HTPU 和 HEVE 參考手冊

產品號碼：AA-PXHGB-TE

2005 年 8月

本手冊爲中文文字處理公用程式（Hanyu Text Processing Utility, 簡稱 HTPU）和中文擴充多用途編輯程式 (Hanyu Extensible Versatile Editor, 簡稱 HEVE）之概略介紹。

修訂/更新資料： 本修訂文件取代 VMS/Hanyu AXP 1.5 版的 HTPU/HEVE 參考手冊

軟體版本: HP OpenVMS/Hanyu Alpha V8.2 版

Hewlett−Packard Company
Palo Alto, California

Printed in Singapore

目 錄

附圖

附表

序 言

手冊之目的

本手冊描述中文文字處理公用程式（HTPU）語言以及使用者介面中文擴充多用途編輯程式（HEVE）。其內容主要著重於 HTPU 和 HEVE 與 OpenVMS DECTPU 和 EVE 之間的差異，因此我們建議使用者參閱《DEC Text Processing Utility Reference Manual》以及《Extensible Versatile Editor Reference Manual》，以便瞭解與中文特色無關的特色描述。本手冊應作爲參考文件使用，而非詳細的操作手冊。

適用對象

本手冊描述 HTPU 的中文特色，因而適合對 DECTPU 的一般特色稍有瞭解的使用者研讀。請參閱《DEC Text Processing Utility Reference Manual》以及《Extensible Versatile Editor Reference Manual》，以便獲知 DECTPU和 EVE 一般特色的描述。

本書之結構

此手冊包含兩個部份和兩個附錄。第一部份有一章，包含 HTPU/HEVE 的一般說明。其中並包括 HTPU/HEVE 主要功能的詳細敘述。第二部份有兩章；分別提供了新增或修訂過的 HTPU 內建程序及 HEVE 命令的詳盡資訊。附錄 A 討論 HTPU/HEVE 的限制，而附錄 B 則列出新的 HTPU/HEVE 訊息。

相關文件

1 《HEVE 使用者手冊》

2 《Guide to the Extensible Versatile Editor》

3 《Guide to the DEC Text Processing Utility》

4 《Extensible Versatile Editor Reference Manual》

5 《DEC Text Processing Utility Reference Manual》

第 1 章

HTPU/HEVE 概述

HTPU（中文文字處理公用程式）在 OpenVMS/Hanyu 系統下執行。由於是以 DEC 文字處理公用程式爲基礎，所以 HTPU 在文字處理過程中能夠同時處理多位元組字元[1]及 ASCII 字元。透過一組完整新的 HTPU 和修訂過的 DECTPU 內建程序，就可完成 HTPU 的中文文字處理能力。

HTPU 提供以下額外的特色：

- 辨識並處理多位元組字元
- 符號組合及畫線能力
- 字元分類
- 全形與半形文數字之字元轉換
- 多位元組字元刪除及游標移動
- 包括多位元組字元之圖樣比對
- DECwindows Motif/Hanyu 視窗環境支援

HEVE（中文擴充多用途編輯程式）被設計來提供使用者和 HTPU 之間具親和力的介面。除了與 EVE[2] 有相同的功能之外，HEVE 還提供了一些新的加強功能來處理中文字元。

---

[1] 多位元組字元包含三種字元：a) 中文字，b) 同等於 DEC SICGCC 字 元集表中一位元組 ASCII 字元的二位元組字元，即全形字，c) 二位元組符號

[2] 除了 HEVE 不支援 MCS 字元編輯之外

1.1 引動 HTPU/HEVE

1.1.1 HTPU 的新增 DCL 命令動詞

以 HEVE（既定的編輯程式）引動 HTPU 的基本 DCL 命令如下：

```
$ EDIT/HTPU
```

HTPU 使用邏輯名稱 HTPU$SECTION，以便找出編輯程式使用者介面，該介面是經由系統定義以指出 HEVE 的區段檔案，亦即 HEVE$SECTION。

爲了節省鍵字，您可以定義如下的外部命令：

```
$ HEVE :== EDIT/HTPU
```

使用此外部命令，您只須鍵入如下的命令就可引動 HTPU 并使用 HEVE：

```
$ HEVE
```

若要引動 HTPU 并使用 HEVE 的 DECwindows Motif/Hanyu 介面，您可以使用下列 DCL 命令：

```
$ SET DISPLAY/CREATE/NODE=<您的工作站的節點名稱>
$ EDIT/HTPU/DISPLAY=MOTIF
```

而如果 HEVE 的定義如上，您亦可使用下面這個命令：

```
$ HEVE/DISPLAY=MOTIF
```

如果您將某個檔名納入作爲 HTPU 的一個參數，則該檔會被開啟以供編輯。

HTPU 使用如 DECTPU 中既定命令限定詞值的相同設定。新的命令限定詞、修訂過的命令限定詞值、邏輯名稱和檔案名稱均將於下面各章節中予以描述。

請注意：您必須在引動 HTPU 之前，先使用 HANYUGEN 公用程式來設定適當的終端機特性和處理環境。有關細節，請參閱《OpenVMS/Hanyu 使用者手冊》。

1.1.2 變更的命令限定詞值

/COMMAND
在 HTPU 中 /COMMAND 限定詞的既定值是 HTPU$COMMAND。

/DISPLAY
/DISPLAY 限定詞的既定值是 CHARACTER_CELL，該既定值將搜尋影像 HTPU$CCTSHR.EXE。而如果指定 MOTIF，則會使用 HTPU$MOTIFSHR.EXE。

/INITIALIZATION
HEVE 編輯程式的既定初設檔是 HEVE$INIT.EVE。如果您要使用邏輯名稱作爲 HEVE 初設檔，請將 HEVE$INIT 定義爲初設檔。

/SECTION
在 HTPU 中 /SECTION 限定詞的既定值是 HTPU$SECTION。

/WORK
HTPU 的既定工作檔案名稱是 HTPU$WORK.TPU$WORK。

1.1.3 變更的邏輯名稱

雖然 HTPU/HEVE 與 DECTPU/EVE 中有不同的邏輯名稱，但是它們卻有相似的用法。下表摘要了 HTPU/HEVE 中邏輯名稱的改變。

表 1-1 邏輯名稱的改變

DECTPU/EVE	HTPU/HEVE	描述
TPU$COMMAND	HTPU$COMMAND	命令檔案邏輯
TPU$DEBUG	HTPU$DEBUG	除錯檔案邏輯
TPU$DISPLAY_ MANAGER	HTPU$DISPALY_ MANAGER	螢幕更新器
TPU$JOURNAL	HTPU$JOURNAL	日誌檔案目錄
TPU$SECTION	HTPU$SECTION	區段檔案邏輯
TPU$WORK	HTPU$WORK	工作檔案邏輯
EVE$INIT	HEVE$INIT	HEVE初設檔
EVE$KEYPAD	HEVE$KEYPAD	HEVE 既定副鍵區

OpenVMS/Hanyu 具有定義爲 HEVE$SECTION 的系統性邏輯名稱 HPTU$SECTION。

1.1.4 變更的既定檔案名稱

下表摘要了 HTPU/HEVE 中的既定檔案名稱的改變。

表 1-2 既定檔案的改變

DECTPU/EVE	HTPU/HEVE	描述
TPU.DAT	HTPU.DAT	HTPU 既定 DECwindows Motif/Hanyu 資源檔案
TPU$COMMAND.TPU	HTPU$COMMAND.TPU	既定命令檔案
TPU$DEBUG.TPU	HTPU$DEBUG.TPU	既定偵錯檔案
TPU$WORK.TPU $WORK	HTPU$WORK.TPU $WORK	既定工作檔案
EVE.DAT	HEVE.DAT	HEVE 既定 DECwindows Motif/Hanyu 資源檔案
EVE$SECTION.TPU $SECTION	HEVE$SECTION.TPU $SECTION	既定區段檔案
EVE$INIT.EVE	HEVE$INIT.EVE	HEVE 的既定初設檔
EVE$WIDGETS _MOTIF.UID	HEVE$WIDGETS _MOTIF.UID	HEVE 的既定使用者介面描述 (UID) 檔

1.2 DEC SICGCC 字元集分類之新增 HTPU 關鍵字定義

爲了增強功能，HTPU 中加入以下關鍵字：

- ASCII_CHAR
- DISPLAY_CURSORM
- FULL_FORM
- HANYU
- JUSTIFY_BOTH
- JUSTIFY_LEFT
- JUSTIFY_RIGHT
- NON_HANYU
- SIZE_INVERT

ASCII_CHAR 代表 ASCII 字元的類別。關於關鍵字 ASCII_CHAR、FULL_FORM 和 SIZE_INVERT 的使用法，請參閱 CHANGE_CASE 內建程序的描述。關於半形字轉成全形字或者是相反轉換之限制，亦請參閱 CHANGE_CASE 內建程序的描述。

JUSTIFY_LEFT、JUSTIFY_RIGHT、以及 JUSTIFY_BOTH 均指定爲 JUSTIFY 內建程序，依此可執行方向的對齊。有關細節，請參閱 JUSTIFY 內建程序的描述。

用於 SET (DISPLAY_CURSOR) 內建程序中的 DISPLAY_CURSOR，可開啟或關閉 HPTU 的顯示游標模態。有關細節，請參閱 SET(DISPLAY_CURSOR) 內建程序。

HANYU 和 NON_HANYU 用來區分 DEC SICGCC 字元集表的不同區域。以下的圖表是字元集表分類的關鍵字定義。

所有四位元組的字元都被歸爲 HANYU 字元。

圖 1-1 DEC SICGCC 第一字面表

區段 1 - 9	SICGCC NON-HANYU 關鍵字 = NON_HANYU	A1A1 A9FE
區段 10 - 35	DEC RESERVED 關鍵字 = UNSPECIFIED	AAA1 C3FE
區段 36 - 93	SICGCC HANYU 關鍵字 = HANYU	C4A1 FDFE
區段 94	DEC RESERVED 關鍵字 = UNSPECIFIED	FEA1 FEFE

圖 1-2 DEC SICGCC 第二字面表

區段 1 - 82	SICGCC HANYU 關鍵字 = HANYU	A121 F27E
區段 83 - 94	DEC RESERVED 關鍵字 = UNSPECIFIED	F321 FE7E

1.3 HTPU/HEVE 與 DECTPU/EVE 在 DECwindows Motif 工作環境中的分別

HTPU/HEVE 和 DECTPU/EVE 在 DECwindows Motif 環境下基本上是一樣的，只不過 HTPU/HEVE 支援了中文編輯。以下的子部份將描述二者之間的差異。

1.3.1 中文懸垂及即現式功能表

HEVE 有英文和中文的下拉及懸垂功能表。您可以交換 HEVE 的功能表文字語言， 就如您交換 DECwindows Motif/Hanyu 的對話期管理程式的語言。

您只需要從對話期管理程式的 "選項" 懸垂式功能表使用 "語言" 項目。在 "語言選項" 懸垂功能表選擇 "中文－台灣" 來交換爲中文功能表文字，或英國英文交換爲英文功能表文字。

1.3.2 HTPU Widget 名稱

DECTPU 建立如表 1-4 所示的三種 widgets。

表 1-3 DECTPU widgets

類別	名稱	描述
Tpu	tpu	應用程式 shell
Main	tpu$mainwindow	主視窗
DECterm	DECwindows DECTPU	DECterm widget

在 HTPU 中，widget 的名稱已改變爲如表 1-5 所示。

表 1-4 HTPU widgets

類別	名稱	描述
Htpu	htpu	應用程式 shell
Main	tpu$mainwindow	主視窗
DECterm	DECwindows HTPU	DECterm widget

通常，這不會影響配備在 HTPU 和 / 或 HEVE 頂層的應用程式。然而，如果應用程式設定了那些 widgets 的資源，則應使用 HTPU所建立的 widgets 新名稱。

1.3.3 CSText Widget 支援

您可以從 HTPU 中建立 CSText widget，然後藉由 SET (TEXT) 內建程序來處理 CSText 的文字資源。

1.3.4 中文輸入法

當 HTPU 從 DECwindows Motif/Hanyu 下啟動時，您可以使用 LK201 鍵盤中的 <COMPOSE><SPACE> 鍵序，或僅使用 LK401 鍵盤中的 <COMPOSE> 鍵，來啟動中文字元輸入。一旦啟動中文輸入方法，則除了您必須使用與啟動時相同的鍵序來結束輸入方法外，其運作情形一如終端機。

在編輯對話期間，DECwindows Motif/Hanyu 的輸入方法處理可能會因某些理由而終止。當重新啟動輸入方法處理時，HTPU 並不會自動使用該新的輸入方法處理。因此您必須重新設定 HTPU 的輸入方法處理。

HEVE 在懸垂功能表 "選項" 下，提供了 "重設輸入方法" 項目。當輸入方法處理一度終止並重新啟動時，您就可以使用此命令來重新設定 HTPU 的輸入方法處理。

1.4 終端機支援

一般而言，HTPU/HEVE 在 OpenVMS/Hanyu 之下執行，並在 OpenVMS/Hanyu 所支援的影像終端機上提供了中文及 ASCII 文字的螢幕導向編輯。

1.5 雙語線上說明

HTPU/HEVE 提供了次系統層次的雙語功能。引動 HTPU/HEVE 之前，可以使用 HANYUGEN 公用程式來設定您所喜好的語言。

1.6 雙語訊息

HEVE 可以顯示中文或英文的資訊性訊息。引動 HTPU/HEVE 之前，使用者可以用 HANYUGEN 公用程式來設定其喜好的語言。

第 2 章

新增及修訂的 HTPU 內建程序

本章的內容主要是描述 HTPU 中新增的或修訂過的內建程序。若想獲知 DECTPU 內建程序的完整清單，請參閱 《DEC Text Processing Utility Reference Manual》。

