HTPU와 HEVE 참조서

주문 번호 : BA322-90029

2005년5월

본 참조서는 한글 텍스트 처리 유틸리티(HTPU) 와 한글 확장 다능 편집기 (HEVE)에 대한 개요입니다.

개정/갱신	정보 :	본 개정 참조서는 VMS/Hangul AXP V1.5 용 HTPU와 HEVE 참조서를 대신한 것입니다.
소프트웨어	버전 :	OpenVMS/Hangul I64 Version 8.2 OpenVMS/Hangul Alpha Version 7.3−2

Hewlett−Packard Company
Palo Alto, California

Printed in Singapore

목 차

그림 목록

표 목록

서 문

소개

이 지침서는 HTPU(Hangul Text Processing Utility) 언어와 사용자 인터페이스 HEVE(Hangul Extensible Versatile Editor)를 설명하고 있습니다.

이 지침서는 HTPU 및 HEVE가 OpenVMS DECTPU 및 EVE 와 상이한 부분만을 설명하므로, 사용자는 한글 기능에 해당되지 않는 기능들에 대해서는 DEC Text Processing Utility Reference Manual 이나 Extensible Versatile Editor Reference Manual 을 참조하기 바랍니다.
이 지침서는 자세한 지침서라기 보다는 참조용입니다.

사용자

이 지침서는 HTPU의 한글 기능을 설명하는 것으로서, DECTPU의 일반적인 기능에 대한 기본 지식이 있는 사용자를 위한 것입니다.

DECTPU와 EVE의 일반적인 기능설명은 DEC Text Processing Utility Reference Manual 이나 Extensible Versatile Editor Reference Manual 을 참고 하십시오.

구성

이 지침서는 세 개의 장과 두 개의 부록으로 구성되어 있습니다. 제1장은 HTPU와 HEVE의 일반적인 설명이 포함되어 있습니다. 또한 HTPU와 HEVE의 중요한 기능에 대해서도 자세히 설명되어 있습니다. 제2장과 제3장은 신규 및 수정된 HTPU 내장 프로시저와 신규 및 수정된 HEVE 명령어에 관해 상세한 정보를 제공합니다. 부록 A에서는 HTPU와 HEVE의 제한 사항에 대하여 설명하고 있으며, 부록B에는 HTPU와 HEVE의 새로운 메시지를 열거합니다.

관련 책자

1. *HEVE* 사용자 지침서
2. *Guide to the DEC Text Processing Utility*
3. *Extensible Versatile Editor Reference Manual*
4. *DEC Text Processing Utility Reference Manual*

제 1 장

HTPU와 HEVE 의 개요

HTPU (Hangul Text Processing Utility)는 OpenVMS/Hangul 하에서 실행됩니다. HTPU는 DEC Text Processing Utility 를 기초로 하여, 원문 처리시에 ASCII 문자 뿐만 아니라 복수 바이트 문자[1]도 처리할 수 있습니다. HTPU의 한글 처리 기능은 완전한 세트의 새로운 HTPU 또는 수정된 DECTPU 내장 프로시저를 통해 수행됩니다.
HTPU에는 다음과 같은 추가 기능이 있습니다.

- 복수 바이트 문자의 인식 및 처리
- 심볼 및 선 그리기 기능
- 문자 분류
- 전자(full form) 문자 및 반자(half form) 문자 변환
- 복수 바이트 문자의 삭제 및 커서 이동
- 복수 바이트 문자를 포함한 패턴 매칭
- DECwindows Motif/Hangul 윈도우 작업 환경 지원

HEVE(Hangul Extensible Versatile Editor)는 사용자와 HTPU 간에 긴밀한 인터페이스를 제공합니다. HEVE에서는 EVE[2] 기능 이외에 한글 문자 처리를 위한 기능이 새로 추가 및 향상되었습니다.

---

[1]복수 바이트 문자란 1. 한글 문자 2. DEC KS C 5601-1987 문자 표에서 1바이트 ASCII 문자에 해당하는 2 바이트 문자 즉 전자 문자 3. 2바이트 기호 를 포함한 문자를 의미합니다.

[2]HEVE가 MCS 문자 편집을 지원하지 않음을 제외

1.1HTPU와 HEVE의 호출

1.1.1 HTPU의 새로운 DCL 명령 동작어

다음과 같은 기본 DCL 명령을 사용하면 HTPU가 HEVE(기본 편집기) 와 함께 호출됩니다.

```
$ EDIT와 HTPU
```

HTPU는 편집기 사용자 인터페이스를 찾을 때 논리명 HTPU$SECTION 을 사용하며, 이 논리명은 시스템이 HEVE의 섹션 파일 즉, HEVE$SECTION을 지정하도록 정의한 것입니다.

이 명령 입력의 번거로움을 덜기 위하여, 다음과 같이 외부 명령을 정의할 수도 있습니다.

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

이 외부 명령을 사용하면 다음과 같이 간단히 HEVE만 입력하여 HTPU를 시작할 수 있습니다.

```
$ HEVE
```

DECwindows Motif/Hangul 인터페이스 기능이 있는 HEVE를 사용하여 HTPU를 호출하려면 다음과 같은 DCL 명령을 사용하십시오.

```
$ SET DISPLAY/CREATE/NODE=<사용자 워크스테이션 노드명>
$ EDIT/HTPU/DISPLAY=MOTIF
```

또는

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

여기서, HEVE는 위에서 정의된 것입니다.

HTPU의 변수에 파일명을 포함시키면, 해당 파일이 편집시에 자동적으로 열립니다.

