmp 1f
    tad o60
    alss 9
    xor buf
    dac buf
1:
    lac t1
    cll
    idiv
    6
    lacq
    cll
    idiv
    10
    tad o56060
    dac buf+6
    lacq
    cll
    idiv
    10
    tad o60
    dac buf+5
    lacq
    cll
    idiv
    6
    tad o72060
    dac buf+4
    lacq
    cll
    idiv
    10
    tad o60
    dac buf+3
    lacq
    cll
    idiv
    6
    tad o72060
    dac buf+2
    lac d1
    sys write; buf; 8
    sys exit
```

BA20

```
d1: 1
d13: 13
d2: 2
o60: 060
o56060: 056060
o72060: 072060
t1: 0
t2: 0
buf: .=.+7; 012
```

BA20

```
t=0
main:
  lac 017777 i
  sad d4
  jmp 2f
  sad d8
  jmp 1f
  law 9
  tad 017777
  dac 0f
  law 017
  sys creat;0:0
  dac output
1:law 5
  tad 017777
  dac 0f
  sys open;0:0;0
  dac input
  spa
  sys exit
2:
  jms advance; jmp 1f
  jmp rinterp

0:lac 2f
  jms obuild
1:lac owrite
  spa
  jmp 0b
  jms oflush
  las
  spa
  sys exit
  sys save
2:o777


"special machine language code

"puts out and octal strin g from symtab entry

symoct:
  lac equ
  add equbot
  dac 9f+t
  lac 1f
  jms twoktab; lac 9f+t i
1:gf  .+1 x
  lac ii
  dac 8
  lac 8 i
  add symbot
  dac 9f+t

2:lac 9f+t i
  jms putoct
  lac onenl
  jms obuild
  lac 9f+t i
  and o600600
  sza    "skip unless word contained eof (0777)
  jmp ggoon
```

```
  isz 9f+t
  jmp 2b
t=t+1
o600600:0600600
```

0821
0822
0823

```
"    recognition stack frame advance

advance:0
  lac frame
  dac 8
  lac advance i
  dac 8 i
  lac ii
  dac 8 i
  lac ignore
  dac 8 i
  lac j
  dac 8 i
  lac k
  dac 8 i
  lac frame
  dac nframe i
  lac nframe
  dac frame
  add dffrmsz
  dac nframe
  dac nframe
  jms between; add rbot; add rtop
  jms halt
  isz advance
  jmp advance i

retreat:
  dzm junk
  lac gflag
  sza
  jmp 1f
  jms bundlep
  dac junk
1:lac frame
  dac nframe
  lac frame i
  dac frame
  dac 8
  lac 8 i
  dac 3f   "retrun address
  lac 8 i
  dac ii
  lac 8 i
  dac ignore
  lac fflag
  sna
  jmp 2f
  lac 8 i   "restore j and k on failure
  dac j
  lac 8 i
  dac k
2:lac junk
  sna
  jmp 3f
  dac nframe i   "stass reslts
  isz nframe
3:jmp

" bundle up results and return single pointer to them in ac
" return 0 if no results
```

```
bundlep:0
  lac fflag
  sza
  jmp 2f   "no results on failure
  jms nframe0
  dac 9f+t
  cma
  tad nframe
  cma
  dac 9f+t+1
  sma
  jmp 2f
  sad m1
  jmp 3f   "only one result, no bundling necessary
  lac 9f+t
  tad m1
  dac 8

1:lac 8 i
  jms kput
  isz  9f+t+1
  jmp 1b
  lac k   "make up result pointer
  add 1.gk
  jmp bundlep i

2:cla
  jmp bundlep i
3:lac 9f+t i
  jmp bundlep i
t=t+1   "where to find results
t=t+1   "negative of result count

