

9294
RA

**stichting
mathematisch
centrum**



LR 2.2

RA

JUNI 1971

D. GRUNE
HANDLEIDING BIJ DE TEKSTCORRECTOR

2e boerhaavestraat 49 amsterdam

BIBLIOTHEEK MATHEMATISCH CENTRUM
AMSTERDAM

1. Doel.

Het programma tcpp (tekst-corrector van papierband naar papierband) ponst, uitgaande van een gegeven ponsband en een ponsband met correctie-instructies een nieuwe band waarin de gespecificeerde correcties aangebracht zijn. Van het geheel wordt verslag uitgebracht op de regeldrukker. Alle ponsbanden zijn in MC-flexowriter code gesteld.

2. Inleiding.

2.1. De tekstcorrector tcpp heeft de volgende algemene eigenschappen:

2.1.1. De te corrigeren band (verder te noemen de datsymband) wordt sequentieel behandeld. Het is dus niet mogelijk bij voorbeeld eerst regel 50 te wijzigen en daarna regel 40.

2.1.2. Voor het specificeren van correcties is geen kennis omtrent de layout (plaatsing van spaties en tabs) nodig.

2.1.3. Voor het specificeren van correcties is geen kennis omtrent de regelnummering nodig; indien de regelnummering bekend is, kan hiervan wel nuttig gebruik gemaakt worden.

2.1.4. Eerst wordt de hele band met correcties ingelezen en de informatie hieruit opgeslagen; het einde van de correctieband wordt aangegeven door een instructie; zie 4.3.. Vervolgens worden de opgeslagen correctie-instructies in volgorde uitgevoerd: in het algemeen zal daarbij de datsymband worden ingelezen en een gecorrigeerde band geponst.

2.2. Bij de plaatsbepaling in de datsymband spelen vier begrippen een rol:

- het eerstvolgende symbool dat gelijk is aan een in een instructie gegeven symbool.
- de eerstvolgende regel die begint met een in een instructie gegeven rij symbolen.
- het begin van de eerstvolgende regel die een in een instructie gegeven regelnummer heeft.
- het einde van een regel.

3. Terminologie.

In deze beschrijving worden een aantal termen gebruikt die nadere toelichting behoeven.

3.1. Datsymband.

De datsymband is de oorspronkelijke band waarop de correcties aangebracht moeten worden.

3.2. Resultband.

De resultband is de nieuwe band waarop de correcties aangebracht zijn.

3.3. Symbool.

Onderstreepte en/of doorbalkte symbolen worden met hun onderstreping en/of doorbalking als een geheel beschouwd. De symbolen n , $\underline{n}$, $\#$, $\underline{\#}$, N , $\underline{N}$, $\#$ en $\underline{\#}$ zijn dus alle verschillend.

3.4. Regelnummer.

Het programma tcpp houdt de regelnummering van de datsymband bij, waarbij symbolen voorafgaande aan de eerste nlcr op regel 0 staan, die tussen de eerste nlcr en de tweede nlcr op regel 1, enz.. Zie echter 4.4.15. voor een instructie die de regelnummering kan bijstellen.

4. Correctie-instructies.

4.1. De correctie-instructies kunnen drie handelingen specificeren:

- Copieer de datsymband naar de resultband tot een bepaald punt in de datsymband.
- Skip de datsymband tot een bepaald punt.
- Voeg gegevens toe aan de resultband op het punt dat bereikt werd door de laatst uitgevoerde copier- of skip-handeling.

Bij elk van deze handelingen wordt de datsymband volledig sequentieel van voren naar achteren verwerkt. Algemeen geldt dat bij het zoeken van een bepaalde plaats in de datsymband door middel van vergelijking met gegeven strings of symbolen, spaties en tabs niet in deze vergelijking betrokken worden; nlcr's hebben alleen bij speciale correctie-instructies invloed. Voor het specificeren van een plaats in de invoertekst hoeft men dus niet de preciese layout te kennen, en is alleen datgene van belang wat zichtbaar op papier staat.

4.2. De correctieband bevat een of meer correctie-instructies; deze correctie-instructies hebben het volgende formaat:

<naam> <informatie> <nlcr>

De <naam> bestaat uit twee letters die de instructie identificeren. Spaties, tabs en nlcr's voor en in de naam worden geskipt; wordt de naam gevolgd door een spatie of een tab, dan wordt deze ene spatie of deze ene tab nog geskipt (dit om de leesbaarheid te verhogen); eventueel volgende spaties en/of tabs horen echter bij de <informatie>. De beide letters van de instructienaam mogen elk hoofdletter of kleine letter zijn. De <informatie> is afhankelijk van de instructie; er komen vier vormen voor:

1. string.