HTPU는 DECTPU에서 사용하는 것과 유사한 세트의 기본 명령 한정자 값을 사용합니다. 이들 새로운 명령 한정자, 수정된 명령 한정자 값, 논리명 및 파일명에 대해서는 다음 절에서 설명합니다.

주 : 사용자는 HTPU를 시작하기 전에 HANGULGEN 유틸리티를 사용하여 해당 단말기 특성 및 프로세스 환경을 설정해 주어야 합니다. 자세한 내용은 OpenVMS/Hangul 사용자 지침서 버전 6.0을 참조하십시오.

1.1.2 변경된 명령 한정자 값

/COMMAND
HTPU 에서 /COMMAND 한정자의 기본값은 HTPU$COMMAND입니다.

/DISPLAY
이 한정자의 기본값은 CHARACTER_CELL입니다. 이 값을 선택하면 이미지 HTPU$CCTSHR.EXE를 탐색합니다. 이 값으로 MOTIF를 선택하면 HTPU$MOTIFSHR.EXE가 사용됩니다.

/INITIALIZATION
HEVE 편집기의 기본 초기 설정 파일은 HEVE$INIT.EVE입니다. HEVE 초기 설정 파일의 논리명을 사용하려면, 논리명 HEVE$INT가 사용자의 초기화 파일을 지정하도록 정의하십시오.

/SECTION
HTPU에서 /SECTION 한정자의 기본값은 HTPU$SECTION입니다.

/WORK
HTPU의 기본 작업 파일명은 HTPU$WORK.TPU$WORK입니다.

1.1.3 변경된 논리명

HTPU에서의 논리명이 DECTPU에서의 논리명과 다르더라도, 그 용도는 유사합니다.

다음표는 HTPU에서의 논리명 변경 내용을 요약한 것입니다.

표 **1-1** 변경된 논리

DECTPU/EVE	**HTPU/HEVE**	설명
TPU$COMMAND	HTPU$COMMAND	명령 파일의 논리명
TPU$DEBUG	HTPU$DEBUG	디버거 파일의 논리명
TPU$DISPLAY_MANAGER	HTPU$DISPLAY_MANAGER	화면 갱신
TPU$JOURNAL	HTPU$JOURNAL	저널 파일의 목록
TPU$SECTION	HTPU$SECTION	섹션 파일의 논리명
TPU$WORK	HTPU$WORK	작업 파일의 논리명
EVE$INIT	HEVE$INIT	HEVE의 초기 설정 파일
EVE$KEYPAD	HEVE$KEYPAD	HEVE의 기본 키패드

OpenVMS/Hangul은 시스템 전반에서 HTPU$SECTION으로 정의된 논리명 HEVE$SECTION을 사용합니다.

1.1.4 변경된 기본 파일명

다음 표에 HTPU와 HEVE의 기본 파일명 중에서 변경된 것만을 요약해 놓았습니다.

표 **1-2** 기본 파일의 변경사항

DECTPU/EVE	**HTPU와 HEVE**	설명
TPU.DAT	HTPU.DAT	HTPU 기본 DECwindows Motif/Hangul 리소스 파일
TPU$COMMAND.TPU	HTPU$COMMAND.TPU	기본 명령 파일
TPU$DEBUG.TPU	HTPU$DEBUG.TPU	기본 디버그 파일
TPU$WORK.TPU$WORK	HTPU$WORK.TPU$WORK	기본 작업 파일
EVE.DAT	HEVE.DAT	HEVE 기본 DECwindows Motif/Hangul 리소스 파일
EVE$SECTION.TPU$SECTION	HEVE$SECTION.TPU$SECTION	기본 섹션 파일
EVE$INIT.EVE	HEVE$INIT.EVE	HEVE의 기본 초기 설정 파일
EVE$WIDGETS_MOTIF.UID	HEVE$WIDGETS_MOTIF.UID	HEVE의 기본 사용자 인터페이스 설명(UID) 파일

1.2 DEC KS C 5601-1987 문자세트를 위한 새 HTPU 핵심어 정의

기능 향상을 위해 다음과 같은 새 핵심어(keyword)가 HTPU에 추가되었습니다.

- ASCII_CHAR
- DISPLAY_CURSOR
- FULL_FORM
- HANGUL
- JUSTIFY_BOTH
- JUSTIFY_LEFT
- JUSTIFY_RIGHT
- NON_HANGUL
- SIZE_INVERT

