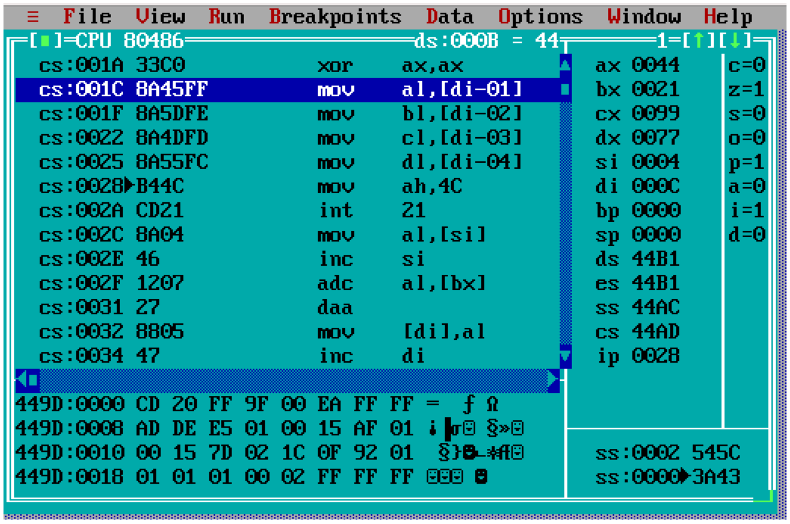
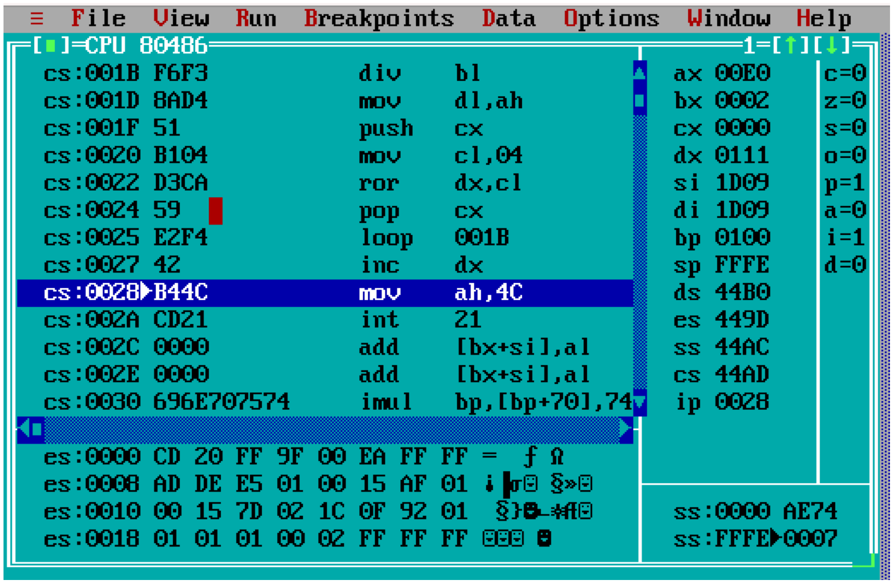
```
C:\>edit 11_1.asm

C:\>tasm 11_1.asm /z
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   11_1.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  491k


C:\>tlink 11_1 /v
Turbo Link  Version 4.01 Copyright (c) 1991 Borland International
Warning: No stack

C:\>11_1
02D0
C:\>_

C:\>edit 11_2.asm

C:\>tasm 11_2.ASM /z
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   11_2.ASM
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  490k


C:\>tlink 11_2 /v
Turbo Link  Version 4.01 Copyright (c) 1991 Borland International
Warning: No stack

C:\>td 11_2
```



```
 ≡  File  View  Run  Breakpoints  Data  Options  Window  Help
[■]=CPU 80486=================ds:000B = 44========1=[↑][↓]=
  cs:001A 33C0           xor    ax,ax           ax 0044   c=0
  cs:001C 8A45FF         mov    al,[di-01]      bx 0021   z=1
  cs:001F 8A5DFE         mov    bl,[di-02]      cx 0099   s=0
  cs:0022 8A4DFD         mov    cl,[di-03]      dx 0077   o=0
  cs:0025 8A55FC         mov    dl,[di-04]      si 0004   p=1
  cs:0028►B44C           mov    ah,4C           di 000C   a=0
  cs:002A CD21           int    21              bp 0000   i=1
  cs:002C 8A04           mov    al,[si]         sp 0000   d=0
  cs:002E 46             inc    si              ds 44B1
  cs:002F 1207           adc    al,[bx]         es 44B1
  cs:0031 27             daa                    ss 44AC
  cs:0032 8805           mov    [di],al         cs 44AD
  cs:0034 47             inc    di              ip 0028

449D:0000 CD 20 FF 9F 00 EA FF FF =  ƒ Ω
449D:0008 AD DE E5 01 00 15 AF 01 ¡▌σ☺ §»☺        ss:0002 545C
449D:0010 00 15 7D 02 1C 0F 92 01  §}☻∟☼Æ☺        ss:0000►3A43
449D:0018 01 01 01 00 02 FF FF FF ☺☺☺ ☻
```

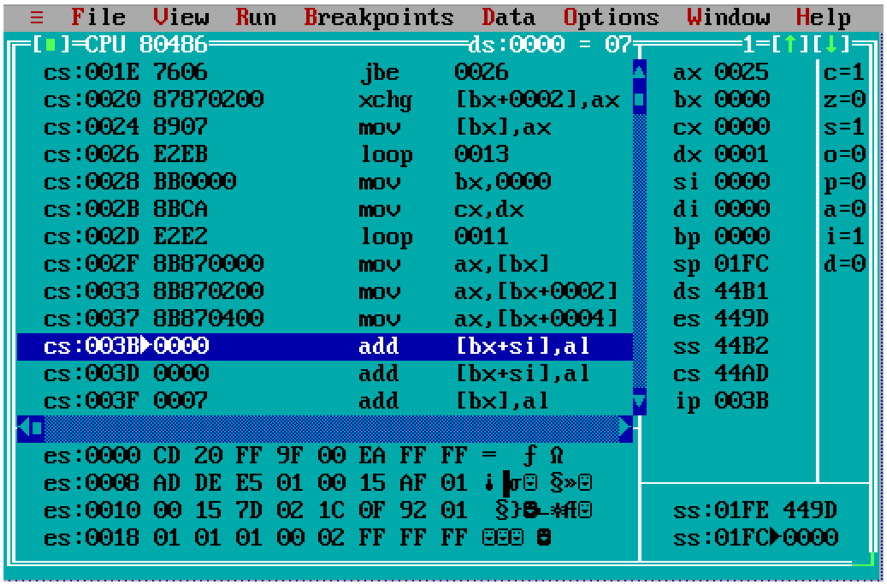
```
C:\>edit 11_3.asm

C:\>tasm 11_3.asm /z
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   11_3.asm
in:      div      bl
*Warning* 11_3.asm(15) Reserved word used as symbol: IN
Error messages:    None
Warning messages:  1
Passes:            1
Remaining memory:  491k


C:\>tlink 11_3 /v
Turbo Link  Version 4.01 Copyright (c) 1991 Borland International
Warning: No stack