下表列出 HTPU 中新增的內建程序。

表 2-1 新增的 HTPU 內建程序

內建程序
ALIGN_CURSOR
CHANGE_SIZE
CHARACTER_CLASS
COMPOSE
DEC_HANYU
IS_CLASS
JUSTIFY
SET(ALIGNMENT_DEFAULT)
SET(DISPLAY_CURSOR)
SET(FILL_NOT_BEGIN)
SET(FILL_NOT_END)
SET(MARGIN_ALLOWANCE)
SET(UNIT_DEFAULT)

下表列出 HTPU 中修訂 的內建程序。

表 2-2 修訂的 HTPU 內建程序

內建程序
CHANGE_CASE
CURRENT_OFFSET
CURSOR_HORIZONTAL
CURSOR_VERTICAL
EDIT
FILL
GET_INFO
INDEX
KEY_NAME
LENGTH
MARK
READ_CHAR
READ_KEY
SCROLL
SEARCH
SELECT
SELECT_RANGE
SPLIT_LINE
SUBSTR

ALIGN_CURSOR

此內建程序將游標移動到字元之第一欄，並且不產生任何副作用。

格式 **ALIGN_CURSOR**

描述 當游標不在多位元組字元的第一位元組時，ALIGN_CURSOR會將游標移到字元的第一欄。

如果游標正在現用字元的第一位元組，此內建程序不產生任何作用。

CHANGE_CASE

這個內建程序修訂某指定本文單元中所有字母的大小寫字體。並將二位元組（2-byte）全形字轉變爲 ASCII 半形字，或者是相反之轉換。

格式

[return_value :=]CHANGE_CASE ***(*** { *buffer* / *range* / *string* } **,**
*keyword1**[,***
*keyword2**])***

參數

buffer
是您欲改變大小寫字體的字元緩衝區。請注意如果第一個參數指定爲一緩衝區，則第三個參數就不能指定關鍵字 NOT_IN_PLACE。

range
是您欲改變大小寫字體字元的範圍。請注意，如果第一個參數指定爲一範圍，則第三個參數就不能指定關鍵字 NOT_IN_PLACE。

string
是一個變數名稱，它代表一個字串常數或是一個指定字串的式子。這個字串就是您欲改變的字串。若您將第三個參數指定爲 IN_PLACE，則 CHANGE_CASE 會對第一個參數所指定的字串做指定的更改。CHANGE_CASE 對字串常數無效。

keyword1
此關鍵字指明您想做何種改變。

- LOWER - 將某指定字串中的字母字元（全形或半形）改變成小寫。
- UPPER - 將某指定字串中的字母字元（全形或半形）改變成大寫。
- INVERT - 將某指定字串中的字母字元（全形或半形）大寫的改成小寫；小寫的改成大寫。
- ASCII_CHAR - 將指定字串中屬於 DEC SICGCC 字元表格的文數字元改成 ASCII (半形) 字元。
- FULL_FORM - 將某指定字串中的 ASCII 文數字元（半形）改成二位元組全形字元。
- SIZE_INVERT - 將某指定字串中的 ASCII 字母字元（半形）改成二位元組全形字元；二位元組全形字元改爲 ASCII 半形字元。

keyword2
此關鍵字指定結果的回返位置

- IN_PLACE - 告訴 HTPU 將改變緩衝區（buffer）、範（range）或某指定的字串（string）。此爲既定值。
- NOT_IN_PLACE - 告訴 HTPU 不改變原字串，但把改變後的字串當做回轉值傳回。若第一個參數指定爲範圍或緩衝區，則不能使用 NOT_IN_PLACE。若使用 NOT_IN_PLACE，您一定要指定一個回轉值給 CHANGE_CASE。

回轉值　指定一個變數當做傳回結果。

- returned_buffer - 如果您把一個緩衝區指定給第一個參數，則回轉值將被放進這個形式爲 "緩衝區" 的變數中。此 "returned_buffer"變數所指向的緩衝區, 與指定給第一個參數的緩衝區相同。
- returned_range - 若您指定一個範圍變數給第一個參數，則有一個叫做 "returned_range" 的範圍將含有修改過後的文字。

而且這個範圍與參數中指定的範圍都有相同的文字。但他們卻是獨立的範圍。若您接著改變或刪除其中一個範圍, 則對另一個範圍將無影響。

- return_string - 若您第一個參數指定爲一字串變數，則一個叫做 "return_string" 的字串將也含有修改過後的文字。即使您指定 IN_PLACE，CHANGE_CASE 也會傳回這個字串。

描述

若指定回轉值，則 CHANGE_CASE 傳回改變後的結果。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_ARGMISMATCH	ERROR	CHANGE_CASE 中的一個參數資料型態不對。
TPU$_INVPARAM	ERROR	CHANGE_CASE 中的一個參數資料型態不對。
TPU$_BADKEY	WARNING	您給了 CHANGE_CASE 錯誤的關鍵字。
TPU$_NOTMODIFIABLE	WARNING	您不能在不能修改的緩衝區中改變文字的大小寫字體。
TPU$_CONTROLC	ERROR	在執行 CHANGE_CASE 的過程中，您按了 CTRL/C。

CHANGE_SIZE

此內建程序把二位元組全形字元轉換爲半形 ASCII 字元，或者是相反轉換，來修改某指定單位文字中的所有字母字元的字形大小。

格式

[return_value :=]CHANGE_SIZE ***(*** { *buffer* / *range* / *string* } **,** *keyword1* ***[,*** *keyword2* ***])***

參數

buffer
是您欲改變字形大小的字元緩衝區。請注意如果第一個參數指定爲一緩衝區，則第三個參數就不能指定關鍵字 NOT_IN_PLACE。

range
是您欲改變字形大小的字元範圍。請注意，如果第一個參數指定爲一範圍，則第三個參數就不能指定關鍵字 NOT_IN_PLACE。

string
是一個變數名稱，它代表一個字串常數或是一個指定字串的式子。這個字串就是您欲改變的字串。若您將第三個參數指定爲 IN_PLACE，則 CHANGE_SIZE 會對第一個參數所指定的字串做指定的更改。CHANGE_SIZE 對字串常數無效。

keyword1
此關鍵字指明您想做的改變。

- ASCII_CHAR - 將某指定字串中 DEC SICGCC 字元表中的二位元組文數字字元改成 ASCII (半形) 字元。

- FULL_FORM - 將某指定字串中的 ASCII 文數字字元 (半形) 成二位元組全形字元。

- SIZE_INVERT - 將某指定字串中的 ASCII 文數字字元 (半形) 改成二位元組全形字元，二位元組全形字元改成 ASCII 半形字元。

keyword2
此關鍵字指明傳回結果的位置。

- IN_PLACE - 告訴 HTPU 在緩衝區，範圍或指定的字串中做指定的改變。這是既定的。

- NOT_IN_PLACE - 告訴 HTPU 保留原字元不更改並傳回一個新的更改後的字串，若第一個參數指定爲緩衝區或範圍時，不能使用NOT_IN_PLACE。使用 NOT_IN_PLACE 時，您必須指定一個傳回值給 CHANGE_SIZE。

回轉值 指定給傳回結果的變數。

- returned_buffer – 若您第一個參數指定爲緩衝區型式的變數，則一個叫做 "returned_buffer" 的變數將指向那個含有修改過文字的緩衝區。

- returned_range – 若您第一個參數指定爲一個範圍變數，則一個叫做 "returned_range" 的範圍將也含有修改後的本文。這個範圍與參數指定的範圍指向同一本文，但他們卻是獨立的範圍。若您接著改變或刪除其中一個範圍，則對另一個範圍無影響。

- return_string – 若您第一個參數指定爲一個字串變數，則一個叫做 "return_string"的字串也將含有修改後的本文。即使您指定 IN_PLACE，CHANGE_SIZE 也會傳回這一個字串。

描述	若指定回轉值，則 CHANGE_SIZE 會傳回修改結果。		
示錯訊息	TPU$_TOOFEW	ERROR	參數太少。
	TPU$_TOOMANY	ERROR	參數太多。
	TPU$_ARGMISMATCH	ERROR	CHANGE_SIZE 中的一個參數資料型態不對。
	TPU$_INVPARAM	ERROR	CHANGE_SIZE 中的一個參數資料型態不對。
	TPU$_BADKEY	WARNING	指定錯誤的關鍵字給 CHANGE_ SIZE。
	TPU$_NOTMODIFIABLE	WARNING	您不能在不可修改的緩衝區中改變文字的字形大小。
	TPU$_CONTROLC	ERROR	您在執行 CHANGE _SIZE 的過程中，按 CTRL/C。

CHARACTER_CLASS

此內建程序能夠辨認輸入字串中的第一個字元的類別（例如中文、非中文）。請注意，HTPU 會將多位元組字元視爲一個字元。

格式 **keyword := CHARACTER_CLASS** *(string)*

參數 ***string***
一個加上引號的字串，它是表示一個字串常數的變數名稱或是一個指定字串的式子。若字串長度超過一個字元，則第一個字元的字元種類會被傳回。

描述 表示字串中第一個字元的字元種類之所有關鍵字如下：

- ASCII_CHAR - ASCII 字元 (半形字元)。
- NON_HANYU - 請參閱本手冊中圖 1-1 及圖 1-2 對此關鍵字的說明。
- HANYU - 若想獲知有關這個關鍵字的描述，請參閱本手冊中的圖1-1 和圖 1-2。

  請注意，所有四位元組的字元都歸入 HANYU 類別
- UNSPECIFIED - 請參閱本手冊中圖 1-1 及圖 1-2 對此關鍵字的說明。

示錯訊息	TPU$_TOOFEW	ERROR	參數太少。
	TPU$_TOOMANY	ERROR	參數太多。
	TPU$_INVPARAM	ERROR	參數型態不對。
	TPU$_NEEDTOASSIGN	ERROR	CHARACTER _CLASS 必須在賦值述句右側。

COMPOSE

此內建程序會將 ASCII 字元字串轉換爲某些特殊的二位元組符號。

格式 **string2 := COMPOSE** *(string1)*

參數 ***string1***
一個加上引號的字串，它是表示一個字串常數的變數名稱或是一個指定字串的式子。下表列出 ASCII 字元的字串及其將被轉換成的二位元組符號及二位元組字元。

表 2-3 **ASCII 字串及相對的二位元組符號**

ASCII 字	相應的二位元組符號
0	│
1	└
2	┴
3	┘
4	├
5	┼
6	┤
7	┌
8	┬
9	┐
-	─
||	‖
|^	↑
|V	↓
([	〔

表 2-3 **ASCII** 字串及相應的二位元組符號（續）

ASCII 字	相應的二位元組符號
()	◯
)]	】
[[	「
[(	【
[]	□
])	〕
]]	」
<<	≪
<=	≦
<>	◇
<\	△
</	▽
<-	←
>>	≫
>=	≧
><	☆
>-	→
:-	÷
+-	±
xx	×
=/	≠
=|	∵
,.	…
..	‥
.;	∴
.C	℃
``	“
''	″
C/	¢
|-	£
SS	§
OO	∞
O>	♉
O+	♀

表 2-3 **ASCII** 字串及相應的二位元組符號（續）

ASCII 字	相應的二位元組符號
K><	★
K()	●
K<>	◆
K[]	■
K<\	▲

示錯訊息	TPU$_TOOFEW	ERROR	參數太少。
	TPU$_TOOMANYE	ERROR	參數太多。
	TPU$_INVPARAM	ERROR	參數型態錯誤。
	TPU$_NEEDTOASSIGNE	ERROR	COMPOSE 必須在賦值述句右側。

範例

```
1     PROCEDURE user_compose

      COPY_TEXT (COMPOSE ('K><ss'))

      ENDPROCEDURE
```

以上程序將 '★§' 插入現在游標所在的

```
2     PROCEDURE user_part_compose

      COPY_TEXT (COMPOSE ('=/aK><'))

      ENDPROCEDURE
```

以上程序將'≠a★'插入項在游標所在的編輯位置。

CURRENT_OFFSET

此內建程序傳回一個整數，此整數爲現用列中現用字元位置的位移。

格式 **integer := CURRENT_OFFSET *[(keyword)]***

參數 ***keyword***
可被指定的關鍵字如下：

- CHARACTERS - 字元位移。位移值將根據字元單位計數。
- BYTES - 位元組位移。位移值將根據位元組單位計數。

若您不指定關鍵字，則 CHARACTERS 爲既定的。您也可以用 SET(UNIT_DEFAULT) 內建程序改變既定值。請參閱 SET(UNIT_DEFAULT) 的完整描述。

描述 有關 CURRENT_OFFSET 內建程序的基本描述，請參閱《DEC Text Processing Utility Reference Manual》。

在 HTPU 中，一旦加入關鍵字參數，則可以使用字元位移和位元組位移。傳回的位移位於字元界。

示錯訊息

TPU$_NEEDTOASSIGN	ERROR	CURRENT_OFFSET 必須在賦值述句的右側。
TPU$_TOOFEW	ERROR	參數太少。
TPU$_NOCURRENTBUF	ERROR	您並不位於緩衝區中。

範例

```
1     my_char_off := CURRENT_OFFSET (CHARACTERS)
```

此賦值述句用來指定一個整數給變數 my_char_off。此整數代表現用字元的字元位移。

```
2     my_byte_off := CURRENT_OFFSET (BYTES)
```

此賦值述句用來指定一個整數給變數 my_byte_off，此整數代表現用字元的位元組位移。

CURSOR_HORIZONTAL

此內建程序依指定的螢幕行數，水平移動游標。

格式 **[integer2 :=]CURSOR_HORIZONTAL** *(integer1 **[**,keyword**]**)*

參數

integer1
以帶正負（+/-）號的整數值指定游標所要移動的螢幕行數。

keyword
可以指定下列關鍵字：

- CHARACTERS - 如果游標移動之後，它的位置不在多位元組字元的 第一個位元上，則 CURSOR_HORIZONTAL 會讓游標變回第一欄。
- BYTES - 移動與緩衝區中內容無關的游標，除非啟用 DISPLAY_CURSOR 模態，因而，雖然內部游標的位置可能並不位於第一欄，但螢幕游標卻會顯示於字元的第一欄。有關細節，請參閱SET(DISPLAY_CURSOR) 的描述。

