Program B (*Multiplication of permutations in cycle form*)

```
001 MAXWDS    IS      1200                          Maximal size for table T
002 t         IS      $255
003 tt        GREG    0
004 _lpren    GREG    #20202028                     "␣␣␣("
005 _rpren    GREG    #20202029                     "␣␣␣)"
006 _nlnull   GREG    #0a000000                     Newline and a zero byte
007 ip        IS      $2                            Pointer for input permutation
008 op        IS      $3                            Pointer for output permutation
009 p         IS      $4                            A symbol of the permutation
010 size      IS      $5                            The length of the input permutation
011 x         IS      $6                            An element of the names table
012 z         IS      $7                            The variables of the algorithm.
013 i         IS      $8
014 j         IS      $9
015 n         IS      $10                           The number of different elements
016           LOC     Data_Segment
017           GREG    @
018 NoArg     BYTE    "Missing argument: file with input permutation expected",#a,0
019 NoFile    BYTE    "Can't open the file given in first argument.",#a,0
020 BUFSIZE   IS      80+1+1                        80 Bytes plus newline can be read
021 INP       IS      3                             Handle for input file
022 ArgIn     OCTA    0,TextRead                    First octabyte is later filled with argument
023 ArgRead   OCTA    0,BUFSIZE                     Ditto
024 X         GREG    @                             Location to store the different elements
025           LOC     @+MAXWDS
026 T         GREG    @                             Location to store table T
027           LOC     @+MAXWDS
028 Perm      GREG    @                             Location to store the permutations
029           LOC     #100
030 Error1    LDA     t,NoArg
031           JMP     PrtAns
032 Error2    LDA     t,NoFile
033           JMP     PrtAns
034 Main      SET     tt,Perm
035           LDO     t,$1,8
036           BZ      t,Error1                      No argument: error
037           STO     t,ArgIn                       Otherwise use the argument.
038 0H        LDA     t,ArgIn                       Open input file.
039           TRAP    0,Fopen,INP
040           BN      t,Error2                      −1 indicates an error.
041 ReadLine  STO     tt,ArgRead                    Read the input.
042           LDA     t,ArgRead
043           TRAP    0,Fgets,INP
044           BN      t,EndRead
045           ADD     tt,tt,t
046           SUB     t,tt,t                        Output the input line.
047           TRAP    0,Fputs,StdOut
048           SUB     tt,tt,1                       Remove the newline byte.
049           JMP     ReadLine
050 EndRead   TRAP    0,Fclose,INP                  Close the input file.
051           SUB     op,tt,Perm            1       Start output after the equal sign.
052           SET     size,op               1       Remember the length of the permutation.
```

```
053             SUB     ip,size,8               1   ip ← last symbol of permutation.
054             SET     n,4                     1   n ← 1, ready to store an element.
055  Right      SET     z,0                     A   Prepare for step B2, set Z ← 0.
056  B2         LDT     p,Perm,ip               B   B2. Next element.
057             SUB     ip,ip,4                 B
058             CMP     t,p,_rpren              B   Is it a right parenthesis?
059             BZ      t,Right                 B
060             CMP     t,p,_lpren              D   Is it a left parenthesis?
061             BZ      t,B4                    D
062  Search_p   SET     i,n                     E   Prepare to search names table.
063  1H         SUB     i,i,4                   F
064             BNP     i,New_x                 F   Branch if end of table is reached.
065             LDT     x,X,i                   G   Get a known x_i.
066             CMP     t,p,x                   G   Is it a match?
067             PBNZ    t,1B                    G   No, continue search.
068  B3         LDT     p,T,i                   J   B3. Change T[i].
069             SET     t,z                     J   Save the value of Z.
070             STT     z,T,i                   J   Exchange Z with T[i].
071             SET     z,p                     J
072             PBNZ    t,B2                    J   Test former value of Z.
073             SET     j,i                     K   Z was zero, so j ← i.
074             JMP     B2                      K
075  New_x      STT     p,X,n                   H   Insert a new element in names table.
076             STT     n,T,n                   H   Initialize T[n].
077             SET     i,n                     H   i ← n.
078             ADD     n,n,4                   H   n ← n + 1.
079             JMP     B3                      H   Now continue with step B3.
080  B4         STT     z,T,j                   L   B4. Change T[j].
081             PBP     ip,B2                   L
082  Output     SET     i,n                     1   Output the permutation in cycle form.
083  0H         STT     _lpren,Perm,op          P   Start with a cycle.
084             ADD     op,op,4                 P
085  1H         SUB     i,i,4                   Q
086             BNP     i,Done                  Q   Branch if table of names is processed.
087             LDT     x,X,i                   T   Get an element name.
088             BN      x,1B                    T   Branch if it already tagged.
089  2H         STT     x,Perm,op               R   Otherwise output the element.
090             ADD     op,op,4                 R
091             NEG     x,0,x                   R   Tag element
092             STT     x,X,i                   R     and store it.
093             LDT     i,T,i                   R   Load the successor.
094             LDT     x,X,i                   R   Load the name of that element.
095             PBP     x,2B                    R   Branch if name is not tagged.
096             STT     _rpren,Perm,op          S   Otherwise close cycle.
097             ADD     op,op,4                 S
098             SUB     tt,op,3*4               S   Check for singleton cycle.
099             LDT     p,Perm,tt               S
100             CMP     t,p,_lpren              S   Appears a '(' two tetras earlier?
101             CSZ     op,t,tt                 S   Reset op if yes.
102             JMP     0B                      S
103  Done       LDA     t,Perm,size                 Start output after the equal sign.
104             SUB     op,op,4
105             CMP     tt,op,size                  Test if output is empty.
```

```
106             BNZ     tt,1F
107             STT     _lpren,t,0              Yes, so output the identity permutation.
108             STT     _rpren,t,4
109             ADD     op,size,8
110  1H         STT     _nlnull,Perm,op         Add newline and a null byte to output string.
111  PrtAns     TRAP    0,Fputs,StdOut
112             TRAP    0,Halt,0                ∎
```