C:\>td 11_3_
```


≡ File View Run Breakpoints Data Options Window Help
[■]=CPU 80486
1=[↑][↓]

```
  cs:001B F6F3              div    bl             ax 00E0   c=0
  cs:001D 8AD4              mov    dl,ah          bx 0002   z=0
  cs:001F 51                push   cx             cx 0000   s=0
  cs:0020 B104              mov    cl,04          dx 0111   o=0
  cs:0022 D3CA              ror    dx,cl          si 1D09   p=1
  cs:0024 59                pop    cx             di 1D09   a=0
  cs:0025 E2F4              loop   001B           bp 0100   i=1
  cs:0027 42                inc    dx             sp FFFE   d=0
  cs:0028▶B44C              mov    ah,4C          ds 44B0
  cs:002A CD21              int    21             es 449D
  cs:002C 0000              add    [bx+si],al     ss 44AC
  cs:002E 0000              add    [bx+si],al     cs 44AD
  cs:0030 696E707574        imul   bp,[bp+70],74  ip 0028

  es:0000 CD 20 FF 9F 00 EA FF FF =  ƒ Ω
  es:0008 AD DE E5 01 00 15 AF 01 ↓ σ☺ §»☺
  es:0010 00 15 7D 02 1C 0F 92 01  §}☻∟☼Æ☺        ss:0000 AE74
  es:0018 01 01 01 00 02 FF FF FF ☺☺☺ ☻           ss:FFFE▶0007
```

```
C:\>edit 11_4.asm

C:\>tasm 11_4.asm /z
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   11_4.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  490k
```

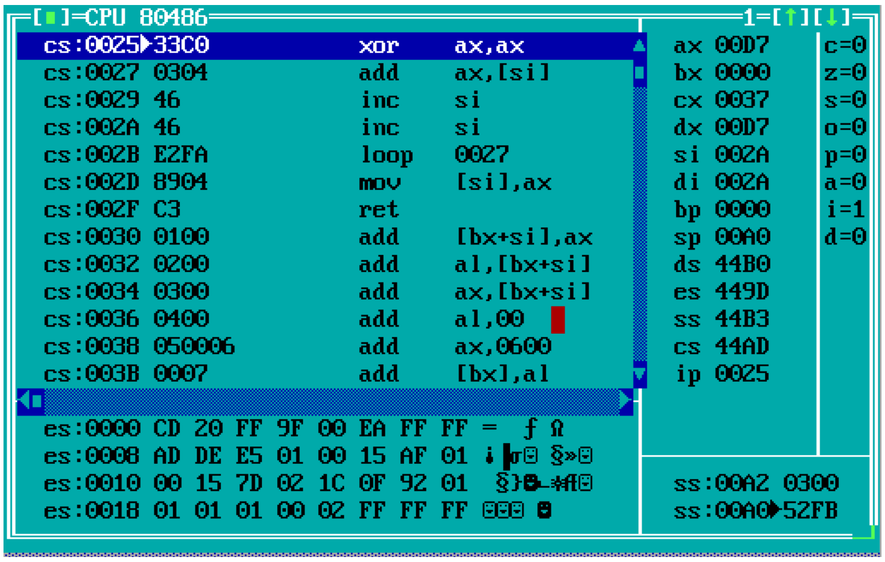
```
C:\>tlink 11_4 /v
Turbo Link  Version 4.01 Copyright (c) 1991 Borland International

C:\>11_4
```



```
[■]=CPU 80486=====================ds:0000 = 07==========1=[↑][↓]=
 cs:001E 7606           jbe    0026            ax 0025    c=1
 cs:0020 87870200       xchg   [bx+0002],ax    bx 0000    z=0
 cs:0024 8907           mov    [bx],ax         cx 0000    s=1
 cs:0026 E2EB           loop   0013            dx 0001    o=0
 cs:0028 BB0000         mov    bx,0000         si 0000    p=0
 cs:002B 8BCA           mov    cx,dx           di 0000    a=0
 cs:002D E2E2           loop   0011            bp 0000    i=1
 cs:002F 8B870000       mov    ax,[bx]         sp 01FC    d=0
 cs:0033 8B870200       mov    ax,[bx+0002]    ds 44B1
 cs:0037 8B870400       mov    ax,[bx+0004]    es 449D
 cs:003B►0000           add    [bx+si],al      ss 44B2
 cs:003D 0000           add    [bx+si],al      cs 44AD