De string bestaat uit een rij van nul of meer symbolen welke eindigt bij de <nlcr>. Deze nlcr behoort niet meer tot de string.

2. blok.

Het eerste symbool van de <informatie> wordt blokquote genoemd. Het blok bestaat nu uit een op de eerste blokquote volgende rij symbolen, afgesloten door een symbool dat gelijk is aan de blokquote. Beide blokquotes behoren niet tot het blok. Eventuele symbolen tussen de tweede blokquote en de afsluitende <nlcr> zijn commentaar en worden geskipt.

B.v. de <informatie>:

```

† then begin int n; <nlcr>
    n:= read; <nlcr>
    fix(3,2,n,A);† gebruik fix <nlcr>

```

levert het blok

```

then begin int n; <nocr>
      n:= read; <nocr>
      fix(3,2,n,A);

```

waarbij <nocr> een nocr-poncing aanduidt, en † een doorbalkt not-teken is.

3. integer.

De <informatie> bestaat uit een <unsigned integer>, waarbij tabs en spaties buiten beschouwing blijven, eventueel gevolgd door commentaar dat niet met een cijfer mag beginnen. Bij het ontbreken van een <unsigned integer> wordt de waarde 0 aangenomen.

B.v.

```

18 <nocr>      geeft 18
+18 <nocr>     geeft 0
18+3 <nocr>    geeft 18
      <nocr>    geeft 0

```

4. geen.

Eventuele symbolen tussen <naam> en <nocr> zijn commentaar en worden geskipt.

4.3. De correctieband wordt afgesloten met de instructie 're' (zie 4.4.19.) of met de instructie 'st' (zie 4.4.18.). Het eerste symbool dat volgt op de <nocr> van deze sluitinstructie is het eerste symbool van de datsymband.

4.4. Instructies:

1. naam: cl
 informatie: string
 effect: copy line:

de datsymband wordt gecopieerd naar de resultband tot en met het begin van de eerste regel die begint met de gegeven string. Het laatste gecopieerde symbool is het laatste symbool van de string. Bij het vergelijken van het begin van een regel met de string worden spaties en tabs buiten beschouwing gelaten. Het copieren van de band geschiedt, ook voor het begin van de laatste regel, uiteraard met behoud van layout. Indien de string uit nul symbolen bestaat, wordt de datsymband gecopieerd tot en met de eerstvolgende nocr.

Voorbeeld:

de instructie

cl bgnintn; <nocr>

vindt de regel die begint met:

```

bgn int n;

```

2. naam: dl
 informatie: string

effect: delete line:

De datsymband wordt geskipt tot en met het begin van de eerste regel die begint met de gegeven string. Het laatste geskipte symbool is het laatste symbool van de string. Bij het vergelijken van het begin van een regel met de string worden spaties en tabs buiten beschouwing gelaten. Indien de string uit nul symbolen bestaat, wordt de datsymband geskipt tot en met de eerstvolgende nlcr.

3. naam: cc
 informatie: string
 effect: copy characters:

De datsymband wordt gecopieerd tot en met het eerste symbool uit de string, vervolgens tot en met het tweede symbool uit de string, enz.. Het laatste symbool dat gecopieerd wordt is het laatste symbool uit de string. Spaties en tabs in de string worden buiten beschouwing gelaten.

Voorbeeld:

de instructie

cc mmcas <nlcr>

vindt achtereenvolgens de symbolen aangeduid met een punt:

" programma's geschreven <nlcr>

in ALGOL 60 als " .
 . .

en is ekwivalent met

cc m <nlcr>
 cc m <nlcr>
 cc c <nlcr>
 cc a <nlcr>
 cc s <nlcr>

4. naam: dc
 informatie: string
 effect: delete characters:

De datsymband wordt geskipt tot en met het eerste symbool uit de string, vervolgens tot en met het tweede symbool uit de string, enz.. Het laatste symbool dat geskipt wordt is het laatste symbool uit de string. Spaties en tabs in de string worden buiten beschouwing gelaten.

5. naam: cx
 informatie: string
 effect: copy exclusively:

Een (1) symbool van de datsymband wordt gecopieerd en daarna wordt de datsymband gecopieerd tot (niet tot en met) het eerste symbool van de string. Vervolgens wordt een (1) symbool van de datsymband gecopieerd en daarna wordt de datsymband gecopieerd tot (niet tot en met) het tweede symbool van de string, enz.. Het eerste symbool dat niet meer gecopieerd wordt is het laatste symbool van de string. Spaties en tabs in de string worden buiten beschouwing gelaten.

