```
Stream Compiler Steering
 *** Compilers 12/5/78

    get 'GLOBALS'
    get 'PRIVATE GLOBALS'
    get 'LIBRARY GLOBALS'

  5 get 'Common Decls'


    static
    §s
 10     WF            = 0
        InputStream   = 0
        OutputStream  = 0
        Compiled      = false
        WorstError    = NOERROR
 15 $s

    static  || initialising globals
    §s
        Mode          = 6
 20     NameSize      = 1000
        NumberSize    = 500
        TextSize      = 7500
        StringSize    = 200
        StructureSize = 12000
 25     AppSize       = 300
        CodeSize      = 6000
        ReportSize    = 30
        GlobSize      = 50
    $s
 30
    manifest
    §m
        AddtoCUC[Rt] is
        §   let c = NewVec[2]
 35         c↓2 := Rt
            EnterinCUChain[c]
        §

        LoadF[S] is LoadFile[LookUp[S, 'IC', SCI]]
 40
        WorseErrorfoundthan[Type] = (WorstError > Type)
    $m


 45 let Prog[] be
    §P
        SetUpCompiler[]
        Run[Compiler]
        RunReports[]
 50     ReportS['*nCompilation *S', Compiled → 'succeeds', 'fails']
        if WorstError = COMPERROR do Run[PM]
    $P


 55 and Compiler[] be
    §C
        let c = CPtr
        LoadSystemFile['BitMap Rts']
        DoPass0[]
 60     if WorseErrorfoundthan[HARD] return
        DoPass1[]
```

```
        Unload[c]
        DoPass2[]
        DoPass3[]
65      if WorseErrorfoundthan[SOFT] return
        DoPass4[]
        unless WorseErrorfoundthan[SOFT] do Compiled := true
    $C

70
    and DoPass0[] be
    §P0
        let c = CPtr
        LoadF['PP']
75      Output := OuttoBFile[WF]

        PP[]

        Unload[c]
80  §P0


    and DoPass1[] be
    §P1
85      let c = CPtr
        LoadF['Pass 1']
        In := InfromBFile[WF]
        Output := OuttoBFile[WF]

90      Pass1[]

        Close[In]
        Close[Output]
        Unload[c]
95  §P1


    and DoPass2[] be
    §P2
100     let c = CPtr
        LoadF['Pass 2']
        In := RevInfromBFile[WF]
        Output := OuttoBFile[WF]

105     Pass2[]

        Close[In]
        Close[Output]
        Unload[c]
110 §P2


    and DoPass3[] be
    §P3
115     let c = CPtr
        LoadF['Pass 3a']
        LoadF['Pass 3b']
        In := RevInfromBFile[WF]
        Output := OuttoBFile[WF]
120
        Pass3[]

        Close[In]
        Close[Output]
125     Unload[c]
    §P3
```

```
    and DoPass4[] be
130 §P4
        let c = CPtr
        LoadF['Pass 4']
        In := RevInfromBFile[WF]
        Output := OutputStream
135
        Pass4[]

        Close[In]
        Close[Output]
140     Unload[c]
    $P4


    and RunReports[] be
145 §RR
        Close[ErrorStream]
        LoadSystemFile['Quicksort']
        LoadF['Print Reports']
        In := InputStream
150     Reset[In]
        Run[PrintReports]
    $RR


155 and SetUpCompiler[] be
    §SC
        SCI := LookUp['Compiler', 'Index', SystemIndex]
        WF := WorkFile['WorkFile']
        ErrorStream := OuttoFile[WorkFile['Error Reports']]
160     GlobList := NewVec[GlobSize × 2 + 1] + 1
        GlobList↓(-1) := 0
        GlobList↓0 := 0
        InputStream := In
        OutputStream := Output
165     Peep := NullProgram
        LoadSystemFile['Bilateral File Streams']
        Out[ReportStream, `*n`]
        AddtoCUC[ClearUpCompiler]
    $SC
170

    and ClearUpCompiler[] be
    §CC
        DeleteBody[WF]
175     DeleteBody[WorkFile['Error Reports']]
        DeleteBody[WorkFile['Names']]
        DeleteBody[WorkFile['Strings']]
        DeleteBody[WorkFile['Numbers']]
        DeleteBody[WorkFile['Gets']]
180 $CC


    and Error[n, Type, Char, Routine, P] be
    §E
185     let RepMax = (Type = SOFT → ReportSize/2, ReportSize)
        unless Char = ERROR ∨ ReportNo ≥ RepMax do
        §   ReportNo := ReportNo + 1
            if ReportNo = RepMax do
                test Type = SOFT
190               ifso  n := 11
                  ifnot n, Routine := 1, Finish
```

```
            Out[ErrorStream, PassNo]
            Out[ErrorStream, GetNo]
            Out[ErrorStream, LineNo]
195         Out[ErrorStream, n]
            Out[ErrorStream, Char]
        $
        unless WorseErrorfoundthan[Type] do WorstError := Type
        Routine[P]
200 $E


    and CompilerError[n] be
    §CE
205     ReportS['Compiler error *N', n]
        for i = 1 to 20 do
        §   OutS[ReportStream, '*O*C', Ch, i rem 10 = 0 → '*n', '*s']
            if Endof[In] do
            §   ReportS['End of Input']
210             break
            $
            Ch := Next[In]
        $
        Dump[]
215     WorstError := COMPERROR
        Finish[]
    §CE

    and PM[] be
220 §   Prog := DefaultProg
        LoadSystemFile['DumpPM']
        Run[Prog]
    $

****
```