 cs:003F 0007           add    [bx],al         ip 003B

 es:0000 CD 20 FF 9F 00 EA FF FF = ƒ Ω
 es:0008 AD DE E5 01 00 15 AF 01 ¡ σ☺ §»☺
 es:0010 00 15 7D 02 1C 0F 92 01  §}☻∟☼Æ☺   ss:01FE 449D
 es:0018 01 01 01 00 02 FF FF FF ☺☺☺ ☻      ss:01FC►0000
```

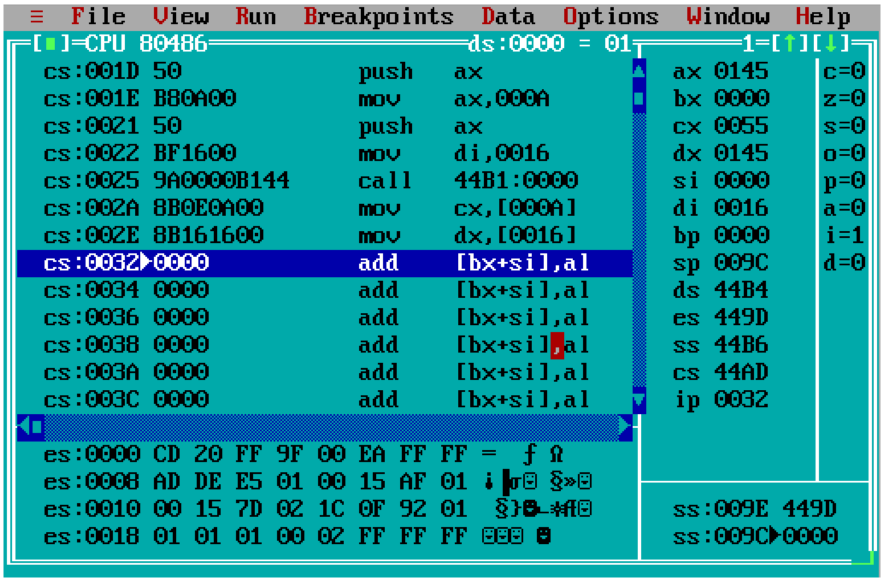
```
C:\>tasm 11_5.asm /z
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   11_5.asm
desg     ends
*Warning* 11_5.asm(29) Unmatched ENDS: DESG
Error messages:    None
Warning messages:  1
Passes:            1
Remaining memory:  490k


C:\>tlink 11_5 /v
Turbo Link  Version 4.01 Copyright (c) 1991 Borland International
Warning: No stack

C:\>11_5
312564654321
C:\>
```

```
C:\>edit 12_1.asm

C:\>tasm 12_1.asm /z
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   12_1.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  490k


C:\>tlink 12_1 /v  z
Turbo Link  Version 4.01 Copyright (c) 1991 Borland International
```



```
[■]=CPU 80486                                        1=[↑][↓]
  cs:0025►33C0           xor    ax,ax         ax 00D7     c=0
  cs:0027 0304           add    ax,[si]       bx 0000     z=0
  cs:0029 46             inc    si            cx 0037     s=0
  cs:002A 46             inc    si            dx 00D7     o=0
  cs:002B E2FA           loop   0027          si 002A     p=0
  cs:002D 8904           mov    [si],ax       di 002A     a=0
  cs:002F C3             ret                  bp 0000     i=1
  cs:0030 0100           add    [bx+si],ax    sp 00A0     d=0
  cs:0032 0200           add    al,[bx+si]    ds 44B0
  cs:0034 0300           add    ax,[bx+si]    es 449D
  cs:0036 0400           add    al,00         ss 44B3
  cs:0038 050006         add    ax,0600       cs 44AD
  cs:003B 0007           add    [bx],al       ip 0025

  es:0000 CD 20 FF 9F 00 EA FF FF =  ƒ Ω
  es:0008 AD DE E5 01 00 15 AF 01 ¡ σ☺ §»☺
  es:0010 00 15 7D 02 1C 0F 92 01  §}☻∟☼Æ☺      ss:00A2 0300
  es:0018 01 01 01 00 02 FF FF FF ☺☺☺ ☻         ss:00A0►52FB
```

```
C:\>edit 12_2.asm

C:\>tasm 12_2.asm /z
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   12_2.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  489k


C:\>tlink 12_2 /v