"     the main interpreter loop
" locate original value of nframe for present stack level.
nframe0:0
  jms s1get; add d,ii
  dac junk
  lac junk i
  dac junk
  lac junk i
  and opmask
  sad 1.rw
  jmp 1f
  lac refrsz
  jmp 2f
1:lac junk i
  and o17777
2:add frame
  jmp nframe0 i

halt:0
  lac 1f
  jms obuild
  lac halt
  jms putoct
  lac onenl
  jms obuild
  xct rstack+1
1:.+1;012077;end
```

```
rinterp:
  las   "trace check
  and d5
  sna
  jmp .+3
  lac bugr
  jms bug

  lac fflag
  ral
  lac ii i
  and opmask
  sad l.ra
  jmp rera
  sad l.rb
  jmp rerb
  szl
  jmp retreat
  lrs 14
  and o17
  add rbranch
  dac .+1
  jmp

rbranch:
  jmp .+1 i
  reno
  rerx
  rerc
  regc
  rerf
  rerw
  rera
  reuu
  rero
  rerm
  rers
  rerv
  reuu
  reuu
  reuu
  reuu

reuu:
  jms halt

rerb:
  cml
rera:
  dzm fflag
  snl
  jmp goon
  jms aget
  dac ii
  jmp rinterp

backup:
  lac jsav
  dac j
nuts:
  lav
```

B21 · B22 · B23

```
  dac fflag

reno:
goon:
  lac ii i
  isz ii
  and exitmask
  sza
 jmp retreat
  jmp rinterp


rerw:
  jms aget
  add frame
  dac nframe
  jmp goon

rerc:
  jms advance; jmp goon
  jms aget
  dac ii
  jmp rinterp


gegf:
rerf:
  jms aget
  add ljmp
  dac .+1
  jmp

rerx:
  lac j
  dac jsav
  jms aget
1:dac 9f+t
  jms lchar
  sad o777
  jmp goon
  dac 9f+t+1
  jms getj
sad 9f+t+1
  jmp 2f
  jmp backup
2:lac 9f+t
  add o400000
  jmp 1b
t=t+1   "address of next comparison char
t=t+1   "character itself

aget:0
  lac ii i
  and o17777
  jmp aget i

regc:
  lac ii i
  and o757777
  xor exitmask
  dac nframe i
```

```
  isz nframe
  jmp goon

kput:0
  isz k
  dac junk1
  lac k
  jms between; add d0; add kmax
  jms halt
  add kbot
  dac junk
  lac junk1
  dac junk i
  jmp kput i

s0get:0
  lac frame
  xct s0get i
  dac junk
  lac junk i
  isz s0get
  jmp s0get i

s1get:0
  lac frame i
  xct s1get i
  dac junk
  lac junk i
  isz s1get
  jmp s1get i

s0put:0
  lmq
  lac frame
  xct s0put i
  dac junk
  lacq
  dac junk i
  isz s0put
  jmp s0put i
"     here is the generaion interpreter
"     the k table cant move while its active

geno:
ggoon:
  lac ii i
  isz ii
  and exitmask
  sza
  jmp retreat
ginterp:
  las   "trace check
  and d6
  sna
  jmp .+3
  lac bugg
  jms bug

  lac ii i
  lrss 14
  and o7
```

821
822
823

```
  add gbranch
  dac .+1
  jmp

gbranch:
  jmp .+1 i
  geno
 gegx
  geuu
  gegc
  gegf
  gegk
  gegp
  gegq

geuu:
  jms halt

gegx:
  lac ii i
  and o417777
  jms obuild
  jmp ggoon

gegq:
  jms advance; jmp ggoon
  lac env
  add d.ii
  dac junk
  jms aget
  add junk i
  dac junk
  lac junk i
  dac ii
  lac env
  add d.env
  dac junk
  lac junk i
  dac env
  jmp ginterp

gegp:
  jms advance; jmp ggoon
  lac env
  add d.ii
  dac junk
  lac frame i
  dac env
  jms aget
cma
  add junk i
  dac ii
  jmp ginterp

gegk:
  lac ii i
  jms aget
  add kbot
  dac ii
  jms s0put; add d.ii
  lac frame
```

821
822
823