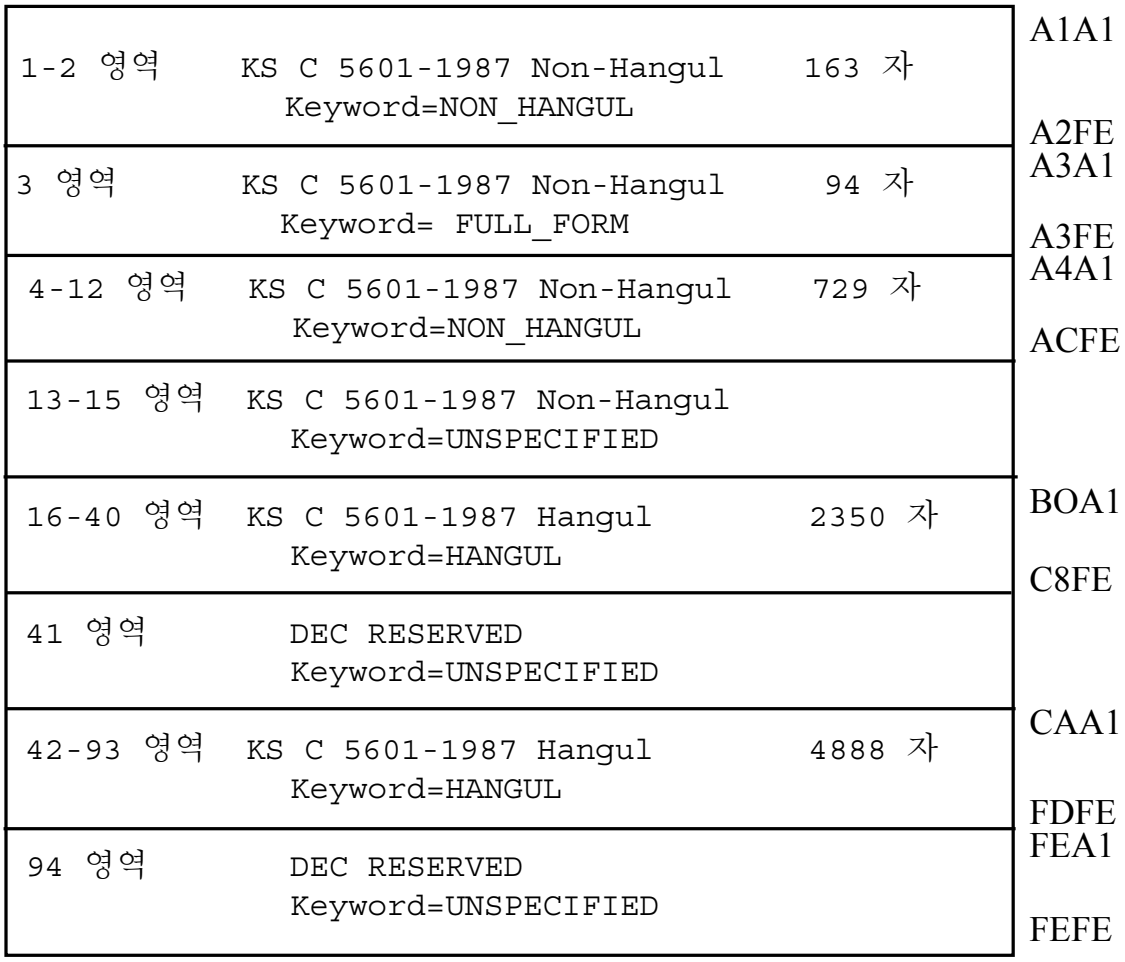
ASCII_CHAR는 ASCII 문자의 그룹을 표시합니다. 핵심어 ASCII CHAR, FULL_FORM 및 SIZE_INVERT의 사용법은 CHANGE_CASE 내장 프로시저의 설명에 자세히 나와 있습니다. 반자는 전자로, 전자는 반자로의 문자 변환에 있어서의 제한사항도 또한 CHANGE_CASE 내장 프로시저의 설명에 나와 있습니다.

JUSTIFY_LEFT, JUSTIFY_RIGHT 및 JUSTIFY_BOTH는 JUSTIFY 내장 프로시저가 정렬될 방향으로 수행되도록 지정합니다. 상세한 내용은 JUSTIFY 내장 프로시저의 설명을 참조하십시오.

DISPLAY_CURSOR는 HTPU의 화면 커서 표시 모드를 켜고/끄는 SET(DISPLAY_CURSOR) 내장 프로시저에서 사용됩니다. 상세한 내용은 SET (DISPLAY_CURSOR) 내장 프로시저를 참조하십시오.

FULL_FORM, HANGUL 및 NON_HANGUL은 DEC KS C 5601-1987 문자 세트 표에서 이들 영역을 구분하는데 사용됩니다. 다음 그림은 문자 세트의 그룹을 핵심어로 정의한 것을 보여줍니다.

그림 **1-1 DEC KS C 5601-1987** 한글 세트 표



1.3 DECwindows Motif 환경에서의 HTPU와 HEVE 및 DECTPU/EVE 간의 차이점

HTPU와 HEVE와 DECTPU/EVE는, HTPU와 HEVE가 한글 편집 기능을 지원하는 점 외에는, DECwindows Motif 환경에서 기본적으로 같습니다. 아래의 소절들에서는 이들 간의 차이점을 설명합니다.

1.3.1 한글 풀다운 메뉴 및 팝업 메뉴

HEVE는 영문과 한글 둘 다 풀다운 메뉴와 팝업 메뉴를 가지고 있습니다. 사용자는 DECwindows Motif/Hangul의 세션 관리자 (Session Manager)에서 언어를 전환하는 것과 같은 방법으로 HEVE의 메뉴 원문 언어를 전환할 수 있습니다.

사용자는 세션 관리자의 "옵션" 풀다운 메뉴중 "언어..."항목을 선택하기만 하면 됩니다. "언어 옵션" 팝업 메뉴에서 한글을 선택하면 한글 메뉴 원문으로 전환할 수 있으며, 영어를 선택하면 영어 메뉴 원문으로 전환할 수 있습니다.

1.3.2 HTPU Widget의 명칭

DECTPU는 표 1-4에서와 같이 세개의 widgets을 생성합니다.

표 1-3 DECTPU widgets

종류	명칭	설명
Tpu	tpu	어플리케이션 쉘
Main	tpu$mainwindow	주 윈도우
DECterm	DECwindows DECTPU	DECterm widget

HTPU에서는 widget 명칭이 표 1-5에서와 같이 변경되었습니다.

