

RA

**stichting
mathematisch
centrum**



REKENAFDELING

NR 14/71

JANUARI

P.W. HEMKER en J.P. HOLLENBERG
MC-BGE CONVERSIE-PROGRAMMA

RA

2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O), by the Municipality of Amsterdam, by the University of Amsterdam, by the Free University at Amsterdam, and by industries.

Voorwoord

Op de hierna volgende 6 linkerbladzijden staat de tekst van een X8 - ALGOL 60 - programma, dat ALGOL 60 - teksten en procedures, die met behulp van RESYM ingelezen kunnen worden, omvormt tot Bull General Electric ALGOL 60 - teksten in ASCII - code.

Dit programma heeft ten doel ALGOL 60 - programma's en procedures die op het Mathematisch Centrum beschikbaar zijn op eenvoudige wijze bereikbaar te maken voor een Bull General Electric time sharing terminal.

Als object voor dat omzet-programma is in dit geval het programma zelf gekozen.

Het resultaat is zichtbaar gemaakt met behulp van een teletype en staat op de 6 hierna volgende rechterbladzijden.

P.W. Hemker en J.P. Hollenberg

begin comment Dit programma is geschreven om ALGOL 60-programma's die beschikbaar zijn in MC-flexowriter-code [2] om te zetten in gelijke programma's in ASCII-code [4].

In principe vindt alleen een omzetting plaats per 'basic symbol' [1].

De lay out van het programma zal i.h.a. ongewijzigd blijven. Een tab komt echter overeen met 2 spaties en bovendien wordt er voor gezorgd dat vanaf een nader te specificeren positie op de regel op een geschikt gekozen plaats een nieuwe regel begonnen wordt. Zoals vereist in B.G.E. time-sharing ALGOL wordt iedere nieuwe regel voorafgegaan door een regelnummer gevolgd door een niet-numeriek symbool (spatie). Deze regelnummering klimt op met tientallen om tussenvoegen van regels in het time-sharing system mogelijk te maken.

Aan het oorspronkelijk programma wordt de eis gesteld dat het geschreven is in correct ALGOL waarbij word-delimiters zijn onderstreept of tussen apostrophe's geplaatst en niet afgekort.

De omzetting per 'basic symbol' houdt in dat wanneer een omgezet programma niet direct bruikbaar is in de G.E. Time-Sharing Service dit slechts zijn oorzaak kan vinden in: 1. het verschillen van de input en output procedures in MC-ALGOL en time-sharing ALGOL. Zie [2] en [3].

2. de drastische beperking van de mogelijkheden van ALGOL 60 in het time-sharing ALGOL systeem.

Zie [1] en [3, p22 - 26].

Literatuur.:

- [1]. Revised report on the algorithmic language ALGOL 60. A/S Regnecentralen, Copenhagen.
- [2]. Het MC-ALGOL 60-systeem voor de X8. MR 81. Mathematisch Centrum, Amsterdam.
- [3]. G.E. Time-sharing system. Command system, Reference manual. Bull General Electric.
- [4]. G.E. Time-sharing system. ALGOL. Reference manual. Bull General Electric.;

```

10
20 BEGIN COMMENT DIT PROGRAMMA IS GESCHREVEN OM ALGØL
30 60-PROGRAMMA S
40 DIE BESCHIKBAAR ZIJN IN MC-FLEXØWRITER-CØDE [2]
50 OM TE ZETTEN
60 IN GELIJKE PROGRAMMA S IN ASCII-CØDE [4].
70 IN PRINCIPE VINDT ALLEEN EEN ØMZETTING PLAATS
80 PER
90 BASIC SYMBOL [1].
100 DE LAY ØUT VAN HET PROGRAMMA ZAL I.H.A. ØNGEWIJZIGD
110 BLIJVEN.EEN TAB KØMT ECHTER ØVEREEN MET 2 SPATIES
120 EN ØVEN-
130 DIEN WØRDT ER VØØR GEZØRGD DAT VANAF EEN NADER
140 TE SPECIFICEREN
150 ØSITIE ØP DE REGEL ØP EEN GESCHIKT GEKØZEN PLAATS
160 EEN
170 NIEUWE REGEL BEGØNNEN WØRDT. ZØALS VEREIST IN
180 B.G.E. TIME-
190 SHARING ALGØL WØRDT IEDERE NIEUWE REGEL VØØRAFGEØAAN
200 DØØR EEN
210 REGELNUMMER GEVØLGD DØØR EEN NIET-NUMERIEK SYMBOL(SPATIE)
220 .
230 DEZE REGELNUMMERING KLIMT ØP MET TIENTALLEN ØM
240 TUSSENVØEGEN
250 VAN REGELS IN HET TIME-SHARING SYSTEM MØGELIJK
260 TE MAKEN.
270 AAN HET ØØRSPRØNKELIJK PROGRAMMA WØRDT DE EIS
280 GESTELD
290 DAT HET GESCEVEN IS IN CØRECT ALGØL WAARBIJ
300 WØRD-DELIMITERS
310 ZIJN ØNDERSTREPT ØF TUSSEN APØSTROPHE S GEPLAATST
320 EN NIET
330 AFGEKØRT.
340 DE ØMZETTING PER BASIC SYMBOL HØUDT IN DAT
350 WANNEER EEN ØMGEZET PROGRAMMA NIET DIRECT BRUIKBAAR
360 IS IN DE
370 G.E.TIME-SHARING SERVICE DIT SLECHTS ZIJN ØØRZAAK
380 KAN VINDEN
390 IN: 1. HET VERSCHILLEN VAN DE INPUT EN ØUTPUT
400 PRØCEDURES
410 IN MC-ALGØL EN TIME-SHARING ALGØL.ZIE [2] EN
420 [3].
430 2. DE DRASTISCHE BEPERKING VAN DE MØGELIJKHEDEN
440 VAN
450 ALGØL 60 IN HET TIME-SHARING ALGØL SYSTEEM.
460 ZIE [1] EN [3,P22 - 26].
470
480 LITERATUUR.:
490 [1]. REVISED REPØRT ØN THE ALGØRITHMIC LANGUAGE
500 ALGØL 60.
510 A/S REGNECENTALEN,CØPENHAGEN.
520 [2]. HET MC-ALGØØL 60-SYSTEEM VØØR DE X8. MR 81.
530 MATHEMATISCH CENTRUM.AMSTERDAM.
540 [3]. G.E.TIME-SHARING SYSTEM. CØMMAND SYSTEM.REFERENCE
550 MANUAL.
560 BULL GENERAL ELECTRIC.
570 [4]. G.E.TIME-SHARING SYSTEM. ALGØL.REFERENCE
580 MANUAL.
590 BULL GENERAL ELECTRIC.;
600
610

```

```

integer i,j,thousands,hundreds,tens,maximal width,equals,
underlining,less,greater,not,division,space,bar,
and,or,tab,carriage,semicolon,close,comma,ttp,
quote,integer division,return,rubout,linefeed;
integer array conversion[0:255];
boolean space just punched;

```

```

procedure read and punch;
begin integer symbol;

```

```

procedure bar last symbol;
begin symbol:= resym;
    if symbol = less ∨ symbol = greater
        then outsymbol(quote) else
    if symbol = and then outsymbol(ttp) else
    if symbol = equals then outstring("/=) else
        goto again2
end bar last symbol;

```

```

procedure underlining last symbol;
begin
again: symbol:= resym;
    if symbol = underlining then goto again;
    if symbol = equals then outstring({equiv}) else
    if symbol = less then outstring({<=}) else
    if symbol = greater then outstring({>=}) else
    if symbol = not then outstring({imply}) else
    if symbol = division then
        outsymbol(integer division) else
            word delimiter
end underlining last symbol;