Turbo Link  Version 4.01 Copyright (c) 1991 Borland International
```



```
[■]=CPU 80486==========================ds:0000 = 01=========1=[↑][↓]=
  cs:001D 50              push   ax              ax 0145   c=0
  cs:001E B80A00          mov    ax,000A         bx 0000   z=0
  cs:0021 50              push   ax              cx 0055   s=0
  cs:0022 BF1600          mov    di,0016         dx 0145   o=0
  cs:0025 9A0000B144      call   44B1:0000       si 0000   p=0
  cs:002A 8B0E0A00        mov    cx,[000A]       di 0016   a=0
  cs:002E 8B161600        mov    dx,[0016]       bp 0000   i=1
  cs:0032▸0000            add    [bx+si],al      sp 009C   d=0
  cs:0034 0000            add    [bx+si],al      ds 44B4
  cs:0036 0000            add    [bx+si],al      es 449D
  cs:0038 0000            add    [bx+si],al      ss 44B6
  cs:003A 0000            add    [bx+si],al      cs 44AD
  cs:003C 0000            add    [bx+si],al      ip 0032

  es:0000 CD 20 FF 9F 00 EA FF FF =  ƒ Ω
  es:0008 AD DE E5 01 00 15 AF 01 ↓ σ☺ §»☺
  es:0010 00 15 7D 02 1C 0F 92 01  §}☻∟☼Æ☺        ss:009E 449D
  es:0018 01 01 01 00 02 FF FF FF ☺☺☺ ☻           ss:009C▸0000
```

实验源代码

```
codseg    segment
          assume    cs:codseg
main:     mov       dx,0
          mov       ax,1
          mov       bx,0
          mov       cx,6
loop1:    inc       bx
          mul       bx
          loop      loop1
          push      ax
          mov       al,ah
          call      disprg
          pop       ax
          call      disprg
          mov       ah,4ch
          int       21h
disprg    proc
          push      cx
          push      dx
```

```
        push    ax
        and     al,0f0h
        mov     cl,4
        shr     al,cl
        add     al,30h
        cmp     al,3ah
        jb      ds1
        add     al,07
ds1:    mov     dl,al
        mov     ah,02h
        int     21h
        pop     ax
        and     al,0fh
        add     al,30h
        cmp     al,3ah
        jb      ds2
        add     al,07
ds2:    mov     dl,al
        mov     ah,02h
        int     21h
        pop     dx
        pop     cx
        ret
disprg  endp
codseg  ends
        end
```

```
code    segment
main    proc    far
        assume  cs:code,ds:data,es:data