표 1-4 HTPU widgets

종류	명칭	설명
Htpu	htpu	어플리케이션 쉘
Main	tpu$mainwindow	주 윈도우
DECterm	DECwindows HTPU	DECterm widget

일반적으로, 이 명칭은 HTPU 또는 HEVE의 최상부 계층의 어플리케이션에는 영향을 미치지 않습니다. 그러나, 사용자의 어플리케이션이 이들 widgets의 자원을 설정하는 경우에는 widgets을 생성한 HTPU를 위한 새로운 명칭을 사용하여야 합니다.

1.3.3 CSText Widget의 지원

HTPU 내부에서 CSText widget를 생성한 후, SET(TEXT) 내장 프로시저를 사용하여 CSText의 원문 자원(text-resoruce)을 처리할 수 있습니다.

1.3.4 한글 입력 방법

HTPU가 DECwindows Motif/Hangul 하에서 기동된 경우에는, LK201

키보드에서 <COMPOSE><SPACE> 키를 동시에 누르거나 또는 LK401 키보드에서 <COMPOSE> 키를 눌러서 한글 문자를 입력할 수 있습니다. 한글 입력 모드가 시작되면, 단말기에 표시되는 데로 입력되며, 한글 입력을 시작할 때와 같은 키를 눌러서 한글 입력 모드를 종료합니다.

편집중에는 일부 요인에 의해 DECwindows Motif/Hangul의 입력 처리가 중단 될 수도 있습니다. 이 경우, 입력 처리 모드가 재개되면 HTPU는 자동으로 새로운 입력 처리 모드를 사용할 수 없습니다. 대신에, 이 경우에는 사용자가 HTPU의 입력 처리 모드를 재설정하여야 합니다.
HEVE는 "옵션" 풀다운 메뉴에 "입력 방법 재시작" 항목을 제공하고 있습니다. 입력 처리 모드가 중단되었다가 재개되면, 이 명령을 사용하여 HTPU의 입력 처리 모드를 재설정할 수 있습니다.

1.4 지원되는 단말기

일반적으로 HTPU와 HEVE는 OpenVMS/Hangul 하에서 실행되며 OpenVMS/Hangul이 지원하는 영상 단말기(video terminals)에서 한글 및 ASCII 원문의 화면 편집(screen-oriented editing)을 지원합니다.

1.5 2개국어 도움말

HTPU와 HEVE는 부시스템 레벨에서 도움말을 2개국어로 제공합니

다. HTPU와 HEVE를 호출하기 전에, HANGULGEN 유틸리티를 사용하여 도움말 언어를 한글로 설정할 수 있습니다.

1.6 2개국어 메시지

HEVE는 정보 메시지가 영어 또는 한글로 화면에 표시될 수 있게 합니다. 사용자는 HTPU와 HEVE를 호출하기 전에 HANGULGEN 유틸리티를 사용하여 원하는 언어를 설정할 수 있습니다.

제 2 장

신규 및 수정된 **HTPU** 내장 프로시저

이 장은 HTPU에서의 신규 및 수정된 내장 프로시저에 대해 설명합니다. 완전한 DECTPU 내장 프로시저의 목록은 DEC Text Processing Utility Reference Manual을 참조하십시오.

다음 표는 HTPU에 새로 추가된 신규 내장 프로시저의 목록입니다.

표 2-1 신규 HTPU 내장 프로시저

내장 프로시저
ALIGN_CURSOR
CHANGE_SIZE
CHARACTER_CLASS
COMPOSE
DEC_HANGUL
IS_CLASS
JUSTIFY
SET(ALIGNMENT_DEFAULT)
SET(DISPLAY_CURSOR)
SET(FILL_NOT_BEGIN)
SET(FILL_NOT_END)
SET(MARGIN_ALLOWANCE)
SET(UNIT_DEFAULT)

다음 표는 HTPU의 수정된 내장 프로시저 목록입니다.

표 2-2 수정된 HTPU 내장 프로시저

내장 프로시저
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

이 내장 프로시저는 커서를 문자의 첫번째 열로 이동합니다.

형식

ALIGN_CURSOR

설명

커서가 복수 바이트 문자의 두번째 열에 위치해 있을 때, ALIGN_CURSOR는 커서를 그 문자의 첫번째 열로 이동시킵니다. 커서가 현재 문자의 첫번째 바이트에 위치되면, 이 내장은 유효하지 않습니다.

CHANGE_CASE

이 내장 프로시저는 원문 내의 특정 부분에서의 모든 알파벳 문자의 문자형(case)을 변경하고, 2-바이트 전자 문자를 반자의 ASCII 문자로, 반자의 ASCII 문자를 2-바이트 전자 문자로 변경합니다.

형식

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

매개변수

buffer
문자형을 변환하고자 하는 문자가 들어있는 버퍼. 첫번째 매개변수에 버퍼를 지정하면, 세번째 매개변수에 핵심어 NOT_IN_PLACE를 사용할 수 없음에 주의하십시오.

range
문자형을 변환하고자 하는 범위. 첫번째 매개변수에 범위를 지정하면, 세번째 매개변수에 핵심어 NOT_IN_PLACE를 사용할 수 없음에 주의하십시오.

string
사용자가 변환하고자 하는 문자열을 지정하는 문자열을 나타내는 문자열 상수 또는 표현식을 나타내는 변수명입니다. 세번째 매개변수

에 IN_PLACE를 사용한 경우에는 CHANGE_CASE가, 첫번째 매개변수에서 지정한 문자열에 대해 지정된 변환을 수행합니다.

keyword1
이 핵심어는 사용자가 원하는 변환의 종류를 지정합니다.

- LOWER - 지정된 문자열 내의 알파벳 문자(전자, 반자)를 소문자로 바꿉니다.