```

```

procedure word delimiter;
begin outsymbol(space); outsymbol(symbol);
again: symbol:= resym;
    if symbol = underlining then goto again;
    outsymbol(symbol); symbol:= resym;
    if symbol = underlining then goto again;
    outsymbol(space);
    goto again2
end word delimiter;

```

```

620  INTEGER I, J, THOUSANDS, HUNDREDS, TENS, MAXIMAL WIDTH,
630  EQUALS,
640  UNDERLINING, LESS, GREATER, NOT, DIVISION, SPACE,
650  BAR,
660  AND, OR, TAB, CARRIAGE, SEMICOLON, CLOSE, COMMA, TTP,
670
680  QUOTE, INTEGER DIVISION, RETURN, RUBOUT, LINEFEED;
690
700  INTEGER ARRAY CONVERSION(0:255);
710  BOOLEAN SPACE JUST PUNCHED;
720
730  PROCEDURE READ AND PUNCH;
740  BEGIN  INTEGER SYMBOL;
750
760  PROCEDURE BAR LAST SYMBOL;
770  BEGIN  SYMBOL:= RESYM;
780  IF SYMBOL = LESS OR SYMBOL = GREATER
790  THEN OUTSYMBOL(QUOTE) ELSE
800  IF SYMBOL = AND THEN OUTSYMBOL(TTP) ELSE
810
820  IF SYMBOL = EQUALS THEN OUTSTRING("/=") ELSE
830
840  GOTO AGAIN2
850  END BAR LAST SYMBOL;
860
870  PROCEDURE UNDERLINING LAST SYMBOL;
880  BEGIN
890  AGAIN:  SYMBOL:= RESYM;
900  IF SYMBOL = UNDERLINING THEN GOTO AGAIN;
910  IF SYMBOL = EQUALS THEN OUTSTRING("EQUIV")
920  ELSE
930  IF SYMBOL = LESS THEN OUTSTRING("<=") ELSE
940
950  IF SYMBOL = GREATER THEN OUTSTRING(">=")
960  ELSE
970  IF SYMBOL = NOT THEN OUTSTRING("IMPLY") ELSE
980
990  IF SYMBOL = DIVISION THEN
1000  OUTSYMBOL(INTEGER DIVISION) ELSE
1010  WORD DELIMETER
1020  END UNDERLINING LAST SYMBOL;
1030
1040  PROCEDURE WORD DELIMETER;
1050  BEGIN  OUTSYMBOL(SPACE); OUTSYMBOL(SYMBOL)
1060  ;
1070  AGAIN:  SYMBOL:= RESYM;
1080  IF SYMBOL = UNDERLINING THEN GOTO AGAIN;
1090
1100  OUTSYMBOL(SYMBOL); SYMBOL:= RESYM;
1110  IF SYMBOL = UNDERLINING THEN GOTO AGAIN;
1120
1130  OUTSYMBOL(SPACE);
1140  GOTO AGAIN2
1150  END WORD DELIMETER;
1160

```

```

again1: symbol:= resym;
again2: if symbol = carriage then new line else
        if symbol = underlining then underlininglastsymbol else
        if symbol = bar then bar last symbol else
        outsymbol(symbol);
        goto again1
end read and punch;

procedure new line;
begin outsymbol(return);
      outsymbol(linefeed);
      outsymbol(rubout);
      tens:= tens + 1;
      if tens = 10 then
      begin tens:= 0; hundreds:= hundreds + 1;
            if hundreds = 10 then
            begin hundreds:= 0;
                  thousands:= thousands + 1
            end
      end;
      if thousands ≠ 0 then goto 11 else
      if hundreds ≠ 0 then goto 12 else go to 13;
11: outsymbol(thousands);
12: outsymbol( hundreds);
13: outsymbol(      tens);
      outsymbol(0); outsymbol(space)
end new line;

procedure outstring(s);
string s;
begin integer i,symbol;
      i:= 0; outsymbol(space);
      for symbol:= STRINGSYMBOL(i,s) while symbol ≠ 255 do
      begin outsymbol(symbol);
            i:= i + 1
      end;
      outsymbol(space)
end outstring;