如果未指定關鍵字，則既定爲 BYTES。您也可以使用 SET(ALIGNMENT_DEFAULT) 內建程序來變更此既定。若需有關完整的描述，請參閱 SET(ALIGNMENT_DEFAULT) 部分。

回轉值 回轉值表示游標已經移動的行數。若 HTPU 無法依 integer1 所指定的行數移動，則游標將儘可能地移動最多行數。HTPU 允許回轉值爲負值，但此表示法保留給將來的 HTPU 版本。此回轉值表示游標向右或左移動的確實行數。若游標未移動，回轉值爲 0。若 CURSOR_HORIZONTAL 產生錯誤，則回轉值無法預期。

描述

有關 CURSOR_HORIZONTAL 內建程序的基本描述，請參閱《DEC Text Processing Utility Reference Manual》

在 HTPU 中，您可用關鍵字參數來控制內建運算之後游標的位置。

由於中文字元佔用多欄，因此 CURSOR_HORIZONTAL 可以藉由在第二個參數中指定 BYTES 關鍵字，而將游標移動至多位元組字元第一欄以外的地方。

當啟用 DISPLAY_CURSOR 模態時，螢幕上顯示的游標都位於字元的第一個位元組，而不論該字元爲單一位元組或是多位元組。然而，內部儲存的游標位置卻可以位於字元的非第一個位元組上。若需更多的資訊，請參閱有關 SET(DISPLAY_CURSOR) 的參考部份。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_INVPARAM	ERROR	參數型態不對。

CURSOR_VERTICAL

此內建程序依指定的螢幕列數，上下移動游標。

格式 **[integer2 :=]CURSOR_VERTICAL** (*integer1* ***[****,keyword****]****)*

參數 ***integer1***
以帶正負(+/-)的整數值，指定游標移動的螢幕列數。

keyword
可以指定下列關鍵字：

- CHARACTERS - 如果游標在移動之後並非位於多位元組字元的第 一個位元組上，則 CURSOR_VERTICAL 會將游標回到第一欄。

- BYTES - 移動與緩衝區中內容無關的游標，除非啟用了 DISPLAY_CURSOR 模態，因而，雖然內部游標的位置可能並不位於第一欄，但螢幕游標卻會顯示於字元的第一欄。有關細節，請參閱 SET(DISPLAY_CURSOR) 的描述。

如果未指定關鍵字，則既定爲 BYTES。您也可以使用 SET(ALIGNMENT_DEFAULT) 內建程序來變更此既定。若需有關完整的描述，請參閱 SET(ALIGNMENT_DEFAULT) 部分。

回轉值 此回轉值表示游標上或下移動的列數。若 HTPU 無法依 integer1 指定的列數移動時，則游標將儘可能最多移動列數。若 CROSS_WINDOW_BOUNDS 設爲 ON，CURSOR_VERTICAL 可以移動到另 一個視窗，在此狀況下，回轉值爲負數。自負的回轉值不表示游標已經向上移動。若游標沒有移動，則回轉值爲 0，若 CURSOR_VERTICAL 產生錯誤，則回轉值無法預期。

描述

有關 CURSOR_VERTICAL 內建式程序的基本描述，請參閱《DEC Text Processing Utility Reference Manual》。

在 HTPU 中，您可用關鍵字參數來控制內建運算之後游標的位置。

此程序基本上並未改變游標的欄位。然而，如果 CHARACTERS 已經於關鍵字欄位中加以指定，則已移動至多位元組字元的非第一個欄位的游標，會被轉換至第一欄。倘使您不想移動游標的欄位位置，則請於關鍵字欄位中指定 BYTES。

當您在顯示游標模態設定爲 ON 時使用了 BYTES 關鍵字，則儘管內部儲存游標所在的欄位並未改變，但螢幕游標卻通常位於多位元組字元的字元界上。有關進一步的資訊，請參閱 SET(DISPLAY_CURSOR) 部份。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_INVPARAM	ERROR	參數型態不對。

DEC_HANYU

此程序完成 ASCII 碼或 DEC SICGCC 字元碼之間的字碼轉換。

格式 **integer := DEC_HANYU** *(string1)*

格式 **string := DEC_HANYU** *(integer1)*

參數

integer1
代表您想轉換的 ASCII 字元或多位元組字元的一個整數。

此整數 (十進位數) 決定於 ASCII 或 DEC SICGCC 字元集。

string1
是您想要得到 ASCII 碼或 DEC SICGCC 字元碼的字元。

描述

如果參數是整數，則 DEC_HANYU 內建程序會根據所指定的數字而傳回一個一位元組或多位元組的字元。至於數字與字元的比對，主要是依循ASCII 或 DEC SICGCC 字元集的規則。然而，DEC_HANYU 並不會提供所指定的整數在 DEC SICGCC 表格中是否有效的資訊。

若參數是字串，則 DEC_HANYU 內建程序會根據 ASCII 或 DEC SICGCC 字元表格傳回此字串第一個字元對應的整數。

示錯訊息	TPU$_NEEDTOASSIGN	ERROR	DEC_HANYU 必須在賦值述句的右邊。
	TPU$_TOOFEW	ERROR	參數太少。
	TPU$_TOOMANY	ERROR	參數太多。
	TPU$_ARGMISMATCH	ERROR	參數的資料型態不對。
	TPU$_NULLSTRING	ERROR	您傳了一個長度是 0 的字串給 DEC_HANYU。

範例

下列程序將一個 DEC SICGCC 字元區段中的一個表格插到現用緩衝區。例：當您執行 user_hanyu_list(20)，第 20 區段的表會插入正在編輯的位置上。

```
PROCEDURE user_hanyu_list (section)
LOCAL   cnt, col, low_byte;

cnt := section * 256 + 41120;
max := cnt + 94;

COPY_TEXT (FAO (' 第 :ZL 行',section));
SPLIT_LINE;

COPY_TEXT (user_hex (cnt) + '   ');
col := 1;
LOOP
   EXITIF cnt > max;
   IF col > 16 THEN
      SPLIT_LINE; UPDATE (current_window);
      col := 1;
      COPY_TEXT (user_hex (cnt) + '   ');
   ENDIF;
   low_byte := cnt - ((cnt / 256) * 256);
```

```
   IF low_byte <> 160 THEN
      COPY_TEXT (DEC_HANYU (cnt));
   ELSE
      COPY_TEXT ('   ');
   ENDIF;
   cnt := cnt + 1;
   cnt := col + 1;
ENDLOOP;
SPLIT_LINE;
ENDPROCEDURE

PROCEDURE user_hex (dec_num)
LOCAL   res, rmn, temp;

temp := dec_num;
IF temp = 0 THEN res := '0' ELSE res := " ENDIF;
LOOP
   EXITIF temp <=0;
   rmn := temp - ((TEMP / 16) * 16);
   temp := temp / 16;
   IF (0 < rmn) AND (rmn < 16) THEN
      res := SUBSTR ('123456789ABCDEF', rmn, 1) + res;
   BLSE
      res := '0' + res;
   ENDIF;
ENDLOOP;
user_hex := res;
ENDPROCEDURE
```

下列幾列是執行 user_hanyu_list(40) 命令之後，於現在編輯位置上顯示的內容。

第 40 行

```
          １２３４５６７８９ＡＢＣＤＥＦＧ
C8A0        年式弛忙忖戎戌戍成扣扛托收早旨
C8B0      旬旭曲曳有朽朴朱朵次此死氖汝汗汙
C8C0      江池汐汕污汎汍汎灰牟牝百竹米糸缶
C8D0      羊羽老考而耒耳聿肉肋肌臣自至臼舌
C8E0      舛舟艮色艾虫血行衣西阡串亨位住佇
C8F0      佗佞伴佛何估佐佑伽伺伸佃佔似但
```

EDIT

此內建程序會根據指定的關鍵字，修改一個字串。EDIT 類似 DCL 詞彙功能 F$EDIT，但不是完全相同。內建程序與詞彙功能的不同點將在下列描述。

格式

[return_value :=]EDIT *(*{ *buffer* / *range* / *string* }*, keyword1[,...][,keyword2][,keyword3])*

參數

buffer
是您想編輯文稿的緩衝區。注意，若您指定第一個參數爲緩衝區，則第三個參數不能使用 NOT_IN_PLACE。

range
是您想編輯文稿的範圍。注意，若您指定第一個參數爲一範圍，則第三個參數不能使用 NOT_IN_PLACE。

string
是您要修改的字串。若您指定一個傳回值，則此傳回值由第一個參數指定的字串經過第二參數及其下參數指定的修改方式修改而成。若第三個參數指定爲 IN_PLACE，則 EDIT 依指定方法修改在第一個參數的字串。EDIT 對字串常數沒有影響。

keyword1
您可從下列表列中選擇多個關鍵字：

- COLLAPSE - 移去所有空白 (包含全形空白) 及跳欄。
- COMPRESS - 以一個單一半形空白取代多重空白。(包含全形空白) 及跳欄。

- TRIM - 移去前導空白 (包含全形空白) 和跳欄，以及尾部空白 (包含全形空白) 及跳欄。
- TRIM_LEADING - 移去前導空白 (包含全形空白) 及跳欄。
- TRIM_TRAILING -移去尾部空白 (包含全形空白) 及跳欄。
- LOWER - 所有大寫字母 (包含全形字母) 轉換成小寫。
- UPPER - 所有小寫字母 (包含全形字母) 轉換成大寫。
- INVERT - 轉換現用指定的字母小寫 (包含全形字母) 之大小寫字體；小寫轉成大寫，大寫轉成小寫。
- ASCII_CHAR - 將某指定字串中位於 DEC SICGCC 字元表格的兩位元組文數字字元轉換成 ASCII (半形) 字元。
- FULL_FORM - 將某指定字串中 ASCII 字元轉換成兩位元組全形文數字字元。
- SIZE_INVERT - 將指定字串中 ASCII 文數字字元轉換成兩位元組全形文數字字元，全形文數字字元轉爲 ASCII 字元。

若您指定多個 TRIM 關鍵字 (TRIM，TRIM_LEADING，TRIM_TRAILING)，則所有指定的 TRIM 動作都會執行。

若您指定多個轉換關鍵字（UPPER，LOWER，INVERT，ASCII_CHAR FULL_FORM，SIZE_INVERT），則最後一個關鍵字決定字串中的字元要如何修改。

keyword2
最後的參數選項可指定爲 ON 或 OFF。

- ON - 告訴 HTPU 對單或雙引號做分辨。在這種狀況下，引號內的子字串不會被更改。

- OFF - 告訴 HTPU 不對單或雙引號做分辨。引號內的子字串會按指定更改。

keyword3
指明 HTPU 在何處做指定的轉換。可用的轉換字如下：

- IN_PLACE - 在應有的位置完成指定的更改。此爲既定值。
- NOT_IN_PLACE - 保留未改變的指定字串，並且傳回一個指定編輯結果的字串。若第一個參數指定爲範圍或緩衝區，則不能使用 NOT_IN_PLACE。若要使用 NOT_IN_PLACE，您必須指定一個傳回值給 EDIT。

回轉值 指定變數給傳回的結果：

- returned_buffer - 若您第一個參數指定爲一個緩衝區變數，則一個型式爲緩衝區的變數將會指到含有修改後文稿的緩衝區。這個變數和第一個參數指定的緩衝區變數都指到相同的緩衝區。
- returned_range - 若您指定第一個參數爲一個範圍變數，則一個範圍將包含修改過後的文稿。此範圍與第一個參數指定的範圍變數有著相同的內容，但他們是獨立的範圍。若您改變或刪除其中一個範圍，對另一個範圍將無影響。
- return_string - 若您指定第一個參數爲一個字串變數，這個字串變數將含有此修改過的內容。即使您指定 IN_PLACE，EDIT 也會傳回一個字串。

描述 HTPU 修改第一個參數所指定的緩衝區、範圍及字串。

若不指定 keyword2，EDIT 不會修改字串中引號內的文字。例如：下列命令

```
EDIT ('HE SANG "WELL"', LOWER)
```

不會將 WELL 改成小寫。字串變成：

```
he sang "WELL"
```

若您只指定開式引號 (左引號) 而不指定關式引號，則會傳回 TPU$_MISSINGQUOTE 狀態，在此狀況下，文稿由不含閉式引號的開式引號開始到字串結尾將被視爲引號內字串的一部份，並且不會被修改。

您可以使用關鍵字 OFF 爲最後的參數，使 EDIT 內建程序停止辨識單、雙引號。因此當您指定關鍵字 OFF 時，引號將被視爲普通文稿處理。EDIT 類似於 DCL 詞彙功能 F$EDIT。然而您需注意下列不同點：

- F$EDIT 將結果傳回，而 EDIT 在適當地方修改字串。
- 當 F$EDIT 要求以字串來指定編輯命令時，而 EDIT 以關鍵字作爲參數。F$EDIT 無法以中文字元運作。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_INVPARAM	ERROR	參數資料型態不對。
TPU$_ARGMISMATCH	ERROR	EDIT 中的參數資料型態不對。
TPU$_MISSINGQUOTE	ERROR	字元串沒有終止引號。
TPU$_BADKEY	WARNING	使用錯誤的關鍵字。

範例　　下列述句將 "ＰＲＯＤＵＣＴ　ＮＡＭＥ" 轉換成 ASCII 字元，並且將編輯的字串顯示在訊息視窗內。

```
pn := 'ＰＲＯＤＵＣＴ ＮＡＭＥ';

EDIT (pn, ASCII_CHAR);
MESSAGE (pn);
```

FILL

此內建程序把某指定緩衝區或範圍內的文稿重排，使文稿每列的長度幾乎都相同。

格式

FILL *(* { *buffer* / *range* } ***[****,string****[****,integer1****[****,integer2****[****,integer3****]]]])***