- UPPER - 지정된 문자열 내의 알파벳 문자(전자, 반자)를 대문자로 바꿉니다.

- INVERT - 지정된 문자열 내의 알파벳 문자(전자, 반자)를 소문자는 대문자로, 대문자는 소문자로 바꿉니다.

- ASCII_CHAR - 지정된 문자열 내에서, DEC KS C 5601-1987 문자표 내 제 3영역의 2-바이트 전자 문자와 제 1영역의 '＼'와 '~'를 ASCII 문자(반자)로 바꿉니다. ASCII_CHAR 하에서는 CHANGE_CASE가 '₩'에 아무런 영향을 미치지 않음에 주의하십시오.

- FULL_FORM - 지정된 문자열 내의 ASCII 문자(반자)를 2-바이트 전자 문자로 바꿉니다. '\'와 '~'를 제외한 나머지 반자 문자들은 DEC KS C 5601-1987 문자표의 제3영역에 있는 2-바이트 전자 문자로 바꿉니다. '\'와 '~'는 DEC KS C 5601-1987 문자표의 제1영역내에 등가 2-바이트 전자 문자를 가지고 있습니다.

- SIZE_INVERT - 지정된 문자열 내의 ASCII 문자는 전자 문자로, 전자 문자는 ASCII 문자로 바꿉니다. ASCII_CHAR와 FULL_FORM 핵심어를 사용할 때 CHANGE_CASE에 적용되는 제한사항은 SIZE_INVERT를 사용할 때에도 적용됩니다.

keyword2

이 핵심어는 결과가 어디로 복귀될 것인지를 지정합니다.

- IN_PLACE - HTPU로 하여금 지시된 변환을 지정된 버퍼, 범위 또는 문자열내에서 수행하도록 지시합니다. 이것이 기본값입니다.
- NOT_IN_PLACE - HTPU로 하여금 지정된 문자열을 실제로 변환하지는 않고, 지정된 변환이 수행된 결과만 알려주도록 합니다. 첫번째 매개변수가 범위 또는 버퍼로 지정된 경우에는, NOT_IN_PLACE를 사용할 수 없습니다. NOT_IN_PLACE를 사용하기 위해서는 CHANGE_CASE에 대한 복귀값(return value)을 지정해야 합니다.

복귀값

결과에 대한 변수를 지정합니다.

- returned_buffer - 첫번째 매개변수에 버퍼를 지정한 경우, 수정된 원문이 들어갈 버퍼를 가리키는 버퍼 유형의 변수입니다. 변수 "returned_buffer"는 첫번째 매개 변수로 지정된 버퍼 변수가 가리키는 버퍼와 동일한 버퍼를 나타냅니다.
- returned_range - 첫번째 매개변수로 범위를 지정한 경우 수정된

원문이 들어갈 범위. 복귀되는 범위는 매개변수로 지정된 범위와 동일한 원문을 가지지만, 이들 범위는 별개의 범위입니다. 이후, 이들 범위 중 하나에 대하여 변경 작업을 하거나 삭제하여도 다른 범위에는 영향을 미치지 않습니다.

- return_string - 첫번째 매개변수에 문자열을 지정한 경우, 수정된 원문이 들어갈 문자열. IN_PLACE를 지정하더라도, CHANGE_CASE는 문자열을 복귀할 수 있습니다.

설명

CHANGE_CASE는 복귀값이 지정되어 있으면 결과를 복귀합니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_ARGMISMATCH	ERROR	CHANGE_CASE에 대한 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_INVPARAM	ERROR	CHANGE_CASE에 대한 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_BADKEY	WARNING	CHANGE_CASE에 틀린 핵심어를 지

		정하였습니다.
TPU$_NOTMODIF-IABLE	WARNING	수정 불가형 버퍼내에 있는 원문의 문자형은 바꿀 수 없습니다.
TPU$_CONTROLC	ERROR	CHANGE_CASE의 실행 중에 CTRL/C를 눌렀습니다.

CHANGE_SIZE

이 내장 프로시저는 지정된 단위의 원문에 있는 모든 알파벳 문자의 크기를 변경(2-바이트 전자 문자는 반자의 ASCII 문자로, 반자의 ASCII 문자는 2-바이트 전자 문자로) 합니다.

형식

[return_value:=]CHANGE_SIZE ({ buffer / range / string } , Keyword1[, Keyword2])

매개변수

buffer
문자크기를 변환하고자 하는 문자가 들어있는 버퍼, 첫번째 매개변수에 버퍼를 지정하면, 세번째 매개변수에 핵심어 NOT_IN_PLACE를 사용할 수 없음에 주의하십시오.

range
문자크기를 변환하고자 하는 문자가 들어있는 범위. 첫번째 매개변수에 범위를 지정하면, 세번째 매개변수에 핵심어 NOT_IN_PLACE를 사용할 수 없음에 주의하십시오.

string
문자열 상수를 표현하는 변수명 또는 문자열을 표현하는 표현식으로

서, 사용자가 변환하고자 하는 문자열을 지정합니다. 세번째 매개변수에 IN_PLACE를 지정하면, CHANGE_SIZE가 첫번째 매개변수에 지정한 문자열에 대하여 지정된 변환을 수행합니다.

keyword1
이 핵심어는 사용자가 원하는 변환의 종류를 지정합니다.

- ASCII_CHAR - 지정된 문자열 내에서, DEC KS C 5601-1987 문자표 내 제 3영역의 2-바이트 전자 문자와 제1영역의 '\'와 '~'를 ASCII 문자(반자)로 바꿉니다. ASCII_CHAR 하에서는 CHANGE_SIZE가 '₩'에 아무런 영향을 미치지 않습니다.