Voorbeeld:

de instructie

```
cx mmcas <nlcr>
```

vindt, toegepast op dezelfde tekst als in het voorbeeld in 4.4.3., achtereenvolgens:

" programma's geschreven <n!cr>

in ALGOL 60 al

en is ekwivalent met

```
cx m    <n|cr>
cx m    <n|cr>
cx c    <n|cr>
cx a    <n|cr>
cx s    <n|cr>
```

De dubbel-m kan verwerkt worden dank zij het feit dat de cx-instructie altijd minstens 1 symbool verwerkt.

- ```
6. naam: dx
 informatie: string
 effect: delete exclusively:
```

Een (1) symbool van de datsymband wordt geskipt en daarna wordt de datsymband geskipt tot (niet tot en met) het eerste symbool van de string. Vervolgens wordt een (1) symbool van de datsymband geskipt en wordt de datsymband geskipt tot (niet tot en met) het tweede symbool van de string, enz.. Het eerste symbool dat niet meer geskipt wordt is het laatste symbool van de string. Spaties en tabs in de string worden buiten beschouwing gelaten.

- ```
7. naam:      cn
   informatie: integer
   effect:     copy to number;
```

De datsymband wordt gecopieerd tot het begin van de regel waarvan het regelnummer gelijk is aan de gegeven integer. Het laatste symbool dat gecopieerd wordt is de nlcr waarmee de genoemde regel begint.

opmerking:

Aangezien er altijd tot het begin van een regel gecopieerd wordt, kan men in een regel niet naar een regel met datzelfde nummer vragen.

8. naam: dn
informatie: integer
effect: delete to number:

De datsymband wordt geskipt tot het begin van de regel waarvan het regelnummer gelijk is aan de gegeven integer. Het laatste symbool dat geskipt wordt is de nlcr waarmee genoemde regel begint.

9. naam: ce
informatie: integer
effect: copy to end-of-line:

De datsymband wordt gecopieerd tot het einde van de onderhavige regel en daarna worden evenveel gehele regels gecopieerd als de integer aangeeft. Het laatste symbool dat gecopieerd wordt is het laatste symbool van de laatste regel (dus niet de nlcr).

Voorbeeld:

Wil men b.v., beginnend in regel 191, copieren tot het einde van regel 195, dan kan dat met de instructie:

ce 4 <nlcr>

Dit is ekwivalent met

ce 1 <nlcr>
ce 1 <nlcr>
ce 1 <nlcr>
ce 1 <nlcr>

maar niet met

ce <nlcr>
ce <nlcr>
ce <nlcr>
ce <nlcr>

In het laatste geval copieert de eerste instructie tot einde regel en maken de volgende drie instructies passen op de plaats.

10. naam: de
informatie: integer
effect: delete to end of line:

De datsyband wordt geskipt tot het einde van de onderhavige regel en daarna worden evenveel gehele regels geskipt als de integer aangeeft. Het laatste symbool dat geskipt wordt is het laatste symbool van de laatste geskipte regel (dus niet de nlcr).

11. naam: ic
informatie: string
effect: insert characters:

De string wordt, inclusief spaties en tabs, op de resultband gecopieerd. De datsyband is hierbij niet betrokken.

12. naam: il
informatie: string
effect: insert line:

Op de resultband wordt een nlcr geponst, waarna de string, inclusief spaties en tabs, op de resultband gecopieerd wordt. De datsyband is hierbij niet betrokken.

opmerking:

De instructie:

il <nlcr> ponst alleen een nlcr.

13. naam: ib
informatie: blok
effect: insert block:

Het blok wordt, inclusief spaties en tabs, op de resultband gecopieerd. De datsyband is hierbij niet betrokken.

14. naam: ec
informatie: string
effect: erase characters:

De datsyband wordt gecopieerd tot (niet tot en met) het eerste symbool uit de string, waarna dit symbool geskipt wordt. Vervolgens wordt de datsyband gecopieerd tot (niet tot en met) het tweede symbool uit de string, waarna dit symbool geskipt wordt, enz.. Het laatste symbool dat van de datsyband verwerkt wordt is het laatste symbool van de string. Spaties en tabs in de string worden hierbij buiten beschouwing gelaten.

Voorbeeld:

De instructie:

ec mmcas <nlcr>

vindt achtereenvolgens:

```

"                programma's geschreven
in ALGOL 60 als  ..      "
                . .