```
  dac env
  jmp ginterp

gegc;
  jms advance; jmp ggoon
  jms aget
  dac ii
  jmp ginterp

bug;0
  dac 1f+2
  lac onenl
  jms obuild
  lac ii
  jms putoct
  lac ii i
  lrs 14
  and o17
  add 1f+2
  dac 1f+2
  lac 1f+2 i
  dac 1f+2
  lac 1f
  jms obuild
  lac ii i
  jms putoct
  las
  and d4
  sza
  jms halt
  jmp bug i

1;0400000 .+1; 040; 0; 040777
```

8221.
8222
8223

```
move: 0
  dac 9f+t
  lac move i
1:dac 2f
  lac 9f+t
  jms lchar
  dac 9f+t+1
  jms dchar
2: 0
  lac 9f+t+1
  sad o777
  jmp 3f
  lac o400000
  add 9f+t
  dac 9f+t
  lac o400000
  add 2b
  jmp 1b

3:isz move
  lac 2b
  jmp move i
t=t+1   "source address
t=t+1   "source character

lchar:0
  dac junk
  ral
  lac junk i
  snl
  lrs 9
  and o777
  jmp lchar i

dchar:0
  lmq
  lac dchar i
  dac junk
  spa
  jmp 1f
  llss 9
  lac o777
  jmp .+2
1:0777000
  and junk i
  omq
  dac junk i
  isz dchar
  jmp dchar i


"     gets designated character from input
jget:0
1:lac j
  jms cbetween; add jmin; add jmax
  jmp jmore
  cma
  add jmin
  cma
  add jbot
  jms lchar
  jms class; add ignore
```

```
  jmp jget i
  lac j
  add o400000
  dac j
  jmp 1b

" read more input - filthy code, enough to make disk &
" terminal input work.  Theae only deliver full count
" except at eof or 1 word

jmore:
and o377700
  dac jmin
  add ljsiz
  dac 9f+t
  lac jmax
  jms cbetween; add jmin; add 9f+t
  lac jmin
  dac jmax
  dac 1f
  cma
  add jmin
  cma
  add jbot
  dac 2f
  lac input
  sys seek
1:0;0
  =1
  dac 2f i
  lac input
  sys read
2:0;jsiz
  sna
  lac d1
  add jmax
  dac jmax
  jmp jget+1
t=t+1

"     gets next character from input

getj:0
  lac j
  jms jget
  dac junk
  lac j
  add o400000
  dac j
  lac junk
  jmp getj i

" compare two strings - assume both left justified

comp:0
  dac 9f+t
  lac comp i
  dac 9f+t+1
  isz comp
1:lac 9f+t i
  sad 9f+t+1 i
```

```
  jmp 3f
  and 9f+t+1 i    "do both start with eof?
  spa
2:isz comp
  jmp comp i
3:and o600600   "is there an eof?
  sza
  jmp 2b
  isz 9f+t
  isz 9f+t+1
  jmp 1b
t=t+1    "address of string 1
t=t+1   "address of string 2


obuild:0
  lmq
  lac ovrite
  add obot
  dac 2f
  lacq
1:jms move
2:0
  cma
  add obot
  cma
  dac ovrite
  jms cbetween; add d0; add omax
   skp
  jmp obuild i

  lac lochunk
  jms oflush
  lac obot
  dac 2b
  add lochunk
  jmp 1b

oflush:0
  dac 2f
  lac obot
  dac 1f
  lac output
  sys write
1:0
2:0
  jmp oflush i