start:  mov     ax,data
        mov     ds,ax
        mov     es,ax
        mov     si,offset one
        mov     bx,offset two
        mov     di,offset sum
        cld
        clc
        mov     cx,4
LL:     call    abc
        loop    LL
        xor     ax,ax
        mov     al,[di-1]
        mov     bl,[di-2]
        mov     cl,[di-3]
        mov     dl,[di-4]
        mov     ah,4ch
        int     21h
main    endp
abc     proc
        mov     al,[si]
        inc     si
L1:     adc     al,[bx]
        daa
        mov     [di],al
        inc     di
        inc     bx
        ret
abc     endp
code    ends
data    segment
one     db      22h,33h,44h,55h
two     db      55h,66h,77h,88h
sum     db      20      dup(?)
data    ends
        end
```

```
codseg  segment
        assume  cs:codseg,ds:datseg
        mov     ax,datseg
        mov     ds,ax
        lea     dx,prompt
        mov     ah,09
        int     21h
        mov     ah,01h
        int     21h
        and     al,0fh
        mov     ah,0
        push    ax
        mov     cx,4
        mov     bx,2
in:     div     bl
        mov     dl,ah
        push    cx
        mov     cl,4
        ror     dx,cl
        pop     cx
        loop    in
        inc     dx
        mov     ah,4ch
        int     21h
codseg  ends
datseg  segment
prompt  db      "input a number: $"
datseg  ends
        end
```

```
code    segment
        assume  cs:code,ds:data,ss:sstack
main    proc    far
start:  push    ds
        sub     ax,ax
        push    ax
        mov     ax,data
        mov     ds,ax
        mov     bx,0
        mov     cx,buf[bx]
        dec     cx
L1:     mov     dx,cx
L2:     add     bx,2
        mov     ax,buf[bx]
        cmp     ax,buf[bx+2]
        jbe     cont1
        xchg    ax,buf[bx+2]
```

```
          mov       [bx],ax
cont1:    loop      L2
          mov       bx,0
          mov       cx,dx
          loop      L1
          mov       ax,buf[bx]
          mov       ax,buf[bx+2]
          mov       ax,buf[bx+4]
main      endp
code      ends
data      segment
buf       dw        7,15,37,8600,0A768H,3412H,1256H,76H
data      ends
sstack    segment stack 'stack'
sa        dw        100h      dup(?)