參數

buffer
是您想重排文稿的緩衝區。

range
是您想重排文稿的範圍。

string
是您想在緩衝區文稿內當成字分離字元所使用的半形 ASCII 字元串列。ASCII 空格字元通常爲文字分離字元。至於字串本身則不能包含任何多位元組字元。

integer1
是左邊留白的值。左邊留白值必須大於 1 且小於右邊留白。此值的既定值爲緩衝區的左邊留白。

integer2
右邊留白值。此值須大於左邊留白值且不能超過緩衝區的最大紀錄的大小。此值的既定值爲緩衝區的右邊留白。

integer3
第一行內編值。此值用來修改第一行的左邊留白。可爲正或負。然而，此值加到左邊留白的結果必須大於 1 且小於右邊留白。既定值爲 0。

描述

有關 FILL 內建式程序的基本描述，請參閱《DEC Text Processing Utility Reference Manual》。

在 HTPU 中，FILL 內建程序已經予以增強，來處理多位元組字元的字串。如果第二個字元的位置超過了右邊界，則 FILL 會將該字串分成兩個相鄰的多位元組字元。

增強的 FILL 內建程序也可以處理例外字元。使用者可利用 SET(FILL_NOT_BEGIN) 及 SET(FILL_NOT_END) 內建程序指定例外字元。當 FILL 執行時，指定的例外字元會被考慮。以 SET (MARGIN_ALLOWANCE) 設定 MARGIN _ALLOWANCE，使用者可以指定可容忍之超過右邊界位置的例外字元數目。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_ARGMISMATCH	ERROR	FILL的參數之一資料型態不對。
TPU$_BADMARGINS	WARNING	您所指定的邊界之一不對。
TPU$_INVPARAM	ERROR	FILL 的參數之一資料型態不對。
TPU$_NOTMODIFIABLE	WARNING	您不能在一個不可修改的緩衝區中修改文字。
TPU$_CONTROLC	ERROR	在 FILL 的執行過程中按了CTRL/C 鍵。

TPU$_NOCACHE	ERROR	記憶體空間不足無法指定新 的快速存取記憶體。

GET_INFO

這個內建程序提供了有關編輯內文現狀的資訊。表 2-4 和表 2-5 分別列出了 GET_INFO 所傳回的資訊以及相關的 HTPU 增強部份。有關原始的 GET_INFO 功能，請參閱《DEC Text Processing Utility Reference Manual》。

格式 **return_value := GET_INFO** *(parameter1,parameter2)*

格式 **return_value := GET_INFO** *(parameter1,parameter2, parameter3)*

參數

parameter1
可以是 HTPU 資料型態或是關鍵字。GET_INFO 提供關於指定爲 parameter1 的項目之資訊。表 2-4 列出可以被指定爲 parameter1 的資料型態及相關的 HTPU 加強功能。表 2-5 列出可以被指定爲 parameter1 的關鍵字及相關的 HTPU 加強功能。

parameter2
是一個放在引號中的字串變數名稱或是一個列在表 2-4 或表 2-5 的字串常數的式子。這個當成 parameter2 用的字串，指出 parameter1 指定項目所要求的資訊種類。可輸入大寫字串或小寫字串。

parameter3
是一個放在引號中的字串或是一個變數名稱。有關這個參數的詳細描述，請參閱《DEC Text Processing Utility Reference Manual》。

描述 說明 HTPU 新功能摘要如下表：

表 2-4 使用變數作為 GET_INFO 的第一個參數

parameter1	parameter2	回轉值	回轉值簡述
緩衝區變數	"byte_offset"	整數	位移量的計算標準是根據現用字元位置與該列開頭的相對位置。
	"character_length"		整數游標所在字元的位元組數。若游標在一個二位元組字元上，回轉值爲 2 。
	"character_index"	整數	游標位置與字元第一個位元間的距離。若游標在 ASCII 字元上，其回值爲 0。若游標在一個二位元組字元的第二個位元組，其回轉值爲 1。
標示的變數	"byte_offset"	整數	此位移值的計算是根據列的開始到標記的相對位元組數目。
	"character_length"	整數	記所在位置字元的位元組數。當游標在一個二位元組字元上，回轉值爲 2。
	"character_index"	整數	標記位置與字元的第一行間的距離。當游標在 ASCII 字元上時，回轉值爲 0。當游標在一個二位元組字的第二位元組，回轉值爲 1。
視窗變數	"character_index"	整數	意指在 HTPU 內部游標位置和螢幕游標位置之間的距離。只有當顯示-游標模態設定爲 ON 時，該值才有可能不爲零。

表 2-5 使用關鍵字作為 GET_INFO 的第一個參數

parameter1	parameter2	parameter3	回轉值	回轉值描述
SYSTEM	"FILL_NOT_ BEGIN"		字串	SET(FILL_NOT _BEGIN) 內建程序指定的字串。
SYSTEM	"FILL_NOT_ END"		字串	SET(FILL_NOT _END) 內建程序指定的字串。
SYSTEM	"MARGIN_ ALLOWANCE"		整數	由 SET (MARGIN _ALLOWANCE) 內建程序指定的值。
SYSTEM	"ALIGNMENT _DEFAULT"	"CURSOR_ HORIZONTAL", "CURSOR_ VERTICAL", 或 "SCROLL"	關鍵字	特定內建程序的現用 ALIGNMENT _DEFAULT 是關鍵字 CHARACTERS 或 BYTES。
SYSTEM	"UNIT_ DEFAULT"	"SUBSTR", "INDEX", "LENGTH", 或 "CURRENT_ OFFSET"	關鍵字	特定內建程序的現用UNIT_DEFAULT 是關鍵字 CHARACTERS 或 BYTES。
SCREEN	"DISPLAY_ CURSOR"		關鍵字	當顯示游標模態設定爲 ON 時，會傳回 ON,當顯示游標模態並不設定爲 ON，或者再度變成 OFF 時，則會傳回 OFF。

示錯訊息	TPU$_BADREQUEST	WARNING	第二參數的資料型態與第一 參數資料型態不一致。
	TPU$_BADKEY	WARNING	錯誤的關鍵字值或無法辨識的資料型態被傳給第一參數
	TPU$_NOCURRENTBUF	WARNING	現用緩衝區未定義。
	TPU$_NOKEYMAP	WARNING	鍵圖未定義。
	TPU$_NOKEYMAPLIST	WARNING	鍵圖清單未定義。
	TPU$_INVPARAM	ERROR	參數資料型態錯誤。
	TPU$_NEEDTOASSIGN	ERROR	GET_INFO 內建程序只能用在賦值述句右側。
	TPU$_NOBREAKPOINT	WARNING	字串常數只能用在中斷點之後。
	TPU$_NONAMES	WARNING	沒有一個名稱符合要求。
	TPU$_TOOFEW	ERROR	參數太少。
	TPU$_TOOMANY	ERROR	參數太多。
	TPU$_UNKKEYWORD	ERROR	參數使用錯誤的關鍵字。

範例

```
buffer_byte_offset := GET_INFO(CURRENT_BUFFER,"byte_offset");
```

此賦值述句將由一列的開始算出到緩衝區中游標所在位置的位移值，並把此值指定給 buffer_byte_offset 變數。

INDEX

此內建程序能算出字串中某個字元或子字串的位置，並將該位置傳回。

格式 **integer := INDEX** *(string1,string2**[**,keyword**]**)*

參數

string1
是一個含有您想尋找字元或子字串的字串。

string2
您想要在 string1 內找尋的字元或子字串。

keyword
您可指定下列任何關鍵字:

- CHARACTERS - 傳回從字串的啟始到子字串位置的字元數。
- BYTES - 傳回從字串的啟始到子字串位置的位元組數。

若您不指定關鍵字，既定爲 CHARACTERS。您也可以用 SET(UNIT_DEFAULT)內建程序來改變既定值。請參閱 SET(UNIT_DEFAULT) 的詳細說明。

描述

有關 INDEX 內建式程序的基本描述，請參閱 《DEC Text Processing Utility Reference Manual》。

您在 HTPU 中，可以用加入 keyword 參數的方法，選擇得到字元位置的數目以位元組或字元數來表示。

示錯訊息	TPU$_TOOFEW	ERROR	參數太少。
	TPU$_TOOMANY	ERROR	參數太多。
	TPU$_INVPARAM	ERROR	參數資料型態錯誤。
	TPU$_NEEDTOASSIGN	ERROR	INDEX 只能在賦值述句右側。

範例

```
1     char_loc := INDEX ('１２３４５６７','６７',CHARACTERS)
```

此賦值述句將整數值 6 傳回給 char_loc 變數。 因爲子字串 '６７' 位在 '１２３４５６７' 字串的第六個字元。

```
2     loc := INDEX ('１２３４５６７','６７',BYTES)
```

此賦值述句將整數值 11 指定給 loc 變數。因爲子字串 '６７'位在 '１２３４５６７' 字串的第 11 個位元組。

IS_CLASS

此內建程序能辨別您指定的字串之字元種類是否屬於由關鍵字指定的種類。若符合，IS_CLASS 傳回 1，否則傳回 0。

格式 **integer := IS_CLASS** *(string,keyword)*

參數

string
是一個放在引號中的字串，一個表示字串常數的變數名稱，或是指到一個字串的式子。若字串長度超過一個字元，則只辨識第一個字元。

keyword
關鍵字代表一種字元種類。您可指定下列任何一個：

- ASCII_CHAR - ASCII 字元 (半形字元)。
- NON_HANYU - 關於此關鍵字的說明，請參閱本手冊的圖 1-1 以及圖 1-2。
- HANYU - 請參閱本手冊中的圖 1-1 和圖 1-2，以便獲知有關這個關鍵字的描述。

請注意，所有四位元組的字元都被歸爲 HANYU 類別。

描述 若輸入字串的第一個字元屬於 keyword 參數所指定的種類，則本內建程序傳回 1，否則傳回 0。

示錯訊息	TPU$_TOOFEW	ERROR	參數太少。
	TPU$_TOOMANY	ERROR	參數太多。
	TPU$_INVPARAM	ERROR	參數資料型態錯誤。
	TPU$_NEEDTOASSIGN	ERROR	IS_CLASS 只能在賦值述句右側。

JUSTIFY

此內建程序將重新格式化指定緩衝區或範圍中的文字，以致每一行文字都可以向左、向右或者同時向左右內縮。

格式

JUSTIFY *(* { *buffer* / *range* } ***[****,keyword****[****,integer1****[****,integer2 ****[****,integer3****]]]])***

參數

buffer
此緩衝區中的文字是您想要對齊的。

range
此範圍中的文字是您想要對齊的。

keyword
關鍵字可以使用下列各值：

- JUSTIFY_LEFT - 調整第一行以外的每一行，以便都能從 integer1 或左界開始。此爲既定值。
- JUSTIFY_RIGHT - 調整最後一行以外的每一行，以便都能結束於 integer1 或右界。
- JUSTIFY_BOTH - 調整第一行以外的每一行以便都能從 integer1或左界開始，同時，調整最後一行以外的每一行以便都能結束於 integer2 或右界。

integer1
此爲左界的值。左界的值必須大於 1 並小於右界。該值既定爲緩衝區的左界。

integer2
此爲右界的值。右界的值必須大於左界，但不能超過緩衝區的最大記錄大小。該值既定爲緩衝區的右界。

integer3
此爲第一行內縮的值。這個值修改了第一行的左界。它可能是正數或負數。然而將這個值加上左界後所得的值必須大於 1 並小於右界。這個值既定爲 "0"。

描述

JUSTIFY 將單一位元組和多位元組字元之間的空格（%X20）與介面視爲唯一的文字分離字元。最初它會呼叫邊界值爲零的 FILL 內建程序，以便填滿範圍或緩衝區，來執行記錄的邊界設定，以及記錄的分割與附加。

除非該行爲正要對齊的緩衝區或範圍的第一行，否則 JUSTIFY 將每一行的左界設定爲對齊左界。在這種情況下，JUSTIFY 會將該行的左界設定爲對齊左界並加上第一行內縮。然而，如果您正在對齊一個範圍，而此範圍並不是從一行的行首開始，則 JUSTIFY 並不會改變該行的左界。因此，您就必須指定第一行內縮爲零。

如果只需要左邊對齊，則 JUSTIFY 內建程序會在呼叫 FILL 之後傳回。如果需要右邊對齊，則 JUSTIFY 會在每一項記錄的開頭插入空格。然而，若指定了一個範圍，且開始的位移不爲零，則這些空格會被插入該範圍第一項記錄的開始位移。

如果左右兩邊都需要對齊，則空格會被插入在那些空格，或是 ASCII 與多位元組字元之間的位置。倘若找不到這些位置，則不會插入任何空格，於是範圍/緩衝區都不會向右對齊。假如要對齊一個範圍，則只能在開始處之後插入空格。請注意，最後一行將不會向右對齊，而可能超過右界。

請注意，JUSTIFY 內建程序會將空白字元插入緩衝區以便進行對齊工作，並且在對齊之前並不會執行壓縮，因此以不同的邊界設定值連續呼叫 JUSTIFY 並不會重新安排文字，這是因爲先前插入的空格會被視爲資料而予以處理。所以，如果使用者想

要改變對齊的邊界，則應於再次呼叫 JUSTIFY 之前要求 EDIT 進行 COMPRESS。

JUSTUFY 在完成對齊的工作之後，會將游標置於已對齊文字的最後。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_ARGMISMATCH	ERROR	FILL 的某個參數爲錯誤的資料型態。
TPU$_BADMARGINS	WARNING	您所指定的填滿邊界中有一個是錯誤的。
TPU$_INVPARAM	ERROR	FILL 的某個參數爲錯誤的資料型態。
TPU$_NOTMODIFIABLE	WARNING	您不能改變不可修改緩衝區中的文字。
TPU$_CONTROLC	ERROR	您在執行 FILL 時按到了CTRL/C 鍵。
TPU$_NOCACHE	ERROR	沒有足夠的記憶體可配置新的高速緩衝記憶體。