- FULL_FORM - 지정된 문자열 내의 ASCII 문자(반자)를 2-바이트 전자 문자로 바꿉니다. '\'와 '~'를 제외한 나머지 모든 반자 문자는 DEC KS C 5601-1987 문자표의 제3영역의 2-바이트 전자 문자로 바뀝니다. '\'와 '~'는 DEC KS C 5601-1987 문자표의 제1영역내에 등가 2-바이트 전자 문자를 가지고 있습니다.

- SIZE_INVERT - 지정된 문자열내의 ASCII 문자는 전자 문자로, 전자 문자는 ASCII 문자로 바꿉니다. ASCII_CHAR와 적용되는 제한사항은 SIZE_INVERT를 사용할 때에도 적용됩니다.

keyword2
이 핵심어는 결과가 어디로 복귀될 것인지를 지정합니다.

- IN_PLACE - 지시된 변환을 지정된 버퍼, 범위 또는 문자열 내에서 수행하도록 지시합니다. 이것이 기본값입니다.

- NOT_IN_PLACE - 지정된 문자열을 실제로 변환하지는 않고, 지정된 변환이 수행된 결과만 알려지도록 합니다. 첫번째 매개변수가 범위 또는 버퍼로 지정된 경우에는, NOT_IN_PLACE를 사용할 수 없습니다. NOT_IN_PLACE를 사용하기 위해서는 CHANGE_SIZE에 대한 복귀값(return value)을 지정해야 합니다.

복귀값

결과에 대한 변수를 지정합니다.

- returned_buffer - 첫번째 매개변수에 버퍼를 지정한 경우, 수정된 원문이 들어갈 버퍼를 가리키는 버퍼 유형의 변수입니다. 변수 'returned_buffer'는 첫번째 매개 변수로 지정된 버퍼 변수가 가리키는 버퍼와 동일한 버퍼를 나타냅니다.

- returned_range - 첫번째 매개변수로 범위를 지정한 경우 수정된 원문이 들어갈 범위를 나타냅니다. 복귀되는 범위는 매개변수로 지정된 범위와 동일한 원문을 가지지만, 이들 범위는 별개의 범위입니다. 이후, 이들 범위 중 하나에 대하여 변경 작업을 하거나 삭제하여도 다른 범위에는 영향을 미치지 않습니다.

- return_string - 첫번째 매개변수에 문자열을 지정한 경우, 수정된 원문을 포함하고 있는 문자열을 복귀합니다. 만일 IN_PLACE를 지정한다 하여도, CHANGE_CASE가 문자열을 복귀합니다.

설명

CHANGE_SIZE는 복귀값이 지정되어 있으면 결과를 복귀합니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_ARGMISMATCH	ERROR	CHANGE_SIZE에 대한 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_INVPARAM	ERROR	CHANGE_SIZE에 대한 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_BADKEY	WARNING	CHANGE_SIZE에 대하여 틀린 핵심어를 사용하였습니다.
TPU$_NOTMODIF-IABLE	WARNING	수정 불가형 버퍼 내에 있는 원문의 문자형은 바꿀 수 없습니다.
TPU$_CONTROLC	ERROR	CHANGE_SIZE의 실행 중에 CTRL/C를 눌렀습니다.

CHARACTER_CLASS

이 내장 프로시저는 입력 문자열 내의 첫번째 문자의 종류(예: Hangul, Non-Hangul)를 식별합니다. HTPU는 복수 바이트 문자를 한 문자로 취급함에 주의하십시오.

형식

keyword:=CHARACTER_CLASS (string)

매개변수

string
인용된 문자열, 문자열 상수를 표현하는 변수명 또는 문자열을 가리키는 표현식입니다. 그러나 문자열의 길이가 한 자를 초과하면 첫번째 문자의 종류가 복귀됩니다.

설명

복귀되는 문자열에서 첫번째의 문자의 종류를 나타내는 핵심어들은 다음과 같습니다.

- ASCII_CHAR - ASCII문자(반자문자)
- FULL_FORM - DEC KS C 5601-1987 문자표의 제3영역의 2-바이

트 전자 문자

- NON_HANGUL - 이 핵심어에 대한 설명은 이 지침서의 그림 1-1를 참조하십시오.
- HANGUL - 이 핵심어에 대한 설명은 이 지침서의 그림 1-1를 참조하십시오.
- UNSPECIFIED - 이 핵심어에 대한 설명은 이 지침서의 그림 1-1를 참조하십시오.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 틀립니다.
TPU$_NEEDTOA-SSIGN	ERROR	CHARACTER_CLASS가 할당문의 오른쪽에 있어야 합니다.

COMPOSE

이 내장 프로시저는 ASCII 문자의 문자열을 특별한 2-바이트 기호(2-byte symbols)로 바꿉니다.

형식

string2:=COMPOSE (string 1)

매개변수

string1
인용된 문자열, 문자열 상수를 표현하는 변수명 또는 문자열을 가리키는 표현식입니다. ASCII 문자의 문자열은 다음 표에 기술한 대로, 대응되는 2-바이트 기호 및 2-바이트 문자로 변환됩니다.