sstack    ends
          end
```

```
cseg      segment
          assume    cs:cseg,ds:dseg
main:     mov       ax,dseg
          mov       ds,ax
          lea       dx,ary
          mov       ah,9
          int       21h
          mov       si,0
          mov       cx,5
loop1:    mov       dx,cx
loop2:    mov       al,ary[si]
          cmp       al,ary[si+1]
          jg        cont1
          xchg      al,ary[si+1]
          mov       ary[si],al
cont1:    add       si,1
          loop      loop2
          mov       si,0
          mov       cx,dx
          loop      loop1
```

```
        lea     dx,ary
        mov      ah,9
        int     21h
        mov      ah,4ch
        int     21h
cseg     ends
dseg     segment
ary     db        "3","1","2","5","6","4","$"
desg     ends
        end
```

```
code     segment
        assume   cs:code,ds:data,ss:sstack
main     proc     far
start:  mov     ax,data
        mov      ds,ax
        lea     si,ary1
        lea     di,sum1
        mov      cx,0ah
        call    sum
        lea     si,ary2
        lea     di,sum2
        mov      cx,0ah
        call    sum
        mov      cx,sum1
        mov      dx,sum2
main     endp
sum      proc
        xor     ax,ax
l1:      add    ax,[si]
        inc     si
        inc     si
        loop    l1
        mov      [si],ax
        ret
sum      endp
code     ends
data    segment
```

```
ary1      dw        1,2,3,4,5,6,7,8,9,0ah
sum1        dw        ?
ary2      dw        11h,12h,13h,14h,15h,16h,17h,18h,19h,1ah
sum2        dw        ?
data      ends
sstack    segment stack 'stack'
sa         dw        50h       dup(?)
sstack    ends
          end
```

```
mcode     segment
          assume    cs:mcode,ds:mdata,ss:mstack
main      proc far
start:  push      ds
          mov       ax,0
          push      ax
          mov       ax,mdata
          mov       ds,ax
          mov       ax,offset ary1
          push      ax
          mov       ax,0ah
          push      ax
          lea       di,sum1
          call      far       ptr       padd
          mov       ax,offset ary2
          push      ax
          mov       ax,0ah
          push      ax
          lea       di,sum2
          call      far       ptr       padd
          mov       cx,sum1
          mov       dx,sum2
main      endp
mcode     ends
pcode     segment
          assume    cs:pcode,ds:mdata,ss:mstack
padd      proc      far
          push      bx
```

```
            push      cx
            push      bp
            mov       bp,sp
            mov       cx,[bp+10]
            mov       bx,[bp+12]
            mov       ax,0
next:      add       al,[bx]
            daa
            mov       dl,al
            mov       al,0
            adc       al,ah
            daa
            mov       ah,al
            mov       al,dl
            inc       bx
            loop      next
            mov       [di],ax
            pop       bp
            pop       cx
            pop       bx
            ret       4
padd      endp
pcode     ends
mdata     segment
ary1      db        1,2,3,4,5,6,7,8,9,10h
sum1      dw        ?
ary2      db        11h,12h,13h,14h,15h,16h,17h,18h,19h,10h
sum2      dw        ?
mdata     ends
mstack    segment stack      'stack'
sb        dw        50h       dup       (?)
mstack    ends
            end
```