範例

```
JUSTIFY (current_buffer, JUSTIFY_BOTH, 1, 60, 10)
```

上面的敘述將現用緩衝區的左界和右界都予以對齊。第一列從第 1 + 10 （＝11）欄開始，而最後一行並不向右對齊。

KEY_NAME

此內建程序能傳回一個鍵或數個鍵組合的 HTPU 關鍵字。

格式

keyword2 := KEY_NAME ({ *integer* / *key_name* / *string* } **[**,*keyword*]**[**, { *FUNCTION* / *KEYPAD* } **]** **)**

參數

integer
是一個整數，此整數爲一個 關鍵字鍵之整數表示；或是某一個 0 到 255 的值，HTPU 解譯成一個在 DEC 多國字元集 (DEC Multinational Character Set) 中字元的值；或是一個被解譯成 DEC SICGCC 字元碼的整數。

key_name
一個 HTPU 鍵名的關鍵字。

string
所謂字串就是主要鍵盤上的一個鍵的值。

keyword
這個關鍵字修訂指定的鍵，並可使用下列各值：

- SHIFT_KEY - 指定所建立的鍵名應包括 HTPU SHIFT 鍵，同樣的情形也適用於 HEVE 的 GOLD 鍵。多位元組字元只接受此關鍵字修正器。當用於多位元組字元時，其他關鍵字的值會被忽略。

- SHIFT_MODIFIED - 指定所建立的鍵名應包括鍵盤上標示SHIFT的鍵。當應用於多位元組字元時，該值會被忽略。
- ALT_MODIFIED - 指定所建立的鍵名應包括 ALT 鍵。當應用於多位元組字元時，該值會被忽略。
- Ctrl_MODIFIED - 指定所建立的鍵名應包括 Ctrl 鍵。當應用於多位元組字元時，該值會被忽略。
- HELP_MODIFIED - 指定所建立的鍵名應包括 Help 鍵。當應用於多位元組字元時，該值會被忽略。

FUNCTION
此參數指定鍵名爲一個功能鍵。

KEYPAD
此參數表示鍵名屬於副鍵組鍵。

描述

有關 KEY_NAME 內建式程序的基本描述，請參閱 《DEC Text Processing Utility Reference Manual》。

您可以爲參數 string 指定一個多位元組字元。您也可以使用多位元組字元來定義某個鍵。

示錯訊息

TPU$_INCKWDCOM	WARNING	不連續的關鍵字組合
TPU$_MUSTBEONE	WARNING	字串必須爲一個字元長
TPU$_NOTDEFINABLE	WARNING	第二個參數對於一個鍵而言並不是有效的參考

TPU$_NEEDTOASSIGN	ERROR	KEY_NAME 呼叫必須位於指定敘述的右邊
TPU$_ARGMISMATCH	ERROR	傳給 KEY_NAME 內建程序的資料爲錯誤的型式
TPU$_BADKEY	ERROR	KEY_NAME 可以接受 SHIFT_KEY、或 KEYPAD 做爲關鍵字參數
TPU$_TOOFEW	ERROR	傳給 KEY_NAME 內建程序的參數太少
TPU$_TOOMANY	ERROR	傳給 KEY_NAME 內建程序的參數太多

LENGTH

此內建程序會傳回一個整數值，其表示某字串或範圍中字元或位元組之數目。

格式

integer := LENGTH *(* { *buffer* / *range* / *string* } ***[****,keyword****]****)*

參數

buffer
您想知道其長度的緩衝區名稱。注意，若指定一個緩衝區，則列結尾記號不算在內。

range
您想知道其長度的範圍名稱。注意，若指定一個範圍，則列結尾記號不算在內。

string
您想知道長度的字串。

keyword
您可指定下列關鍵字：

- CHARACTERS - 傳回字串長度爲字元數。
- BYTES - 傳回字串長度爲位元組數。

若不指定關鍵字, 既定爲 CHARACTERS。您也可用 SET(UNIT_DEFAULT) 內建程序來改變既定值。參閱 SET(UNIT_DEFAULT) 的說明。

描述

有關 LENGTH 內建式程序的基本描述，請參閱 《DEC Text Processing Utility Reference Manual》。

您在 HTPU 中，可以增加 keyword 參數來選擇以字元單位或以位元組單位得到字串長度。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_NEEDTOASSIGN	ERROR	LENGTH 只能在賦值述句右側。
TPU$_ARGMISMATCH	ERROR	參數的資料型態錯誤。
TPU$_CONTROLC	ERROR	您在 LENGTH 執行過程中按了 CTRL/C 鍵。

MARK

此內建程序能傳回一個在現用緩衝區中設定給現用字元位置的標記。您必須指定如何在螢幕上顯示標記。(NONE，REVERSE，BOLD，BLINK，或 UNDERLINE)。

格式

marker := MARK *(keyword* ***[*** *,* { *buffer* / *widow* } ***[,*** *integer1* ***[,*** *integer2* ***]]]***

參數

keyword
您必須使用下列關鍵字之一：

- BLINK - 指定標記顯示爲閃爍模式。
- BOLD - 指定標記顯示爲粗體模式。
- FREE_CURSOR - 指定此標記爲自由的標記。（即，標記不以一個字元爲界)。指定 FREE_CURSOR 參數，並不會建立一個自由標記， 除非游標落在一列啟始之前，一列結尾之後，標記的中間，或當執行 MARK(FREE_CURSOR) 時，位在緩衝區底部之下。自由標記沒有影像屬性。
- NONE - 指定此標記無影像屬性。
- RESERVE - 指定標記顯示爲凸顯模式。
- UNDERLINE - 指定標記底線模式。

buffer
標記符號所在的緩衝區。根據既定值，HTPU 將標記符號置於現用緩衝區中。

window
對應至標記符號所在緩衝區之視窗。只有當視窗對應至緩衝區時，您才可以指定視窗變數。根據既定值，HTPU 將標記符號置於現用緩衝區中。

integer1
代表標記符號所在螢幕欄位的一個整數。您所指定的整數可以從 1 到 32,767。既定值是將標記符號置於與現用螢幕欄位對應的緩衝區中。

integer2
代表標記符號所在緩衝區中該記錄的一個整數。既定值是將標記符號置於現用記錄中。

描述

有關 MARK 內建程序的基本描述，請參閱《DEC Text Processing Utility Reference Manual》。

標記符號是設定在字元的邊界。因此，如果游標位於多位元組字元上，則即使現用編輯點是位在多位元組字元的第二欄，設定標記符號仍會導致標記符號被設定在字元的第一欄。

以影像屬性標記多位元組字元時，將導致影像模式出現於整個多位元組字元上。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_NEEDTOASSIGN	ERROR	MARK 只能在賦值述句右側。
TPU$_NOCURRENTBUF	WARNING	您必須在緩衝區內設定標記。
TPU$_INVPARAM	ERROR	指定的一個或多個參數型態錯誤。

TPU$_BADKEY	ERROR	關鍵字必須是 NONE,BOLD,BLINK, REVERSE, UNDERLINE 或 FREE_CURSOR。
TPU$_UNKKEYWORD	ERROR	您指定了一個不知名的關鍵字。
TPU$_INSVIRMEM	FATAL	記憶體空間不足以建立標記。

READ_CHAR

此內建程序將傳回從鍵盤上輸入的下一個字元的字串。

格式 **string := READ_CHAR**

回轉值 字串型態的變數，此變數含一個從鍵盤輸入的字元。

描述 請參閱 《DEC Text Processing Utility Reference Manual》 關於 READ_CHAR 內建程序的描述。

在 HTPU 中，READ_CHAR 程序已經增強而能夠輸入多位元組字元。

示錯訊息

TPU$_TOOMANY	ERROR	參數太多。
TPU$_NEEDTOASSIGN	ERROR	READ_CHAR 只能在賦值述句右側。

READ_KEY

此內建程序等待您按鍵，然後傳回此鍵的名稱。READ_KEY 也能處理分離序列 (escape sequences) 及控制字元。

格式 **keyword := READ_KEY**

回轉值 含有剛才按鍵名稱的回轉值。

描述 有關 READ_KEY 內建式程序的基本描述，請參閱 《DEC Text Processing Utility Reference Manual》。

READ_KEY 已經增強爲能夠輸入多位元組字元。如果輸入的字元爲多位元組字元，則該字元將有一個關鍵字被傳回。

SCROLL

此內建程序依您指定的移動列數，使緩衝區的文字在畫面上作往上或往下的方向移動。

格式 **[integer2 :=]SCROLL** *(window[,integer1[,keyword]])*

參數

window
此視窗可定位您想要捲動文字的緩衝區。

integer1
一個帶正負 (+/-) 號的整數值，它表示您想捲動的列數。若指定正整數，視窗內的文稿會由畫面底部向上捲動。若指定負整數，視窗內的文稿會由畫面頂端向下捲動。若指定 0，則不捲動。

此參數可省略。若您省略此參數，則文稿會繼續捲動直到緩衝區的邊界，或除非您按下任何一個鍵。若緩衝區的導向是 FORWARD，則文稿向緩衝區的尾部捲動。若緩衝區的導向爲 REVERSE，則文稿向緩衝區的開頭捲動。若您按的鍵是命令鍵，則停止捲動且 HTPU 執行此鍵代表的命令。

keyword
您可指定下列關鍵字：

- CHARACTERS - 捲動之後，如果游標位於多位元組字元的非第一個位元組上，則 CHARACTERS 會讓游標回到第一欄。

- BYTES - 捲動之後，游標會停留在已回到游標位置而未做任何轉換的字元的位元上。然而，當啟用顯示游標模態時，即使內部游標位置可能位於多位元組字元的非第一個位元上，螢幕游標仍然位於多位元組字元的字元邊界上。有關細節，請參閱 SET(DISPLAY_CURSOR) 命令的描述。

既定爲 BYTES。您可以用 SET(ALIGNMENT_DEFAULT) 的內建程序來改變既定值。若需要有關的細節，請參閱 SET(ALIGNMENT_DEFAULT) 的描述。

回轉值　回轉值表示 SCROLL 實際捲動的列數。

描述　有關 SCROLL 內建式程序的基本描述，請參閱《DEC Text Processing Utility Reference Manual》。

HTPU 已經被增強成可以將 BYTES 或 CHARACTERS 關鍵字指定爲第三個參數。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_INVPARAM	ERROR	指定的參數有一個或多個型態錯誤。
TPU$_CONTROLC	ERROR	您按下了 CTRL/C 停止捲動。
TPU$_WINDNOTMAPPED	WARNING	企圖對一個未映射的視窗作捲動。

SEARCH

此內建程序會在緩衝區或範圍中找尋字元的特殊排列方式，並傳回包含那些字元的範圍。

格式

[range2 :=]SEARCH *(pattern,* { *FORWARD* / *REVERSE* } *[,* { *EXACT* / *NO_EXACT* / *integer* } *[,* { *buffer* / *range* } *]])*

參數

pattern
請參閱《DEC Text Processing Utility Reference Manual》。

FORWARD
指出向前的搜尋。

REVERSE
指出相反方向的搜尋。

EXACT
指出 SEARCH 正試圖比對的字元，必須與用來作爲 SEARCH 第一個參數類型中的字元有相同大小寫字體和字形大小（全形或半形）。

NO_EXACT
指出 SEARCH 正試圖比對的字元，並不需要具有同樣的大小寫字體或字形大小（全形和半形的字母字元被視爲是相同的）。此爲既定值。

integer
指出如果您想比對一種屬性而忽略另一種屬性時，SEARCH 應如何處理大小寫字體和字形大小（全形/半形）的資訊。迪吉多電腦公司建議您可以使用預先定義的常數來指定此整數。預先

定義的常數如下：

- TPU$K_SEARCH_CASE - 等於整數 1。它指定搜尋應符合第一個參數的大小寫字體，但毋需理會第一個參數的字形大小（全形/半形）。

- TPU$K_SEARCH_FORM - 等於整數 4。它指定 HTPU 中的搜尋應符合第一個參數的字形大小，但毋需理會第一個參數的字元大小寫字體。

若想合併數個關鍵字的效果，您只需將預先定義的整數變數值相加即可。例如，欲使字形和大小寫都能完全符合，請指定 TPU$K_SEARCH_CASE + TPU$K_SEARCH_FORM 這個值作爲 "integer" 參數。

請注意，由於 HTPU 對於讀音標記並不具有感應能力，所以如果單獨指定 TPU$K_SEARCH_DIACRITICAL，或者是同時指定了 TPU$K_SEARCH_CASE， TPU$K_SEARCH_DIACRITICAL 會被忽略，而如果同時又指定 TPU$K_SEARCH_FORM，則會出現錯誤，因而產生 TPU$_MAXVALUE 的信號。

buffer
欲搜尋的緩衝區。

range
欲搜尋的範圍。

回轉值 回轉一個包含與指定類型符合的文字的範圍。

描述 有關 SEARCH 內建式程序的基本描述，請參閱 《DEC Text Processing Utility Reference Manual》。

HTPU 改進 SEARCH 內建式程序，使其對於欲搜尋類型的大小寫字體和字形大小產生選擇性的感應能力。

示錯訊息	TPU$_STRNOTFOUND	WARNING	搜尋字串或類型的工作失敗了。
	TPU$_TOOFEW	ERROR	SEARCH 至少需要兩個參數。
	TPU$_TOOMANY	ERROR	SEARCH 無法接受超過四個參數。
	TPU$_ARGMISMATCH	ERROR	SEARCH 有一個參數是錯誤的型態。
	TPU$_INVPARAM	ERROR	SEARCH 有一個參數是錯誤的型態。
	TPU$_BADKEY	WARNING	您爲 SEARCH 指定了一個錯誤的關鍵字。
	TPU$_MINVALUE	WARNING	SEARCH 的整數參數必需大於或等於 1。
	TPU$_MAXVALUE	WARNING	SEARCH 的整數參數必需小於或等於 5。
	TPU$_NOCURRENTBUF	ERROR	如果您並未指定欲搜尋的緩衝區或範圍，則必需在搜尋之前指出一個緩衝區。
	TPU$_CONTROLC	ERROR	您在執行 SEARCH 時按下了 Ctrl/C。
	TPU$_ILLPATA	ERROR	給 SEARCH 的類型變數中有未被定義在現用的內文中的變數。