```

en ponst op de resultband

```

"                prograas' geshreven
in ALGOL 60 l   "

```

Genoemde instructie is ekwivalent met

```

cx m    <nocr>
dc m    <nocr>
cx m    <nocr>
dc m    <nocr>
cx c    <nocr>
dc c    <nocr>
cx a    <nocr>
dc a    <nocr>
cx s    <nocr>
dc s    <nocr>

```

15. naam: sn
 informatie: integer
 effect: set number:

De regeltelling van de datsymband wordt zo gewijzigd dat de onderhavige regel op de datsymband het nummer krijgt dat door de integer wordt aangegeven. De nummering telt daarna op de gebruikelijke wijze door.

opmerking:

Ten gevolge van de aanwezigheid van de monitorkop loopt de regeltelling op een programmaband en die op de afdruk van dat programma door de ALGOL-compiler niet gelijk. Door de correctieband te beginnen met:

```

ce 2    <nocr>
sn 0    <nocr>

```

wordt dit verholpen.

16. naam: td
 informatie: integer
 effect: transfer directly:

In de huidige versie van tcpp moet de integer een toegelaten pusymwaarde zijn, of een van de getallen 110, 111, 113 en 114. Voor een toegestane pusymwaarde wordt het betreffende symbool op de resultband geponst; voor de andere waarden geldt:

110. punch-off, het ponsen van verdere symbolen wordt onderdrukt
 111. punch-on, het ponsen van symbolen wordt hervat

113. new tape, er wordt een bandscheider geponst bestaande uit stopcode, runout, erase, runout, runout
114. runout

andere waarden:

Op de resultband wordt een markering geponst, bestaande uit een aantal schuine banen, voorafgegaan en gevolgd door runout.

opmerking:

van het uitvoeren van een td-instructie wordt op de regeldrukker melding gemaakt met de mededeling:

TRANSFERRED DIRECTLY: <integer>

17. naam: co
informatie: blok
effect: comment:

De instructie heeft geen effect.

opmerking:

Door achter de naam co twee spaties te ponsen wordt de tweede spatie tot blokquote gemaakt; volgt er nu 1 regel met daarin spaties, dan zal de eerste spatie het blok sluiten en de rest tot de nlcr is commentaar. B.v.

co edit op progr. 5, d.d. 1-1-71 <nlcr>

18. naam: st
informatie: geen
effect: stop:

De resultband wordt afgesloten met stopcode en runout en het programma wordt beeindigd.

19. naam: re
informatie: geen
effect: remainder:

De datsymband wordt tot het eind gecopieerd op de resultband, de resultband wordt afgesloten met stopcode en runout en het programma wordt beeindigd.

5. Uitvoer.

5.1. Bandponser.

De resultband wordt in "nette" vorm op de bandponser geponst.
Dit houdt het volgende in:

5.1.1. Overbodige spaties en tabs worden niet geponst.

5.1.2. Tabs worden alleen aan het begin van een regel gebruikt en dan optimaal.

5.1.3. Op regelmatige afstanden wordt automatisch runout gegeven, bij voorkeur bij een blanke regel; dit gebeurt echter nooit in het midden van een regel. Op de eerste regel volgt een stuk runout.

5.1.4. Bij een opeenvolging van meerdere blanke regels wordt de automatische runout gegeven vlak voor het laatste nlcr-symbool.

5.1.5. Een symbool wordt hoogstens eenmaal onderstreept en/of doorbalkt.

5.1.6. Niet-ponsbare symbolen geven aanleiding tot een markering op de band, bestaande uit een aantal schuine banen, voorafgegaan en gevolgd door runout (zie ook 6.2.1).

5.2. Regeldrukker.

Op de regeldrukker verschijnt een afdruk van de correctie-instructies, gevolgd door een afdruk van de resultband met regelnummer. Deze regelnummers gelden alleen voor de uitvoer en hebben geen enkele connectie met de regelnummering bij invoer (datsymband). Ook hier geldt dat symbolen voor de eerste nlcr zich op regel 0 bevinden. Indien een regel betrokken is geweest bij een correctie-instructie, wordt het regelnummer gevolgd door een sterretje. Indien direct voorafgaand aan een regel minstens 1 nlcr is geskipt (dus waarschijnlijk een hele regel is uitgewist), wordt het regelnummer gevolgd door een X. Op de regeldrukker worden verder foutmeldingen (zie 6.) afgedrukt en wordt het gebruik van de instructie 'td' gemeld.

6. Foutmeldingen.

6.1. In de correctie-instructies:

De fout wordt gemeld en het afdrukken van correctie-instructies gaat door. De instructies worden echter niet uitgevoerd en er wordt geen nieuwe band geponst.

6.1.1. UNKNOWN OPERATION

Onbekende instructienaam.