" outputs octal string from designated value

octal:
  isz ii
  lac ii i
  dac 2f
  lac 1f
  jms twoktab
  lac 2f i

1:gf geoctal x
2:0
```

```
geoctal:
  lac ii
  dac 8
  lac 8 i
  jms putoct
  jmp ggoon

" converts word in ac into ocatl on output stream

putoct:0
  dac 9f+t
  lac 7f
  jms obuild
  dzm 9f+t+2
  -6
  dac 9f+t+1

1:lac 9f+t
  cll
  lrs 15
  add o60
  dac 8f+1
  lls 18
  dac 9f+t
  lac 9f+t+2   "have nonzero digits been seen?
  sza
  jmp 2f
  lac 8f+1   "no,is this nonzero?
  sad o60
  jmp 3f   "no
2:lac 8f
  jms obuild
  lav
  dac 9f+t+2
3:isz 9f+t+1
  jmp 1b
  jmp putoct i
t=t+1   "value to convert
t=t+1   "digit count
t=t+1   "nonzero digit flag
7: .+1; 060777
8:0400000 .+1;0;end


eof:
  lac j
  dac jsav
  jms jget
  sad o777
  jmp goon
  jmp backup
class:0
  dac junk1
  lrss 7
  sza
  jmp 2f
  lls 3
  xct class i
  isz class
  dac junk
```

0821 0822 0823

```
  cla
  llss 4
  add 1,llss
 dac 1f
  lac junk i
1:llss
  spa
2:isz class
  lac junk1
  jmp class i
```

0821
0822
0823

```
"    put symbuf symbol into table

table:
  lac equwrite
  dac equ
  jms between; add d0; add equmax
  jms halt
  add delta
  dac equwrite
  lac equ
  add equbot
  tad m1
  dac 8
  lac symwrite
  dac 8 i
  add symbot
  dac 2f
  lac mdelta
1:dzm 8 i
  tad d1
  spa
  jmp 1b
  lac sbbot
  jms move
2:0
  add o400000
  cma
add symbot
  cma
  add o400000
  and o17777
  dac symwrite
  jms between; add d0; add symmax
  jms halt
  jmp goon

"    find occurrence of symbuf symbol in equtab

prev:
  lac equ
  jmp find+1
find:
  lac equwrite
  dac 9f+t
  lac o777
  jms sbput
  lac sbbot
  dac 2f
1:lac 9f+t
  tad mdelta
  dac 9f+t
  spa
  jmp nuts
  add equbot
  dac junk
  lac junk i
  add symbot
  jms comp
2:0
  jmp 1b
  lac 9f+t
  dac equ
```

```
  jmp goon
t=t+1 "next equtab location to test

sbput:0
  lmq
  lac sbwrite
  add sbbot
  dac 1f
  lacq
  jms dchar
1:0
  lac sbwrite
  add o400000
  dac sbwrite
  jms cbetween; add d0; add sbmax
  jms halt
  jmp sbput i

getname:
  lac equ
  add equbot
  dac 9f+t
  lac 1f
  jms twoktab; lac 9f+t i

1:gf .+1 x
  lac ii
  dac 8
  lac 8 i
  add symbot
  and o17777
  jms obuild
  jmp ggoon
t=t+1   "equtable entry

" puts double word entries in ktab and gives
"pointer to first as result

twoktab:0
  jms kput
  lac 1.gk
  add k
dac nframe i
  isz nframe
  xct twoktab i
  jms kput
  jmp goon
```

```
char:
  lac j
  dac jsav
 isz ii
  jms ctest
 jmp backup
  jmp goon

string:
  isz ii
  jms ctest
  jmp goon
  jmp string+1

ctest:0
  jms jget
  jms class; add ii i
  jmp ctest i
  jms sbput
  lac j
  add o400000
  dac j
  isz ctest
  jmp ctest i

mark:
  jms jget
  dzm sbwrite
  jmp goon

parsedo:
  isz ii
  jms advance; jmp 3f
  jms advance; jmp 1f
  jms aget
  dac ii
  jmp rinterp

1:lac frame
  add refrsz
  dac ii
  sad nframe
  jmp retreat
  dac gflag
  lac gefrsz
  dac dffrmsz
  jms advance; jmp 2f
  jmp ginterp

