```
 *** Compilers 15/2/77


    let PP[] be
    §P
        let c = NewVec[2]
 5      c↓2 := ClosePP
        PassNo := 0
        LineNo := 1
        GetNo := 0
        ReportNo := 0
10      Send := NormalSend
        DeclIndex := LookUp['Decl', 'Index', SystemIndex]
        SetUpTables[]
        NullNameNo := InsertNullName[]
        Peep[1]
15      EnterinCUChain[c]
        NextCh[] repeatwhile Ch = '*n'
        PreProc[] repeatuntil EndofPP
        Write[ENDPROG]
        Send := NullProgram
20      until ProcVec↓ENDCH = P_End_of_Input do DenestGets[]
        Close[Output]
        ClosePP[]
        RemovefromCUChain[c]
        ReturnVec[c, 2]
25      ReturnTables[]
        Peep[3]
    §P


30  and SetUpTables[] be
    §SUT
        NameMap := BitMap[NameSize + 26]
        for i = 1 to 26 do EnterintoMap[NameMap, i]

35      NameTable := NewTable[NameSize, 51]
        NameTableHash := NewTable[NameSize × 2, 0]
        NameTableLink := NewTable[NameSize, 0]
        NumTable := NewTable[NumberSize, 52]
        StringTable := NewTable[StringSize, 53]
40
        SetUpProcVec[]
        GetList := NewVec[GETLISTSIZE]
        GetList↓0 := 0
        TextVec := NewVec[TextSize]
45      Stack := TextVec
        ResetStack[]
    §SUT


50  and ReturnTables[] be
    §RT
        ReturnVec[TextVec, TextSize]
        ReturnVec[GetList, GETLISTSIZE]
        ReturnVec[ProcVec, MAXCH]
55
        ReturnTable[NameTable]
        ReturnTable[NameTableHash]
        ReturnTable[NameTableLink]
        ReturnTable[NumTable]
60      ReturnTable[StringTable]
    §RT
```

```
   and PrintUsage[] be
65     unless (Mode ∧ 1) = 0 do
       §PU
           ReportTable['name', NameTable]
           ReportTable['number', NumTable]
           ReportTable['string', StringTable]
70         ReportS['*nText size = *N', Stack - TextVec]
       §PU


   and ReportTable[String, Table] be
75 §RT
       let n = Table↓PTR - OFFSET
       OutS[ReportStream, '*N *S*C ', n, String, n = 1 → NULL, `s`]
   §RT

80
   and ClosePP[] be
   §   PrintUsage[]
       SendTables[]
   §
85

   and TermPP[] be
   §TPP
       until ProcVec↓ENDCH = P_End_of_Input do DenestGets[]
90     Finish[]
   §TPP


   and NewTable[n, x] = valof
95 §NT
       let v = NewVec[n + OFFSET]
       v↓SIZE := n + OFFSET
       v↓PTR  := FIRSTEL - 1
       v↓REP := x
100    v↓FIRSTEL := NULL
       resultis v
   §NT


105 and ReturnTable[t] be
       ReturnVec[t, t↓SIZE]


   and SetUpProcVec[] be
110 §SUPV
       ProcVec := NewVec[MAXCH]
       for i = 0 to MAXCH do ProcVec↓i := P_Illegal_Ch
       FillProcVec[]
       ProcVec↓ENDCH := P_End_of_Input
115 §SUPV


   and P_Illegal_Ch[] be
   §I
120    Error[55, HARD, Ch, NullProgram]
       Send[ERROR, 0]
       NextCh[]
   §I

125
   and Delete[Cat] be
```

```
        while Is[Ch, Cat] do NextCh[]


130 and P_End_of_Input[] be EndofPP := true


    and Adjoin_to[Table] = valof
    §AD
135     let n = Table↓PTR + 1
        if n > Table↓SIZE do
            Error[Table↓REP, HARD, DUMMY, TermPP]
        Table↓n := Top[]
        Table↓PTR := n
140     ResetStack[]
        resultis n - OFFSET
    §AD


145 and AddVec_to[Table] = valof
    §AV
        let n = Table↓PTR + 1
        if n > Table↓SIZE do
            Error[Table↓REP, HARD, DUMMY, TermPP]
150     Table↓n := Stack
        Table↓PTR := n
        StepOnStack[]
        resultis n - OFFSET
    §AV
155

    and StepOnStack[] be
    §SOS
        Stack := Stack + StackPtr/2 + 1
160     if Stack + (MAXLENGTH + 2) > TextVec + TextSize do
            Error[60, HARD, DUMMY, TermPP]
        ResetStack[]
    §SOS

165
    and InsertNullName[] = valof
    §INN
        ResetStack[]
        resultis InsertName[]
170 §INN


    and InsertName[] = Insert_in[NameTable,NameTableHash,NameTableLink] + 26

175
    and Insert_in[Table, Hash, Link] = valof
    §IN
        let h = StackHash[]
        let n = Find[h, Hash]
180     PackStack[]
        test n = NULL
          ifso
            §   n := AddVec_to[Table] + OFFSET
                AddHash[n, h, Hash]
185             Link↓n := NULL
                resultis n - OFFSET
            §
          ifnot
            §r
190             test EqS[Table↓n, Stack]
                  ifso
```

```
                    §   if Table = NameTable do
                            EnterintoMap[NameMap, n + 26 - OFFSET]
                        ResetStack[]
195                     resultis n - OFFSET
                    §
                  ifnot test Link↓n = NULL
                          ifso
                            §   let p = AddVec_to[Table] + OFFSET
200                             Link↓n := p
                                Link↓p := NULL
                                resultis p - OFFSET
                            §
                          ifnot n := Link↓n
205         $r repeat
    §IN


    and AddHash[n, h, Hash] be
210 §AH
        let p = Hash↓PTR + 2
        Hash↓PTR := p
        Hash↓(p-1) := n
        Hash↓p :=h
215     Hash↓(p+1) := NULL
    $AH


    and Wrong[String]  = valof
220 §CI
        let n = ((String↓0) rshift 8)/2
        for i = 0 to n do
            test Ch = (8377 ∧ String↓i)
            ifso NextCh[]
225         ifnot resultis true
        resultis false
    $CI

****
```