```



```

1170 AGAIN1: SYMBOL:= RESYM;
1180 AGAIN2: IF SYMBOL = CARRIAGE THEN NEW LINE ELSE
1190
1200     IF SYMBOL = UNDERLINING THEN UNDERLININGLASTSYMBOL
1210     ELSE
1220     IF SYMBOL = BAR THEN BAR LAST SYMBOL ELSE
1230
1240     OUTSYMBOL(SYMBOL);
1250     GOTO AGAIN1
1260 END READ AND PUNCH;
1270
1280 PROCEDURE NEW LINE;
1290 BEGIN OUTSYMBOL(RETURN);
1300     OUTSYMBOL(LINEFEED);
1310     OUTSYMBOL(RUBOUT);
1320     TENS:= TENS + 1;
1330     IF TENS = 10 THEN
1340     BEGIN TENS:= 0; HUNDREDS:= HUNDREDS + 1;
1350         IF HUNDREDS = 10 THEN
1360         BEGIN HUNDREDS:= 0;
1370             THOUSANDS:= THOUSANDS + 1
1380         END
1390     END ;
1400     IF THOUSANDS /= 0 THEN GOTO L1 ELSE
1410     IF HUNDREDS /= 0 THEN GOTO L2 ELSE GO TO L3;
1420
1430 L1: OUTSYMBOL(THOUSANDS);
1440 L2: OUTSYMBOL( HUNDREDS);
1450 L3: OUTSYMBOL( TENS);
1460     OUTSYMBOL(0); OUTSYMBOL(SPACE)
1470 END NEW LINE;
1480
1490 PROCEDURE OUTSTRING(S);
1500 STRING S;
1510 BEGIN INTEGER I,SYMBOL;
1520     I:= 0; OUTSYMBOL(SPACE);
1530     FOR SYMBOL:= STRINGSYMBOL(I,S) WHILE SYMBOL
1540     /= 255 DO
1550     BEGIN OUTSYMBOL(SYMBOL);
1560         I:= I + 1
1570     END ;
1580     OUTSYMBOL(SPACE)
1590 END OUTSTRING;
1600

```

```

.....
procedure outsymbol(symbol);
integer symbol;
begin   own integer position on line;
        if symbol = not then
            begin   outstring({not});
                    goto end outsymbol
            end;
        if symbol = or then
            begin   outstring({or});
                    goto end outsymbol
            end;
        if symbol = and then
            begin   outstring({and});
                    goto end outsymbol
            end;
        if symbol = tab then outsymbol(space);
        if symbol = space
            then   begin   if space just punched
                    then   go to end outsymbol
                    else   space just punched:= true
                    end
            else   space just punched:= false;

        PUHEP(conversion[symbol]);
        position on line:= position on line + 1;
        if symbol = return then position on line:= -2 else
        if (symbol = space  $\vee$  symbol = semi colon  $\vee$ 
            symbol = close  $\vee$  symbol = comma)  $\wedge$ 
            position on line > maximal width then
            begin   new line; outsymbol(tab);
                    outsymbol(tab); outsymbol(tab)
            end;
        end outsymbol:
        end outsymbol;