2:lac refrsz
  dac dffrmsz
  add frame
  dac nframe
  dzm gflag
  jmp retreat

3:jms s0get; add d,k
  dac k
  jmp goon

bundle:
  jms bundlep
```

```
  dac 9f+t
  sna
  jmp goon
  jms nframe0
  dac nframe
  lac 9f+t
  dac nframe i
  isz nframe
  jmp goon
t=t+1

"  jms between; add a; add b; skip if a<=ac<b

between:0
  dac 9f+t
  cma
  xct between i
  isz between
  sma
  jmp 1f
  lac 9f+t
  cma
  xct between i
  isz between
  sma
1:isz between
  lac 9f+t
  jmp between i
t=t+0   "shared with next temporary


"  jms cbetween; add a; add b; skip if a<=ac<b where ac
"   contains a character address

cbetween:0
  dac 9f+t
  cma
  xct cbetween i
  isz cbetween
  ral
  sma
  jmp 1f
  lac 9f+t
  cma
  xct cbetween i
  isz cbetween
  ral
  sma
1:isz cbetween
  lac 9f+t
  jmp cbetween i
t=t+1   "ac contents
```

922 . 921

923

```
rerm:
  jms aget
  jmp 1f
rers:
  jms aget
  add frame
1:dac holdlv
  lac holdlv i
  jmp 2f
rerv:
  jms aget
  lls 6
  lrs 6
2:dac nframe i
  isz nframe
  jmp goon

rero:
  jms decnf
  dac rand1
  jms aget
  and o77
  add obranch
  dac 2f
  cma
  tad unary
  spa
  jmp 1f
  jms decnf
  dac rand2
1:lac rand1
2:xct
  jmp result

obranch:
  xct .+1
  opr         ;op=0400000+1
  jmp rorel   ;le=op;op=op+1
  jmp rorel   ;ne=op;op=op+1
  jmp rorel   ;lt=op;op=op+1
  jmp rorel   ;ge=op;op=op+1
  jmp rorel   ;eq=op;op=op+1
  jmp rorel   ;gt=op;op=op+1
  tad rand2   ;ad=op;op=op+1
  jms sub     ;sb=op;op=op+1
  and rand2   ;an=op;op=op+1
  jmp roor    ;or=op;op=op+1
  xor rand2   ;xo=op;op=op+1
  jmp rosr    ;sr=op;op=op+1
  jmp rosl    ;sl=op;op=op+1
  jmp romn    ;mn=op;op=op+1
  jmp romx    ;mx=op;op=op+1
  lac rand2   ;as=op;op=op+1

  opr         ;pl=op;op=op+1
  jmp romi    ;mi=op;op=op+1
  cma         ;cm=op;op=op+1
  jmp roindir;indir=op;op=op+1
  lac holdlv  ;addr=op;op=op+1
unary:xct obranch+1+pl

rorel:      "<= 001
```

16

0421. 0422 0423

```
  jms sub    "!= 010
  sna        "<  011
  jmp 2f     ">= 100
  spa        "=  101
  jmp 1f     ">  110
  lac d1    "a>b, code 001
  jmp 3f
1:lac d2    "a<b, code 100
2:add d2    "a=b, code 010
3:and ii i
  sza
  -1
  cma
  jmp result

sub:0
  cma
  tad rand2
  cma
  jmp sub i

roor:
  lmq
  lac rand2
  omq
  jmp result

rosr:
  lac rand2
  add l.lrs
  cll
  jmp 1f
rosl:
  lac rand2
  add l.lls
  clq
1:dac .+2
  lac rand1
  0
  jmp result

romn:
  jms sub
  cma
  jmp .+2
romx:
  jms sub
  ral
  lac rand1
  szl
  lac rand2
  jmp result

romi:
  cma
  tad d1
  jmp result

roindir:
  dac holdlv
  lac holdlv i
```

```
  jmp result

result:
  dac junk
  dac nframe i
  isz nframe
  lac ii i
  and stbit
  sna
  jmp exprtest
  lac junk
  dac holdlv i
  lac ii i
  and fibit
  sza
  jms decnf
  jmp goon

exprtest:
  lac ii i
  and fibit
  sna
  jmp goon
  jms decnf
  lac nframe i
  sza
  -1
  cma
  dac fflag
  jmp goon

decnf:0
  -1
  tad nframe
  dac nframe
  lac nframe i
  jmp decnf i
```

0421 0422 0423