SELECT

此內建程序能在現用緩衝區的編輯點上設定一個標記，且傳回此標記。您必須指定標記在畫面上顯示的方式（NONE，REVERSE，BOLD，BLINK，或 UNDERLINE）。

由 SELECT 傳回的標記表示在選擇的範圍內第一個字元的位置。您指定給此標記的影像屬性適用於選擇範圍內的所有字元。關於建立一個選擇範圍之詳細資料，請參閱 SELECT_RANGE 內建程序的說明。

格式

marker := SELECT *(keyword)*

參數

keyword
此關鍵字指明被傳回的標記如何顯示在畫面上。您必須使用下列關鍵字之一：

- BLINK - 指定標記以閃爍模式顯示。
- BOLD - 指定標記以粗體模式顯示。
- NONE - 指定沒有屬性適用此標記。
- REVERSE - 指定標記以凸顯模式顯示。
- UNDERLINE - 指定標記以底線模式顯示。

描述

SELECT 傳回一個特別的標記，建立一個選擇範圍的開始。當執行 內建式程序 SELECT 時，標記符號會位於編輯點的字元位置上。如果編輯點並不是位於多位元字元的第一欄，則標記符號的位置會介於前一個字元與多位元組字元的第一欄之間。

示錯訊息	TPU$_ONESELECT	WARNING	SECECT 已在此緩衝區內動作了。
	TPU$_TOOFEW	ERROR	參數太少。
	TPU$_TOOMANY	ERROR	參數太多。
	TPU$_NEEDTOASSIGN	ERROR	SELECT 只能用在賦值述句的右側。
	TPU$_NOCURRENTBUF	WARNING	使用 SELECT 之前必須先被移置在緩衝區中。
	TPU$_INVPARAMM	ERROR	指定的參數有一個或多個型態不對。
	TPU$_BADKEY	ERROR	指定錯誤的關鍵字給 SELECT。

SELECT_RANGE

此內建程序能傳回一個範圍，此範圍包含位在由 SELECT 設定的標記及現用編輯位置之間的所有字元。執行 SELECT_RANGE 時，選擇的範圍的最後位置不包含現用字元。

格式 **range := SELECT_RANGE**

描述 有關 SELECT_RANGE 內建程序的基本描述，請參閱 《DEC Text Processing Utility Reference Manual》。

在執行 SELECT_RANGE 時，不論現用編輯點是否位於多位元組字元的第一欄，其作用都是一樣的。

示錯訊息

TPU$_NOSELECT	WARNING	在現用緩衝區中沒有待命的選擇範圍。
TPU$_SELRANGEZERO	WARNING	選擇的範圍未包含任何選定的字元。
TPU$_TOOMANY	ERROR	字元參數太多。
TPU$_NEEDTOASSIGN	ERROR	SELECT_RANGE只能用在賦值述句的右側。
TPU$_NOCURRENTBUF	WARNING	沒有現用的緩衝區。

SET(ALIGNMENT_DEFAULT)

格式 **SET** *(ALIGNMENT_DEFAULT, string,*
{ *BYTES* | *CHARACTERS* }*)*

參數

ALIGNMENT_DEFAULT
這個關鍵字表示 SET 界定此游標排列特性給特定的內建程序。

string
是一個字串，它包含一個被設定 ALIGNMENT_DEFAULT 的內建程序名稱。有效的內建程序爲 CURSOR_HORIZONTAL、CURSOR_VERTICAL和 SCROLL。

BYTES
是一個關鍵字，它指定游標以位元組爲單位而調整。

CHARACTERS
是一個關鍵字，它指定游標以字元爲單位而調整。

描述

SET(ALIGNMENT_DEFAULT) 指定是否要調整游標，使其停在多位元組字元的第一個位元組上。CURSOR_HORIZONTAL、CURSOR_VERTICAL以及SCROLL中，ALIGNMENT_DEFAULT 是 BYTES，也就是說游標在這些內建程序中不會被調整。

請參閱 CURSOR_HORIZONTAL、CURSOR_VERTICAL 以及 SCROLL。

示錯訊息	TPU$_TOOFEW	ERROR	參數太少。
	TPU$_TOOMANY	ERROR	參數太多。
	TPU$_INVPARAM	ERROR	引數型態錯誤。

SET(DISPLAY_CURSOR)

格式

[keyword :=]SET *(DISPLAY_CURSOR,* { *ON* / *OFF* } *)*

參數

DISPLAY_CURSOR
指出 SET 將改變 HTPU 顯示-游標模態的一個關鍵字。

ON
此爲被指定將顯示游標模態轉爲 ON 的關鍵字。

OFF
此爲被指定將顯示游標模態轉爲 OFF 的關鍵字。

回轉值 回轉值指出應用此內建程序之前，HTPU 的顯示游標模態爲何。

描述 此內建程序將 HTPU 的顯示游標模態轉爲 ON 或 OFF，並選擇性獲取顯示-游標模態的現用設定值。

當顯示游標模態爲 ON 時，如果游標正位於多位元組字元上，則您在螢幕上所看到的游標 (稱爲螢幕游標) 與儲存於 HTPU 中的游標 (稱爲內部游標) 會有所不同。

例如，如果內部游標正位於多位元組字元的第二個位元上，那麼當顯示游標模態爲 OFF 時，螢幕游標會位於字元的第二欄；而當顯示游標模態爲 ON 時，則螢幕游標會位於字元的第一欄。

顯示游標模態讓螢幕游標恆位於字元界上。請注意，當顯示游標模態爲 ON，而內部游標位於多位元組字元的非第一個位元上時，則 HTPU 的插入或重打，將會如內部游標位於螢幕游標相同位置 -- 亦即恆位於字元界上 -- 的情況下進行。

在啟動 HTPU 時，顯示-游標模態既定爲 OFF，而在執行 HEVE 啟動程序時則設定爲 ON。

SET(FILL_NOT_BEGIN)

格式 **SET** *(FILL_NOT_BEGIN, string)*

參數

FILL_NOT_BEGIN
指出 SET 將定義執行 FILL 和 JUSTIFY 時，不能位於行首特別字元的一個關鍵字。

string
是一個字串, 它包含不能放在列首的例外字元。

描述

SET(FILL_NOT_BEGIN) 爲 FILL 和 JUSTIFY 內建程序指定 FILL_NOT_BEGIN 字元。FILL 和 JUSTIFY 決定應於何處分割一行，以致 FILL_NOT_BEGIN 字元並不會出現於行首。

請參閱 FILL、JUSTIFY、SET(FILL_NOT_END) 以及 SET(MARGIN_ALLOWANCE)。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_INVPARAM	ERROR	引數型態錯誤。

SET(FILL_NOT_END)

格式 **SET** *(FILL_NOT_END, string)*

參數

FILL_NOT_END
指出 SET 將定義執行 FILL 和 JUSTIFY 時，不能位於行尾特別字元的一個關鍵字。

string
是一個字串, 它包含不能放在列結尾的例外字元。

描述

SET(FILL_NOT_END) 爲 FILL 和 JUSTIFY 內建程序指定 FILL_NOT_END 字元。FILL 和 JUSTIFY 決定應於何處分割一行，以致 FILL_NOT_END 字元並不會出現於行尾。

請參閱 FILL、JUSTIFY、SET(FILL_NOT_BEGIN) 以及 SET(MARGIN_ALLOWANCE)。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_INVPARAM	ERROR	引數型態錯誤。

SET(MARGIN_ALLOWANCE)

格式 **SET** *(MARGIN_ALLOWANCE, integer)*

參數

MARGIN_ALLOWANCE
這個關鍵字表示執行 FILL 期間，SET 要界定可以被擺在右界之外的例外字元數。

integer
可被擺在右界之外的例外字元數。

描述

SET(MARGIN_ALLOWANCE) 指定可被擺在右界之外的例外字元數。FILL 內建程序考慮例外字元（FILL_NOT_BEGIN 及 FILL_NOT_END 字元）並填滿此列。

既定值爲 0。當值爲 0 時，FILL_NOT_BEGIN 字元不能超過右界。爲了防止 FILL_NOT_BEGIN 字元出現在一列的開頭，這些字元會和其它字元一起移到下一列。因此，有可能有一列並沒有被填滿到右界。

參閱 FILL、SET(FILL_NOT_BEGIN) 及 SET(FILL_NOT_END)。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_INVPARAM	ERROR	引數型態錯誤。

SET(UNIT_DEFAULT)

格式

SET *(UNIT_DEFAULT, string,* { *BYTES* / *CHARACTERS* } *)*

參數

UNIT_DEFAULT
這個關鍵字指示 SET 界定內建程序的計數單位。

string
是一個字串，它包含將設定 UNIT_DEFAULT 的內建處理程序名稱。有效的內建程序爲 CURRENT_OFFSET、INDEX、LENGTH 和 SUBSTR。

BYTES
這個關鍵字指定計數單位爲位元組。

CHARACTERS
這個關鍵字指定計數單位爲字元。

描述

SET(UNIT_DEFAULT) 指定既定的計數單位爲位元組或字元。CURRENT_OFFSET、 INDEX、 LENGTH 及 SUBSTR 的 UNIT_DEFAULT 爲 CHARACTERS。

請參閱 CURRENT_OFFSET、INDEX、LENGTH 和 SUBSTR。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_INVPARAM	ERROR	引數型態錯誤。

SPLIT_LINE

此內建程序將現用列從現用的編輯位置分割成兩列。

格式 **SPLIT_LINE**

描述 有關 SPLIT_LINE 內建程序的基本描述，請參閱 《DEC Text Processing Utility Reference Manual》。

如果現用編輯點並不位於一個多位元組字元的第一個位元，則 SPLIT_LINE 會以好比編輯點就位於該字元的第一個位元一般來分割一行。SPLIT_LINE 執行之後，編輯位置移動到該字元的第一個位元組上。

示錯訊息

TPU$_NOCURRENTBUF	WARNING	您沒有定位在緩衝區中。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_NOTMODIFIABLE	WARNING	您不能修改一個不許修改的緩衝區。
TPU$_NOCACHE	ERROR	記憶體空間不夠配置新的高速存取記憶體。

SUBSTR

此內建程序能在一字串或範圍內建立一個子字串、並且傳回一個表示此子字串的字串。

格式

string1 := SUBSTR({ *buffer* / *range* / *string* } **,** *integer1,integer2***[,** *keyword***])**

參數

buffer
含有此子字串的緩衝區。

range
含有此子字串的範圍。

string
含有此子字串的字串。

integer1
子字串開始的字元或位元位置。第一個字元位置爲 1。

integer2
此子字串中包含的字元或位元組數。

keyword
您可使用下列關鍵字:

- CHARACTERS - 第二及第三個參數的單位是字元數。
- BYTES - 第二及第三個參數的單位是位元組數。

若您不指關鍵字，既定爲 CHARACTERS；也可以用 SET(UNIT_DEFAULT) 更改既定值。請參閱 SET(UNIT_DEFAULT) 的詳細說明。

描述

有關 SUBSTR 內建程序的基本描述，請參閱《DEC Text Processing Utility Reference Manual》。

如果您指定 BYTES 作爲關鍵字，則在包含多位元組字元的字串中的子字串會依據指定的位元數目而形成，而如果抽選發生於多位元組字元的中間時，在子字串中的部份多位元組字元將被空格所取代。

示錯訊息

TPU$_TOOFEW	ERROR	參數太少。
TPU$_TOOMANY	ERROR	參數太多。
TPU$_NEEDTOASSIGN	ERROR	SUBSTR 只能用在賦值述句的右側。
TPU$_INVPARAM	ERROR	指定的參數有一個或多個型態不對。
TPU$_ARGMISMATCH	ERROR	SUBSTR 中的一個參數型態錯誤。
TPU$_STRTOOLARGE	ERROR	您所指定的長度超過所允許 的字串最大長度。

第 3 章

新增及修訂的 HEVE 命令

本章包括新增及修訂的 HEVE 命令參考資料。有些命令是由 EVE (Extensible Versatile Editor）改良而成。本章應當作爲 《Extensible Versatile Editor Reference Manual》 的補充部份。

本章有關命令的描述是依字母順序排列。與中文字元的用法無關的命令，將不包含在本章中。有關細節，請您參閱 《Extensible Versatile Editor Reference Manual》。請注意，一些並未納入本章的命令也可以處理本地語言字元。對那些命令而言，處理本地語言字元的情形與處理 ASCII 字元是一樣的。有關如何使用 HEVE 命令來執行編輯的描述，請參閱 《HEVE 使用者手冊》。

下表是本章包含的命令名稱。

表 3-1 新增及修訂的 HEVE 命令

命令
DELETE
DRAW BOX
DRAW LINE
ERASE CHARACTER
ERASE PREVIOUS WORD
ERASE WORD
EXTEND HEVE
EXTEND HTPU
FILL/FILL PARAGRAPH/FILL RANGE
FIND
FULLFORM WORD
HALFFORM WORD
HTPU

表 3-1 新增及修訂的 **HEVE** 命令 (續)

命令
LEFT ADJUST
LEFT INDENT
LOWERCASE WORD
MOVE BY WORD
MOVE DOWN/UP
MOVE LEFT/RIGHT
REPLACE
RIGHT ADJUST
RIGHT INDENT
SAVE EXTENDED HEVE
SAVE EXTENDED HTPU
SET TABS
SET [NO]DISPLAY CURSOR
SET FIND CASE [NO]EXACT
SET FIND EXACT
SET FIND FORM [NO]EXACT
SET FIND GENERAL
SPECIAL INSERT
SYMBOL
UPPERCASE WORD