```

```

1610  PROCEDURE OUTSYMBOL(SYMBOL);
1620  INTEGER SYMBOL;
1630  BEGIN  OWN INTEGER POSITION ON LINE;
1640    IF SYMBOL = NOT THEN
1650      BEGIN  OUTSTRING("NOT");
1660        GOTO END OUTSYMBOL
1670      END ;
1680    IF SYMBOL = OR THEN
1690      BEGIN  OUTSTRING("OR");
1700        GOTO END OUTSYMBOL
1710      END ;
1720    IF SYMBOL = AND THEN
1730      BEGIN  OUTSTRING("AND");
1740        GOTO END OUTSYMBOL
1750      END ;
1760    IF SYMBOL = TAB THEN OUTSYMBOL(SPACE);
1770    IF SYMBOL = SPACE
1780      THEN  BEGIN  IF SPACE JUST PUNCHED
1790        THEN  GO TO END OUTSYMBOL
1800        ELSE  SPACE JUST PUNCHED:= TRUE
1810      END
1820    ELSE  SPACE JUST PUNCHED:= FALSE ;
1830
1840  PUHEP(CONVERSION[SYMBOL]);
1850  POSITION ON LINE:= POSITION ON LINE + 1;
1860  IF SYMBOL = RETURN THEN POSITION ON LINE:=
1870    -2 ELSE
1880  IF (SYMBOL = SPACE OR SYMBOL = SEMI COLON
1890    OR
1900  SYMBOL = CLOSE OR SYMBOL = COMMA) AND
1910  POSITION ON LINE > MAXIMAL WIDTH THEN
1920  BEGIN  NEW LINE; OUTSYMBOL(TAB);
1930    OUTSYMBOL(TAB); OUTSYMBOL(TAB)
1940  END ;
1950  END OUTSYMBOL;
1960  END OUTSYMBOL;
1970

```

```

integer procedure decimal(octal);
value octal;
integer octal;
begin integer a,b,c;
      a:= octal:100;
      b:=(octal - a×100):10;
      c:= octal - a×100 - b×10;
      decimal:= a×64 + b×8 + c
end decimal;

```

```

procedure fill conversion table (octal code);
integer octal code;
begin conversion[i]:= decimal(octal code);
      i:=i+1
end fill conversion table;

```

```

equals:=          70;
less:=            72;
greater:=         74;
not:=             76;
or:=              79;
and:=             80;
ttp:=            81;
integer division:= 82;
quote:=          83;
return:=         84;
linefeed:=      85;
rubout:=        86;
comma:=         87;
division:=      90;
semicolon:=     91;
space:=         93;
close:=         99;
tab:=          118;
carriage:=     119;
underlining:=  126;
bar:=          127;

```

```

1980  INTEGER PROCEDURE DECIMAL(OCTAL);
1990  VALUE OCTAL;
2000  INTEGER OCTAL;
2010  BEGIN  INTEGER A,B,C;
2020    A:= OCTAL\100;
2030    B:=(OCTAL - A*100)\10;
2040    C:= OCTAL - A*100 - B*10;
2050    DECIMAL:= A*64 + B*8 + C
2060  END DECIMAL;
2070
2080
2090
2100  PROCEDURE FILL CONVERSION TABLE (OCTAL CODE)
2110    ;
2120  INTEGER OCTAL CODE;
2130  BEGIN CONVERSION(I):= DECIMAL(OCTAL CODE);
2140    I:=I+1
2150  END FILL CONVERSION TABLE;
2160
2170  EQUALS:=      70;
2180  LESS:=        72;
2190  GREATER:=     74;
2200  NOT:=         76;
2210  OR:=          79;
2220  AND:=         80;
2230  TTP:=         81;
2240  INTEGER DIVISION:=  82;
2250  QUOTE:=       83;
2260  RETURN:=      84;
2270  LINEFEED:=    85;
2280  RUBOUT:=      86;
2290  COMMA:=       87;
2300  DIVISION:=    90;
2310  SEMICOLON:=   91;
2320  SPACE:=       93;
2330  CLOSE:=       99;
2340  TAB:=         118;
2350  CARRIAGE:=    119;
2360  UNDERLINING:= 126;
2370  BAR:=         127;
2380