Analysis

This algorithm uses the same input and output conventions as Algorithm A. The lines 31–51 are identical to the lines 25–45 of Algorithm A and the lines 104–113 are identical to lines 103–112 of Algorithm A.

The algorithm needs without input and output $(B+G+2H+2J+L+P+4R+2S+T)\mu+(9+A+6B+E+2F+3G+5H+7J+4K+4L+4Q+7R+9S+2T)\upsilon$.

This time Kirchhoff's law gives the following equations

$$\begin{aligned} A &= 1+B-D; & J &= H+G-F+E=E=D-L;\\ E &= D-L; & P &= 1+S;\\ G &= F-H; & T &= Q-1. \end{aligned}$$

Applying the equations to the used mems and oops several variables are eliminated: $(B+2D+F+H-L+Q+4R+3S)\mu+(8+7B+7D+5F+2H+4K-4L+6Q+7R+9S)\upsilon$. And most of the other values are already known:

$$\begin{aligned} B &= \text{number of words of input;}\\ D &= \text{number of words of input without right parentheses;}\\ H &= \text{number of distinct elements in input;}\\ K=L &= \text{number of cycles in input;}\\ Q-1=R &= \text{number of distinct elements in output;}\\ S &= \text{number of cycles in output (inlusive singletons).} \end{aligned}$$

Of course $H=R$; only the variable F cannot be easily resolved.

With the variables of Eq. (19) the profile of the algorithm is $(3Y+6N-3M+3U+F+1)\mu+(14Y+15N-7M+9U+5F+14)\upsilon$.

Running the program with Eq. (6) as input the `MMIX`-simulator shows at the end: `838 instructions, 204 mems, 964 oops; 201 good guesses, 45 bad`. With this input it follows that $Y=29$, $N=7$, $M=5$, and $U=3$. The algorithm itself needs `801 instructions, 198 mems, 887 oops; 197 good guesses, 43 bad`. The value of F is 74. As expected the following equations hold: $3*29+42-15+9+74+1=198$ and $14*29+15*7-35+27+5*74+14=887$.