표 2-3 ASCII 순서 및 대응되는 2-바이트 기호

ASCII 시퀀스	대응 2 바이트 기호
-	ㅡ
0	ㅣ
1	ㄴ
2	ㅗ
3	ㅜ

표2-3 (계속) ASCII 순서 및 대응되는 2-바이트 기호

ASCII 시퀀스	대응 2 바이트 기호
4	├
5	┼
6	┤
7	┌
8	┬
9	┐
f-	─
f0	│
f1	└
f2	┴
f3	┘
f4	├
f5	┼
f6	┤
f7	┌
f8	┬
f9	┐
t1	└
t2	┴
t3	┘
t4	├
t5	┼
t6	┤

표2-3 (계속) ASCII 순서 및 대응되는 2-바이트 기호

ASCII 시퀀스	대응 2 바이트 기호
t7	┌
t8	┬
t9	┐
y1	└
y2	┴
y3	┘
y4	├
y5	┼
y6	┤
y7	┌
y8	┬
y9	┐
| |	‖
| ^	↑
| v	↓
([	〔
( )	○
)]	】
[[	「
[(	【
[ ]	□
])	〕
]]	」
< <	«

표2-3 (계속) ASCII 순서 및 대응되는 2-바이트 기호

ASCII 시퀀스	대응 2 바이트 기호
< =	≤
< >	◇
< \	△
< /	▽
< -	←
> >	»
> =	≥
> <	☆
> -	→
: -	÷
+ -	±
x x	×
= /	≠
= |	∵
, .	…
. .	‥
. ;	∴
. c	℃
` `	“
' '	″
c /	¢
l -	£
s s	§
o o	∞

표2-3 (계속) ASCII 순서 및 대응되는 2-바이트 심볼

ASCII 시퀀스	대응 2 바이트 기호
o >	♂
o +	♀
K > <	★
K ( )	●
K < >	◆
K [ ]	■
K < \	▲

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 틀립니다.
TPU$_NEEDTO-ASSIGN	ERROR	COMPOSE가 할당문의 오른쪽에 있어야 합니다.

예

```
1  PROCEDURE user_compose
   COPY_TEXT(COMPOSE('K><ss'))
   ENDPROCEDURE
```

위의 프로시저는 '★§'를 현재의 편집 위치에 삽입하는 것입니다.

```
2  PROCEDURE user_part_compose
   COPY_TEXT(COMPOSE('=/aK><'))
   ENDPROCEDURE
```

위 프로시저는 '≠a★'를 현재의 편집 위치에 삽입하는 것입니다.

CURRENT_OFFSET

이 내장 프로시저는 현재 행 내의 현재 문자 위치의 바이트 또는 문자 오프셋에 대한 정수값을 알려줍니다.

형식

integer:=CURRENT_OFFSET [(keyword)]

매개변수

keyword
지정될 수 있는 핵심어는 다음과 같습니다.

- CHARACTERS - 문자 오프셋. 오프셋의 값이 문자 단위로 계산됩니다.

- BYTES - 바이트 오프셋. 오프셋의 값이 바이트 단위로 계산됩니다.

어떤 핵심어도 지정하지 않으면 기본값 CHARACTERS가 사용됩니다. 또한 사용자는 SET(UNIT_DEFAULT) 내장 프로시저를 사용하여 기본값을 변경할 수도 있습니다.
자세한 설명은 SET(UNIT_DEFAULT)을 참조하십시오.

설명

CURRENT_OFFSET 내장 프로시저의 기본적인 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오. HTPU에 핵심어 매개변수가 추가되면, 문자 오프셋과 바이트 오프셋, 둘 다 사용할 수 있습니다. 복귀된 오프셋은 항상 문자 범위 내에 있습니다.

오류신호

TPU$_NEEDTOASSIGN	ERROR	COMPOSE_OFFSET은 할당문의 오른쪽에 있어야 합니다.
TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_NOCURRENTBUF	ERROR	버퍼에 위치하고 있지 않습니다.

예

```
1  my_char_off:=CURRNNT_OFFSET(CHARACRTERS)
```

이 할당문은 현재 문자의 문자 오프셋을 나타내는 정수값을 변수 my_char_off에 할당합니다.

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

이 할당문은 현재 문자의 바이트 오프셋을 나타내는 정수값을 변수 my_byte_off에 할당합니다.

CURSOR_HORIZONTAL

이 내장 프로시저는 화면상의 커서를 지정된 열 수 만큼 수평으로 이동시킵니다.

형식

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

매개변수

integer1
커서를 이동시키고자 하는 화면상의 열 수를 지정하는 부호있는 (+/-) 정수값입니다.

keyword
지정할 수 있는 핵심어는 다음과 같습니다.

- CHARACTERS - 커서 이동 후에 커서 위치가 복수 바이트 문자의 첫번째 바이트에 위치해 있지 않으면, CURSOR_HORIZONTAL 명령이 커서를 첫번째 열로 이동시킵니다.

- BYTES - 커서를 버퍼의 내용에 관계없이 이동시킵니다. (내부 커서 위치가 첫번째 열에 위치해 있지 않더라도, 화면 커서가 문자의 첫번째 열에 표시되도록 DISPLAY_CURSOR 모드를 설정

해 놓은 경우 제외).
보다 상세한 내용은 SET(DISPLAY_CURSOR)를 참조하십시오.

어떤 핵심어도 지정하지 않으면 기본값은 BYTES입니다. 또한 SET(ALIGNMENT_DEFAULT) 내장 프로시저를 사용하여 기본값을 변경할 수도 있습니다.
자세한 설명은 SET(ALIGNMENT_DEFAULT)를 참조하십시오.