在某些受支援的終端機上，有些命令界定在某些字鍵上。在每個命令格式描述的章節裡，我們列出這些定義的字鍵。例如，VT300/VT200 欄底下的資訊指出在 VT300 和 VT200 系列終端機上某個命令的定義字鍵。系統支援的 VT382 中文終端機歸類於 VT300 系列終端機。

DELETE

格式 **DELETE**

VT300,VT200	VT100
<x]	*DELETE*

描述 刪除游標左邊的字元。於插入模態時，該行的其餘部份會向左移動一個字元以消除空格。而在重打模態時，刪除掉的字元則會被與字元的螢幕欄寬相等的空格數所取代，亦即，一個位元組的字元需要一個欄位，兩個位元組的字元需要兩個欄位等。在行首處，這個功能刪除了折行功用，並將現行行移往前一行。

如果要恢復已刪除的資料，請使用 RESTORE CHARACTER 命令。

DRAW BOX

格式 **DRAW BOX**

描述 引動 HEVE 畫框功能。此與使用 DRAW LINE 命令，然後藉由按下 REMOVE 而輸入畫框模態是一樣的。請參閱 DRAW LINE 以獲取相關細節。

DRAW LINE

格式 DRAW LINE

描述 啟動 HEVE 的畫線及畫框功能。這項功能提供一個簡便的使用者界面。您在移動游標的時候，線及框都會顯示出來。您可以用↑、↓、→、← 四個鍵來控制游標的移動。簡單的開關鍵可以控制畫線的功能。當按下控制鍵的時候，在狀態列的繪圖狀態將被更新。使用這項功能時，請遵循下列步驟：

1 按下 <Do> 鍵，得到命令提示號。

2 在命令提示號下，打入 DRAW LINE。HEVE 進入畫線模態。

3 在畫線模態下，您可以用↑、↓、→、← 四個鍵來移動游標而畫線或消除已畫的線。表 3-2 列出所有與畫線有關的控制鍵。

- 按下 <Select> 鍵選擇畫線模態或游標移動模態。在游標移動模態下，游標可以在不影響畫面資料的情況下自由地移動。
- 按下 <Insert Here> 鍵來選擇 Line 或 Erase Line。在 Erase Line 的模態下，當游標移動經過畫線字元時會消去已畫好的線。
- 按下 <Remove>鍵進入畫框模態。在畫框模態下，控制鍵的功能將與在畫線模態中不同。
- 按下 <Return>，離開畫線功能。

4 在畫框模態下，您可以用↑、↓、→、← 四個鍵來移動游標而畫框。表 3-3 列出所有與畫框有關的控制鍵。

- 按下 <Select> 鍵離開畫框模態，返回畫線模態。
- 按下 <Remove> 鍵取消畫框模態，返回畫線模態。
- 按下 <Return> 鍵同時離開畫線與畫框模態。

按下 <Do> 鍵也可以離開畫線和畫框模態。

畫線字元一定與畫面的奇數行對齊。每個畫線字元會佔據畫面兩行。

以下是畫線和畫框模態下的控制鍵摘要：

表 3-2 畫線模態下的控制鍵

功能	開關鍵
畫線/移動游標	<Select> 或 KP1
線/消除線	<Insert Here> 或 KP2
進入畫框模態	<Return> 或 KP3
離開畫線模態	<Return>

表 3-3 畫框模態下的控制鍵

功能	開關鍵
離開畫框模態	<Select> 或 KP1
取消畫框模態	<Remove> 或 KP3
同時離開畫框和畫線模態	<Return>

請參閱 《HEVE 使用者手冊》 第五章中所提到有關如何使用 DRAW LINE 和 DRAW BOX 的範例。

ERASE CHARACTER

格式 ERASE CHARACTER

描述

刪除游標所在的字元。於插入模態時，該行的其餘部份會向左移動一個字元以消除空格。而在重打模態時，刪除掉的字元則會被與字元的螢幕欄寬相等的空格數所取代，亦即，一個位元組的字元需要一個空格，兩個位元組的字元需要兩個空格等。在行尾處，這個功能刪除了換行鍵功用，並將下一行移至現用行。

如果要恢復已經刪除的資料，請使用 RESTORE CHARACTER 命令。

ERASE PREVIOUS WORD

格式 **ERASE PREVIOUS WORD**

描述 抹除前一個單字或是游標所在的單字。如果游標介於兩個單字間，或者位於一個單字的開頭，這個命令將抹去前一個單字。如果游標位於一個單字的中間，則這個命令會抹除游標所在的單字和其後的空格。如果游標位於列首，這個命令將抹除前一列的換行功能，並使現用列向上移。

就多位元組的當地語言資料而言，這個功能的運作就如同單一位元組資料的處理情形一般，只不過「單字」的定義有所改變罷了 -- 當地語言字元的單字乃是一系列具有相同類別的字元（請參閱 HTPU 中的 CHARACTER_CLASS 部份）。當這個命令發現現用的字元與前一個字元不屬於同一類的時候，或是到達列尾時，它就假定其爲一個單字的結尾。

如果要恢復已經抹除的資料，請使用 RESTORE 或 RESTORE WORD 功能。

ERASE WORD

格式 **ERASE WORD**

VT300,VT200	VT100
CTRL/J	*CTRL/J* *Keypad COMMA*

描述 抹除游標所在的單字。如果游標介於兩個單字間，這個命令會抹除其後的一個單字和接續的空格。如果游標在列尾，則會抹除現用列的換行功能，並使下一列往上移動。

就多位元的當地語言資料而言，這個功能的運作就如同單一位元組資料的處理情形一般，只不過「單字」的定義有所改變罷了 -- 當地語言字元的單字乃是一系列具有相同類別的字元（請參閱 HTPU 中的 CHARACTER_CLASS 部份）。當這個命令發現現用的字元與前一個字元不屬於同一類的時候，或是到達列尾的時候，它就假定其爲一個單字的結尾。

如果要恢復已經抹除的資料，請使用 RESTORE 或 RESTORE WORD 命令。

EXTEND HEVE

格式 **EXTEND HEVE** *(procedure-name/*)*

描述 編譯一個或多個 HTPU 程序來擴充 HEVE。EXTEND HTPU 與這個命令是同義字。這個功能取代了 EVE 中的 EXTEND EVE。

參數

procedure-name
待編譯的 HEVE 程序名稱。您可以縮寫程序的名稱。如果有一個以上的名稱與其一致，HEVE 會把所有一致的名稱都顯示出來，並且重新執行 EXTEND HEVE 命令好讓您選擇所要的程序。

*****
通配符號告訴 HEVE 編譯緩衝區中所有的程序和述句。這個參數產生的結果與 EXTEND ALL命令一樣。

範例

```
Command: EXTEND HEVE USER_PROC
```

上例編譯現用緩衝區中的 USER_PROC 程序。

EXTEND HTPU

格式 **EXTEND HTPU** *(procedure-name/*)*

描述 編譯一個或多個 HTPU 程序來擴充 HEVE。EXTEND HEVE 與這個命令是同義字。這個功能取代了 EVE 中的 EXTEND TPU。

參數

procedure-name
待編譯的 HEVE 程序名稱。您可以縮寫程序的名稱。如果有一個以上的名稱與其一致，HEVE 會把一致的名稱通通顯示出來，並且重新執行 EXTEND HTPU 命令好讓您選擇所要的程序。

*
通配符號告訴 HEVE 編譯緩衝區中所有的程序和述句。這個參數產生與 EXTEND ALL 命令一樣的結果。

範例

```
Command: EXTEND HTPU USER_PROC
```

上例編譯現用緩衝區中的 USER_PROC 程序。

FILL/FILL PARAGRAPH/FILL RANGE

格式 FILL/FILL PARAGRAPH/FILL RANGE

描述

把現用段落或選取範圍內的文字根據緩衝區的邊界重排，如此可讓一列含有最多的字數。使用 FILL RANGE 之前，您必須先選取一個範圍。請遵循以下步驟:

- 您可以使用 SELECT 凸顯文字區域以供 FILL 使用。如果使用FILL RANGE 命令，一定要選取一個範圍。
- 使用 FILL 完成重排。原有的影像特質將消失，並且把游標移到這個範圍的結尾處。

如果您沒有選取任何文字，FILL 會使用現用所在的段落。段落由以下的方式區分：

- 空白列。
- 緩衝區的頭或尾。
- 斷頁。
- DIGITAL 標準的 Runoff 命令。

FILL 會將段落或選取範圍開始與結尾中的標號（tab）和空格(space) 移開，卻不影響文字之間的標號和空格。

在處理多位元組字元時，如果只有一個空欄且下一個字元爲多位元組，則 FILL 會於行尾留一個空格。對於 FILL 而言，一個二位元組的空格字元並非真正的空格，所以 FILL 不會使用它來分隔文字。

FIND

格式 **FIND** *(search-string)*

VT300,VT200	VT100
Find	*PF1*

描述 在現用緩衝區中尋找特定的字串。如果找到了這個字串，HEVE 會凸顯那些文字，並且把游標移到該字串的開頭。

要尋找上一個使用過的字串，按 <Find> 鍵兩次。如果您按一次 <Find> 鍵然後再按 <Return> 鍵，則 HEVE 將不會執行 <Find> 命令。

按下 <Return> 鍵往目前的方向開始尋找。如果要往某個特定的方向尋找，請按方向設定鍵。例如，當您按下 <F11> 鍵終止這項功能，"FIND" 會以與緩衝區中目前方向相反的方向開始。如果您按下一個界定爲 FORWARD 鍵，則無論現用緩衝區中的方向爲何，都會向前搜尋。

這項功能首先以您指明的方向在緩衝區中搜尋，如果找不到，<Find> 會往反方向搜尋；如果找到了，您將被問到是否要到該處。如果要去請按 <Return>, 若打入 NO 再按 <Return> 則會結束搜尋。

與選取範圍相似，找到的字串會被凸顯。您可以對其使用 REMOVE、STORE TEXT、LOWERCASE WORD 或其它處理選取範圍的功能。將游標移離凸顯的字串會取消其影像特質。

FIND 功能被增強爲能找尋包含多位元組字元的字串。您的搜尋字串可以包含中文字元以及單一位元組的 ASCII 字元。甚至，視 SET FIND 命令的使用情形而定，當您爲搜尋字串適當

地指定如下 表所示的大小寫字體和字形大小時，可以於執行 FIND 命令之際，啟用或取消大小寫字體和字形大小的感應功能。

表 3-4 FIND 命令的大小寫字體和字形大小感應

正確的大小寫形式	正確的形式	搜尋字串	結果
否	否	全部爲小寫	對大小寫字體和字形大小均不感應
是	否	任何形式	能感應大小寫字體，但無法感應字形大小
否	是	全部爲小寫	無法感應大小寫字體，但能感應字形大小
是	是	任何形式	對大小寫字體和字形大小均能感應

請參閱有關 SET FIND 命令的描述。

參數

search-string

意指您想尋找的文字字串。如果您想比對字母而不管其大小寫字體和字形大小，則請使用小寫的 ASCII 字母。而如果想比對正確大小寫字體和字形大小的字串，則請指定某些非小寫 ASCII 的字母，或請使用 SET FIND 命令。有關細節，請參閱 SET FIND 命令。

FULLFORM WORD

格式 FULLFORM WORD

描述 讓現用文字或凸顯文字中所有單一位元組的 ASCII 字元都成爲全形字元。欲使用這個功能，則請：

- 選擇性地使用 <Select> 鍵或 <Find> 鍵將文字凸顯。
- 使用 FULLFORM WORD。凸顯效果會被取消，且如果有下一個字的話，游標會移到下一個單字的開頭。

如果您並未凸顯任何文字，則這項功能會將現用單字的字母轉換爲全形的字元；而如果游標介於兩個單字之間，則會影響到下一個單字。

HALFFORM WORD

格式 **HALFFORM WORD**

描述 將現用文字或凸顯文字中的所有全形字元變成半形的字元（亦即 ASCII 字母）。欲使用這個功能，則請：

- 選擇性地使用 SELECT 鍵或 FIND 鍵將文字凸顯。
- 使用 HALFFORM WORD。凸顯效果會被取消，且如果有下一個字的話，游標會移到下一個單字的開頭。

如果您並未凸顯任何文字，則這項功能會將現用單字的字母轉換爲半形的字元；而如果游標介於兩個單字之間，則會影響下一個單字。

HTPU

格式 **HTPU** *(procedure-name/statement)*

描述 執行編輯的時候所執行的 HTPU 程序或述句。這項功能與 EVE 的TPU 功能完全相同。

參數 ***procedure-name/statement***
待執行的 HTPU 程序名稱或述句。

範例

```
Command: HTPU COPY_TEXT(FAO ("!%D", 0));
```

上例執行 COPY_TEXT 內建程序來輸入現用的時間和日期。它使用 FAO 內建程序。

LEFT ADJUST

格式 **LEFT ADJUST**

描述 藉由去掉行首的空格，將那些部份或完全包含於現用選取範圍中的文字行，改變爲與現用緩衝區的左界對齊。

這個命令相當於 HEDT 的 ADJL 命令。

LEFT INDENT

格式 LEFT INDENT

描述

將現用選取範圍內之文字重新格式化，使最大數量的文字能放入從該範圍的起始欄開始，並結束於緩衝區右界上的每一行。在選取範圍中第一行以外的每一行，在使用 LEFT INDENT 之後，將會得到與該範圍起始欄相同的一個左界值。

欲使用這項功能，請：

- 使您想要 "LEFT INDENT" 的文字範圍，從您要求的那一欄開始。
- "SELECT" 文字的範圍
- 使用 LEFT INDENT 命令

有關細節，請參閱 《HEVE 使用者手冊》 第三章。

LOWERCASE WORD

格式 **LOWERCASE WORD**

描述 把現用的單字或凸顯的文字改成小寫。使用這項功能的時候，請遵循以下步驟：

- 您可以使用 <Select> 鍵或 <Find>鍵來凸顯文字。
- 用 LOWERCASE WORD 完成工作。這樣將取消那些文字的影像特質，如果其後還有單字的話，游標將移至下一個單字的開頭。