```
9!,=.+t

end = -1

no = 0000000
rx = 0040000; gx = rx
rc = 0100000
rt = 0140000; gc = rt
rf = 0200000; gf = rf
rw = 0240000; gk = rw
ra = 0300000; gp = ra
rb = 0340000;gq = rb
ro = 0400000
rm = 0440000
rs = 0500000
rv = 0540000


ljmp:jmp
l,llss:llss
l,lrs:lrs
l,lls:lls
l,ra:ra
l,rb:rb
l,rw:rw
l,gk:gk
l,gcx:gc x

x = 020000
st = 0100
fi = 0200
opmask:0740000
exitmask: x

m1:0-1
d0:o0:0
d1:o1:1
d2:o2:2
d3:o3:3
d4:o4:4
d5:o5:5
d6:o6:6
d7:o7:7
d8:o10:8
asciisp:040
asciinl:012
nl:012777
onenl:nl
bugr:,+1;<rn>;<rx>;<rc>;<rt>;<rf>;<rw>;<ra>;<rb>
  <ro>;<rm>;<rs>;<rv>
bugg:,+1;<gn>;<gx>;<gz>;<gc>;<gf>;<gk>;<gp>;<gq>

o17:017
o60:060
o77:077
o777:0777

o417777:0417777
o400000:0400000
o740000:0740000
o377700:0377700
o17777:017777
```

+7

0821

0822

0823

```
o757777:0757777
o600600:0600600

junk:0
junk1:0

output:1
input:0

stbit:st
fibit:fi
holdlv:0
rand1:0
rand2:0


symwrite:0
symbot: symtab
symsiz = 500
symmax:symsiz

equwrite:0
equread:0
equ = equread
equbot: equtab
equsiz = 500
equmax: equsiz
delta: 2
mdelta:0-2

sbsiz=50
sbmax:sbsiz
sbwrite:0
sbbot:sbbuf

fflag:0
gflag:0
ignore:.+1;0400000;0;0;0;0;0;4
frame:rstack
nframe:rstack+6
env = ignore
d,ii = d2
d,env = d3
d,blkmod = d3
d,j = d4
d,k = d5
dffrmsz:6
framsiz:4
refrsz = d6
gefrsz = d4
ii: start
k:0

rsiz = 500
rmax: rsiz
rbot:rstack
rtop:rstack+rsiz

owrite:0
obot:obuf
osiz=64
```

821.
822.
823

```
" op.s is the op code defination file
" for the assembler
" no iot ops are defined
" system cal entries are defined
"
dac = 0040000
jms = 0100000
dzm = 0140000
lac = 0200000
xor = 0240000
add = 0300000
tad = 0340000
xct = 0400000
isz = 0440000
and = 0500000
sad = 0540000
jmp = 0600000
eae = 0640000
i = 020000
opr = 0740000
law = opr i
cma = opr 1
cml = opr 2
ral = opr 010
rar = opr 020
sma = opr 0100
sza = opr 0200
snl = opr 0400
skp = opr 01000
spa = opr 01100
sna = opr 01200
szl = opr 01400
cll = opr 04000
cla = opr 010000
las = opr 010004
lrs = eae 0500
lrss = i lrs
lls = eae 0600
llss = i lls
als = eae 0700
alss = i als
lacq = eae 01002
lacs = eae 01001
clq = eae 010000
omq = eae 2
cmq = eae 4
lmq = eae 012000

sys = i
save = 1
open = 3
read = 4
write = 5
creat = 6
seek = 7
close = 9
exit = 14
```