```

```

for i:= 102 step 1 until 255 do conversion[i]:= 63;
maximal width:= READ;

i:= 0;
figures:for j:= 060,261,262,063,264,065,066,267,270,071 do
    fill conversion table (j);
lower case letters: for j:= 101,102,303,104,305,306,107,110,
    311,312,113,314,115,116,317,120,321,322,123,324,125,
    126,327,330,131,132,077 do fill conversion table (j);
upper case letters: for j:= 101,102,303,104,305,306,107,110,
    311,312,113,314,115,116,317,120,321,322,123,324,125,
    126,327,330,131,132,077 do fill conversion table (j);
special characters: for j:= 053,055,252,257,077,077,275,077,
    074,077,276,077 do fill conversion table (j);
undefined: for j:= 077,077,077,077,077 do
    fill conversion table (j);
ttp code:          fill conversion table (336);
int division code: fill conversion table (134);
quote code:       fill conversion table (042);
return code:      fill conversion table (215);
linefeed code:    fill conversion table (012);
rubout code:      fill conversion table (377);
other characters: for j:= 254,056,044,072,273,077,240,077,
    077,077,077,050,251,333,335 do fill conversion table(j);
conversion[121]:= decimal(042);
conversion[tab]:= conversion[120]:= decimal(240);
thousands:= hundreds:= tens:= 0;
space just punched:= false;

for i:= 1 step 1 until 10 do PUHEP(255);
read and punch

```

end

```

2390   FØR I:= 102 STEP 1 UNTIL 255 DØ CØNVERSIOØN[I]:=
2400     63;
2410   MAXIMAL WIDTH:= READ;
2420
2430   I:= 0;
2440   FIGURES: FØR J:= 060,261,262,063,264,065,066,
2450     267,270,071 DØ
2460     FILL CØNVERSIOØN TABLE (J);
2470   LØWER CASE LETTERS: FØR J:= 101,102,303,104,305,
2480     306,107,110,
2490     311,312,113,314,115,116,317,120,321,322,123,
2500     324,125,
2510     126,327,330,131,132,077 DØ FILL CØNVERSIOØN
2520     TABLE (J);
2530   UPPER CASE LETTERS: FØR J:= 101,102,303,104,305,
2540     306,107,110,
2550     311,312,113,314,115,116,317,120,321,322,123,
2560     324,125,
2570     126,327,330,131,132,07Ø DØ FILL CØNVERSIOØN
2580     TABLE (J);
2590   SPECIAL CHARACTERS: FØR J:= 053,055,252,257,077,
2600     077,275,077,
2610     074,077,276,077 DØ FILL CØNVERSIOØN TABLE (J)
2620     ;
2630   UNDEFINED: FØR J:= 077,077,077,077,077 DØ
2640     FILL CØNVERSIOØN TABLE (J);
2650   TTP CØDE:   FILL CØNVERSIOØN TABLE (336);
2660   INT DIVISIOØN CØDE: FILL CØNVERSIOØN TABLE (134)
2670     ;
2680   QUØTE CØDE:   FILL CØNVERSIOØN TABLE (042);
2690   RETURN CØDE:  FILL CØNVERSIOØN TABLE (215);
2700   LINEFEED CØDE: FILL CØNVERSIOØN TABLE (012);
2710   RUBØUT CØDE:  FILL CØNVERSIOØN TABLE (377);
2720   ØTHER CHARACTERS: FØR J:= 254,056,044,072,273,
2730     077,240,077,
2740     077,077,077,050,251,333,335 DØ FILL CØNVERSIOØN
2750     TABLE(J);
2760   CØNVERSIOØN[121]:= DECIMAL(042);
2770   CØNVERSIOØN[TAB]:= CØNVERSIOØN[120]:= DECIMAL(240)
2780     ;
2790   THØUSANDS:= HUNDREDS:= TENS:= 0;
2800   SPACE JUST PUNCHED:= FALSE ;
2810
2820   FØR I:= 1 STEP 1 UNTIL 10 DØ PUHEP(255);
2830   READ AND PUNCH
2840
2850 END
2860 50
2870

```