如果您沒有凸顯任何文字，這項功能會把目前的單字改爲小寫，如果游標介於兩個單字，則下一個單字會改爲小寫。

除非那些字元是中文的文數字，這個功能將不影響它們。這項功能把大寫的字元改爲小寫格式（請參閱 CHARACTER_CLASS 內建一節）。

MOVE BY WORD

格式 MOVE BY WORD

描述 依現用的方向把游標一次移動一個單字。如果方向爲往前，游標移至下一個單字的開頭；當方向爲倒轉，游標移至目前單字的開頭；如果游標已經在該位置，則它將移到前一個單字的開頭。請注意，在此將中文的「單字」界定爲：一個單字是一系列的同類字元 (請參閱 CHARACTER_CLASS 內建一節)。因此，把游標移一個單字則會把它移到另一類的一個字元上，或者如果沒有發現任何不同類字元，就移到列尾。

MOVE DOWN/UP

格式 **MOVE DOWN/UP**

描述 將游標向下或向上移動，一次一行，並使游標停留在現用字元的第一欄上。

當顯示游標設定爲 HEVE 的既定值 ON 時，這個命令將不會改變內部游標欄的位置。然而，當顯示游標設定爲 OFF 時，MOVE 命令將會改變內部游標到每個字元的第一欄，以致當連續使用 MOVE 命令時，游標會逐漸改換至左邊。有關細節，請參閱 SET [NO]DISPLAY CURSOR。

MOVE LEFT/RIGHT

格式 **MOVE LEFT/RIGHT**

描述 把游標一次移動一個字元，其方向根據命令而定。如果游標是自由的，它可以移到緩衝區中的任何位置，不管那裡有沒有字元。如果游標是受限的，在向左移動的時候，它會由一列的開頭移到前一列的結尾，在向右的時候，它會由一列的結尾移到下一列的開頭。對於中文字元中，游標總是停在該字元的第一個位元組上。

REPLACE

格式 **REPLACE** *(old-string new-string/"old-string" "new-string")*

描述 用一個文字串取代另一個文字串。HEVE 搜尋那個舊字串，如果找到了就凸顯它，並要求使用者回答：

- YES - 取代之，並搜尋下一個。
- NO - 跳過，並搜尋下一個。
- ALL - 不再詢問而取代所有的舊字串。
- LAST - 取代找到的字串同時停止。
- QUIT - 跳過找到的字串同時停止。

對 YES 和 ALL 而言，如果搜尋經過緩衝區一次以上，這個命令會問您是否要繼續進行。當這個動作結束之後，系統會在畫面底部顯示總共取代的個數。

REPLACE 命令首先會使用 FIND 來搜尋欲取代的字串。SET FIND 命令對 REPLACE 的影響，就如它對 FIND 命令的影響一樣。有關 細節，請參閱 FIND 命令。

在搜尋目標字串時，如果搜尋的字串屬於下列類別中的一種，而新字串的字母數字字元均爲小寫字體和半形，則會根據大小寫字體和字形大小來進行取代。

- 所有文數字字元均爲小寫、半形
- 所有文數字字元均爲大寫、半形
- 所有文數字字元均爲小寫、全形

- 所有文數字字元均爲大寫、半形
- 所有文數字字元均爲小寫和半形，第一個字元則可爲大寫和半形
- 所有文數字字元均爲小寫和全形，第一個字元則可爲大寫和全形

HEVE 會將取代字串放入現用緩衝區中，但保留被取代字串的大小寫和字形大小。如果不這麼做，則取代字串將會與您所提供的新字串參數有完全相同的大小寫和字形大小。請參閱 SET FIND 命令的描述。

參數

old-string
您想去除的文字。它可能包含單一位元組或多位元組字元。

new-string
您想用以取代舊字串的文字。它可能包含單一位元組或多位元組字元。

範例

```
1     Command: REPLACE Wrod Word
      Replace? Type yes, no, all, last, or quit: ALL
      Replaced 8 occurrences
```

以上的命令把所有的 "Wrod" 字串以 "Word" 字串取代。取代的數目一共是 8 個。
如果要取代片語 (幾個字)，請把字串放在引號中，如下例所示：

```
2     Command: REPLACE "TEXT-EDIT PROCESS" "TEXT PROCESSING"
```

RIGHT ADJUST

格式 **RIGHT ADJUST**

描述 藉由行首插入空格，將部份或完全位於現用選取範圍內的文字行，變成爲與右界對齊。

這個命令相當於 HEDT 的 ADJR 命令。

RIGHT INDENT

格式 RIGHT INDENT

描述 將現用選取範圍內之文字重新格式化，使最大數量的文字能放入從該範圍的起始欄開始，並結束於緩衝區右界上的每一行。甚至，藉由插入空格，RIGHT INDENT 會造成受到影響的每一行都向右對齊，但最後一行除外。在使用 RIGHT INDENT 之後，選取範圍中的每一行都會得到與該範圍起始欄相同的一個左界值，並會結束於緩衝區的右界。

這個命令相當於 HEDT 的 NDTR 命令。

SAVE EXTENDED HEVE

格式

SAVE EXTENDED HEVE *(section-filespec)*

描述

把現用的字鍵定義和其它的擴充功能存到一區段檔案中，以備日後編輯的需要。SAVE EXTENDED HTPU 是這個命令的同義字。

區段檔案是有累積性的。您可以把新增的字鍵定義和擴充功能儲存到一個舊區段檔案中。因此，建立一個區段檔案是創造您習慣的 HEVE環 境的一個辦法。雖然如此，一個區段檔案並不儲存您的版面邊界、游標設定以及標號位置。您可以使用一個初設檔來設定這些既定值。

參數

section-filespec
您要創造的區段檔案。既定檔型爲 TPU$SECTION。

範例

```
Command: SAVE EXTENDED HEVE sys$login:myeve
```

上例產生一個叫做 MYEVE.TPU$SECTION 的區段檔案。

SAVE EXTENDED HTPU

格式 **SAVE EXTENDED HTPU** *(section-filespec)*

描述 把現用的字鍵定義和其它的擴充功能存到一個區段檔案中，以備日後編輯的需要。SAVE EXTENDED HEVE 是這個命令的同義字。

區段檔案是有累積性的。您可以把新增的字鍵定義和擴充功能儲存到一個正在使用的舊區段檔案中。因此，建立一個區段檔案是創造您習慣的 HEVE 環境的方式之一。雖然如此，一個區段檔案並不儲存您的版面邊界，游標設定和標號位置。您可以使用一個初設檔案來設定這些既定值。

參數 ***section-filespec***
您要創造的區段檔案。既定的檔型爲 TPU$SECTION。

範例

```
Command: SAVE EXTENDED HTPU sys$Login:myeve
```

上例產生一個叫做 MYEVE.TPU$SECTION 的區段檔案。

SET [NO]DISPLAY CURSOR

格式 SET [NO]DISPLAY CURSOR

描述 這個命令將 HTPU 的顯示游標模態轉爲 ON 或 OFF。

當顯示游標模態爲 ON 時，如果游標目前位於多位元組字元上，則您在螢幕上所看到的游標（稱爲螢幕游標）會與儲存於 HTPU 中的 游標（稱爲內部游標）有所不同。

例如，當內部游標正位於多位元組字元的第二個位元組上時，如果顯示游標模態爲 OFF，則螢幕游標會位於字元的第二欄，而如果顯示游標模態爲 ON，則螢幕游標會位於字元的第一欄。

顯示游標模態讓螢幕游標恆位於字元的邊界上。請注意，當顯示游標模態爲 ON，而內部游標位於多位元組字元的非第一個位元組上時，則 HTPU/HEVE 插入或重打字元，將會在如內部游標位於螢幕 游標相同的位置 -- 亦即恆位於字元邊界上 -- 的情況下進行。

既定值爲 SET DISPLAY CURSOR。

SET FIND CASE [NO]EXACT

格式 **SET FIND CASE [NO]EXACT**

描述 當欲尋找或取代的字串中的所有字母均爲小寫，而不管其字形大小爲何時，這個命令會爲 FIND 和 REPLACE 設定大小寫感應。至於字形大小感應則由 SET FIND FORM [NO]EXACT 命令所決定。

SET FIND CASE EXACT 將大小寫感應轉爲 ON。
SET FIND CASE NOEXACT 將大小寫感應轉爲 OFF。

既定值爲 SET FIND CASE NOEXACT。

SET FIND FORM [NO]EXACT

格式 **SET FIND FORM [NO]EXACT**

描述 這個命令爲 FIND 和 REPLACE 命令設定字形大小感應（亦即，是否能夠區分全形與半形）。至於大小寫感應則由 SET FIND CASE [NO]EXACT 命令所決定。

SET FIND FORM EXACT 將字形大小感應轉爲 ON。
SET FIND FORM NOEXACT 會將字形大小感應轉爲 OFF。

既定值爲 SET FIND FORM NOEXACT。

SET FIND GENERAL

格式 **SET FIND GENERAL**

描述 這個命令相當於 SET FIND CASE NOEXACT 加上 SET FIND FORM NOEXACT。

請參閱 SET FIND CASE [NO]EXACT 和 SET FIND FORM [NO]EXACT。

SET FIND EXACT

格式 **SET FIND EXACT**

描述 這個命令相當於 SET FIND CASE EXACT 加上 SET FIND FORM EXACT。

請參閱 SET FIND CASE [NO]EXACT 和 SET FIND FORM [NO]EXACT。

SET TABS (MOVEMENT)

格式 **SET TABS** *(MOVEMENT)*

描述 設定 <Tab> 鍵功能，以便於按下 <Tab> 鍵時能立即移至下一個跳格處。如果下一個跳格落在多位元組字元之中，則游標會自動移到下一個字元的開頭。如此可避免游標位於多位元組字元當中，而能落在字元的開頭。

SPECIAL INSERT

格式 **GOLD/[integer]/GOLD+KP3**

描述 使用一個字元的 ASCII 值插入該字元。這項功能被修訂爲無法輸入大於 127 的值。如此就可避免將多位元組字元的一部份輸入文字中。

範例

```
Command: SET KEYPAD EDT Press GOLD key; enter 10； press
GOLD key and follow by KP3
```

上例在緩衝區中現用游標所在的位置輸入一個換列（linefeed）字元。

SYMBOL

格式 **SYMBOL** *(text-string)*

描述 這是 HEVE 才有的功能。它叫用 HSYLIB（作業庫）的 JSY$TRA_SYMBOL 函數來畫線或畫框。如果在這個命令之後沒有接著任何字串，系統會顯示所有可供選擇的符號，這些包括那些不容易使用鍵盤選擇的中文符號。每個符號都對應於某一個特殊的 ASCII 字串。當您在這個函數要求下輸入這些特殊字串，系統會在現用的編輯位置插入其相應的符號。

如果輸入不正確的字串，系統將不翻譯這個字串而原封不動地把它插入現用游標所在的位置。

參數 ***text-string***
包括某些符號所對應的特殊字串。如果其中有些字串是不正確的，系統將不翻譯該個字串而原封不動地把它與其它翻譯過的符號插入文字中。

範例

```
Symbols: 7------------------9
```

上例畫出一個四邊形框的頂部，其寬度爲 40 行寬。

UPPERCASE WORD

格式 **UPPERCASE WORD**

描述 把現用的單字或凸顯的文字區改成大寫。使用這個命令的時候，請遵循以下步驟：

- 您可以用<Select> 鍵或 <Find> 鍵來凸顯文字。
- 使用 UPPERCASE WORD 來完成工作。這樣將取消凸顯，如果其後還有單字的話，將游標移至下一個單字的開頭。

如果您沒有凸顯任何文字，這項功能會把目前的單字改爲大寫；如果游標介於兩個單字，則這項功能會將下一個單字改爲大寫。

除非那些字元是中文的文數字，這項功能將不影響它們。此功能僅將小寫的字元改爲大寫格式 (請參閱 CHARACTER_CLASS 內建一節) 。

附錄 A

HTPU/HEVE 的限制

在 ASCII 內建程序中，如果指定的整數值參數介於 161 和 255之間，在 DECTPU 內 ASCII 會傳回 "多國字元集" (MCS) 中的字元。但是，在 HTPU 中則不然。HTPU 視傳回的字元爲二位元組字元的一部份。因此，我們建議使用者不要使用如 COPY_TEXT 這樣的命令，在螢幕上顯示從 ASCII 內建程序中傳回的結果字元。

HTPU 中的 FAO 內建程序無法處理多位元組字元。如果多位元組字元被納入做爲 FAO 的字串參數，則後果將無法預期。

HTPU 中的 READ_LINE 內建程序無法適當讀取多位元組字元。在輸入多位元組字元的情形中，READ_LINE 在終止動作發生之前，會自行終止並傳回輸入的字元。至於多位元組字元則不會傳回。

HTPU 的 TRANSLATE 內建程序無法處理多位元組字元。如果多位元組字元被納入做爲 TRANSLATE 的參數，則後果將無法預期。

HEVE 的 WILDCARD FIND 不能使用 \+ 符號。這是因爲這個符號只處理介於 128 和 255 之間的 ASCII 碼（亦即單一位元組的字元）。

目前這版 HTPU/HEVE 不支援 EVE$BUILD。

附錄 B

附加的 HTPU/HEVE 訊息

此附錄列示了 HTPU/HEVE 之附加訊息。在下表中，您可找到訊息之解釋及建議之解決方法，用以挽救訊息所表示之錯誤。

至於其他的訊息，請參閱《DEC Text Processing Utility Reference Manual》。

B.1 HTPU 訊息

表 B-1 附加的 HTPU 訊息及其重要程度

縮寫	訊息	嚴重程度
RIMP_NOT_EXIST	輸入方法的 RIMP 並不存在於 DECwindows 伺服器上	INFORMATIONAL