6.1.2. UNDEFINED SYMBOL ENCOUNTERED

Onbekend symbool op correctieband.

6.1.3. INPUT EXHAUSTED

Einde invoer in de correctieband.

6.1.4. INTEGER OVERFLOW

Als <informatie> wordt bij een instructie een integer meegegeven die de integercapaciteit overschrijdt.

6.2. Tijdens het corrigeren.

De fout wordt gemeld; de verdere verwerking hangt af van de fout.

6.2.1. UNDEFINED SYMBOL ENCOUNTERED

Onbekend symbool ter copiering aangeboden. Er wordt een markering (zie 5.1.6.) geponst en het corrigeren gaat door.
N.B. Onbekende ponsingen mogen wel geskipt worden.

6.2.2. SPECIFIED LINE ALREADY PROCESSED

Regelnummer gevraagd dat al gepasseerd is. De rest van de datsymband wordt zonder verdere correcties gecopieerd.

6.2.3. END OF FILE ENCOUNTERED WHILE COPYING

END OF FILE ENCOUNTERED WHILE DELETING

Datsymband uitgeput. De resultband wordt afgesloten.

7. Voorbeeld.

In 7.2. is een kort programma gegeven waarin enkele wijzigingen aangebracht moeten worden. Voor de duidelijkheid is de tekst van dit programmaatje voorzien van regelnummers; deze komen uiteraard niet op de invoerband (de datsymband) voor. In 7.1. wordt de tekst getoond van de correctieband, die de gewenste wijzigingen specificeert. Voor de duidelijkheid is de (anders onzichtbare) tab-ponsing aangeduid met <tab>. In 7.3. volgt een afdruk van de resulterende ponsband, terwijl 7.4. laat zien wat op de regeldrukker verschijnt.

7.1. Correctieband.

```

co  voorbeeld correctieband
cl  b
su  T
cc  m
ic  m
cn  2
dn  3
ec  o
ic  0
cn  5
ib  +<tab>if y > 2  V  y < 0 then
<tab>  begin print(y); exit end;
<tab>  comment het proces convergeert alleen voor
<tab><tab>  0 < y < (2/beginwaarde van x0);
+
cn  6
cx  e
dc  s
ic  8
cn  8
ce
ic  ;
RE

```

7.2. Programma.

```

1  begin  coment delen zonder deling;
2
3      real x0, x1, y;
4  next: y:= read;  x0:= 1;
5  loop: x1:= (2 - x0 × y) × x0;
6      if abs(x1 - x0) > eps then
7          begin x0:= x1; goto loop end;
8      print(y); print(x1); nlcr
9      goto next
10 end
11

```

7.3. Resultaat.

```

begin    comment delen zonder deling;
real x0, x1, y;
next:    y:= read;    x0:= 1;
          if y > 2 ∨ y < 0 then
          begin print(y); exit end;
          comment het proces convergeert alleen voor
              0 < y < (2/beginwaarde van x0);
loop:    x1:= (2 - x0 × y) × x0;
          if abs(x1 - x0) > 10-8 then
          begin x0:= x1; goto loop end;
          print(y); print(x1); nlcr;
          goto next
end

```

7.4. Regeldrukker output.

```

1 * BEGIN    COMMENT DELEN ZONDER DELING;
2X*          REAL X0, X1, Y;
3  NEXT:     Y:= READ;    X0:= 1;
4 *          IF Y ≥ 2  ∨  Y ≤ 0 THEN
5 *          BEGIN PRINT(Y); EXIT END;
6 *          COMMENT HET PROCES CONVERGEERT ALLEEN VOOR
7 *          0 < Y < (2/BEGINWAARDE VAN X0);
8 * LCOP:    X1:= (2 - X0 * Y) * X0;
9 *          IF ABS(X1 - X0) > 10-8 THEN
10          BEGIN X0:= X1; GOTO LOOP END;
11 *          PRINT(Y);  PRINT(X1);  NLCR;
12          GOTO NEXT
13  END
14

```

8. Overzicht van de correctie-instructies:

naam	informatie	effect
cl	string	copy line
dl	string	delete line
cc	string	copy characters
dc	string	delete characters
cx	string	copy characters exclusively
dx	string	delete characters exclusively
cn	integer	copy to line number
dn	integer	delete to line number
ce	integer	copy to end-of-line
de	integer	delete to end-of-line
ic	string	insert characters
il	string	insert characters on new line
ib	blok	insert block
ec	string	erase characters
sn	integer	set line number to integer
td	integer	punch-off(110),punch-on(111), new tape(113),runout(114)
co	blok	comment
st	geen	stop
re	geen	copy rest of input