복귀값

복귀값(return value)은 커서가 이동한 열의 갯수입니다. HTPU가 커서를 integer1으로 지정된 열 수 만큼 이동시킬 수 없는 경우에는, 가능한 열 수 만큼만 이동시킵니다. HTPU는 음수의 복귀값을 허용하고 있지만, 이 표기는 추후 버전의 HTPU를 위하여 예비되어 있습니다. 복귀값은 커서가 오른쪽이나 왼쪽으로 이동한 정확한 수의 열을 보여줍니다. 커서가 이동하지 않은 경우에는 복귀값이 0입니다. CURSOR_HORIZONTAL 에서 오류가 발생되면, 복귀값이 부정확해집니다.

설명

CURSOR_HORIZONTAL 내장 프로시저의 기본적인 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

HTPU에서는, 핵심어 매개변수를 추가하여 내장 프로시저 동작 후

에 복수 바이트 문자 위의 커서 위치를 제어할 수 있습니다.

한글 문자는 2개 이상의 열 위치를 차지할 수 있으므로, CURSOR_HORIZONTAL 명령을 사용하여 두번째 매개변수에 핵심어 BYTES를 지정하여 주므로써 커서를 복수 바이트 문자의 첫번째 이외의 열로 이동시킬 수 있습니다.

DISPLAY_CURSOR 모드가 기동중인 경우에는, 문자가 1-바이트 문자인지 또는 복수 바이트 문자인지에 관계없이, 화면상의 커서가 항상 문자의 첫번째 바이트에 표시됩니다. 그러나, 내부적으로 저장되는 커서 위치는 문자의 첫번째 이외의 바이트 위치일 수 있습니다. 보다 상세한 내용은 SET (DISPLAY_CURSOR)를 참조하십시오.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 틀립니다.

CURSOR_VERTICAL

이 내장 프로시저는 화면상의 커서를 지정된 행 수 만큼 상하로 이동시킵니다.

형식

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

매개변수

integer1
커서를 이동시키고자 하는 화면상의 행 수를 지정하는 부호있는 (+/-) 정수값입니다.

keyword
지정할 수 있는 핵심어는 다음과 같습니다.

- CHARACTERS - 커서 이동 후에, 복수 바이트 문자의 첫번째 바이트에 커서가 위치해 있지 않으면, CURSOR_VERTICAL 명령이 첫번째 열로 커서를 이동시킵니다.

- BYTES - 커서를 버퍼의 내용에 관계없이 이동시킵니다. (내부 커서 위치가 첫번째 열에 위치에 있지 않더라도, 화면 커서가 문자의 첫번째 열에 표시되도록 DISPLAY_CURSOR 모드를 설정

해 놓은 경우 제외).
보다 상세한 내용은 SET(DISPLAY_CURSOR)를 참조하십시오.

만일 아무런 핵심어도 지정하지 않으면 기본값은 BYTE입니다. 또한 SET(ALIGNMENT_DEFAULT) 내장 프로시저를 사용하여 기본값을 변경할 수도 있습니다. 이에 대한 자세한 설명은 SET(ALIGNMENT_DEFAULT)를 참조하십시오.

복귀값

복귀값은 커서가 상하로 이동한 행 수입니다. HTPU가 커서를 integer1으로 지정된 행 수 만큼 이동시킬 수 없는 경우에는, 가능한 행 수 만큼만 이동시킵니다.
이 때, CROSS_WINDOW_BOUNDS가 ON으로 설정되어 있으면 CURSOR_VERTICAL은 커서를 다른 윈도우에 위치시킵니다. 이 경우에 복귀값은 음수입니다. 복귀값이 음수값이라 하더라도, 커서가 위로 이동되지는 않습니다. 커서가 이동되지 않았으면, 복귀값은 0입니다. CURSOR_VERTICAL에서 오류가 발생되면, 복귀값이 부정확해 집니다.

설명

CRUSOR_VERITICAL 내장 프로시저에 대한 기본적인 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

HTPU에서 핵심어 매개변수를 추가하면, 내장 프로시저 수행 후에 복수 바이트 문자상의 커서 위치를 제어할 수 있습니다.

이 프로시저는 기본적으로 커서의 열 위치를 변경시키지 않습니다. 그러나, 핵심어 필드에 CHARACTERS가 지정되어 있으면, 복수 바이트 문자의 첫번째 이외의 열로 이동된 커서가 첫번째 열로 이동됩니다. 커서의 열 위치를 이동시키지 않으려면 핵심어 필드에 BYTES를 지정해 주십시오.

화면 커서 모드를 ON으로 설정한 상태에서 BYTES 핵심어를 사용하면, 내부적으로 저장된 커서의 열 위치가 변경되지 않아도 항상 복수 바이트 문자의 문자 경계에 화면 커서가 표시됩니다. 보다 상세한 내용은 SET(DISPLAY_CURSOR)를 참조하십시오.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 틀립니다.

DEC_HANGUL

이 프로시저는 문자를 문자코드(ASCII 코드 또는 DEC KS C 5601-1987 문자코드)로, 문자코드를 문자로 변환시킵니다.

형식

integer:=DEC_HANGUL (string1)

형식

string:=DEC_HANGUL (integer1)

매개변수

integer1
변환시키고자 하는 ASCII 문자 또는 복수 바이트 문자를 나타내는 정수(십진수)로서, ASCII 또는 DEC KS C 5601-1987 문자 세트를 결정하는 변수입니다.

string1
얻고자 하는 ASCII 코드 또는 DEC KS C 5601-1987 문자 코드의 문자입니다.

설명

정수 매개변수가 지정되면, DEC_HANGUL 내장 프로시저는 지정한 숫자에 따라서 1-바이트 또는 복수 바이트 문자를 복귀합니다. 문자에 대응되는 숫자는 ASCII 또는 DEC KS C 5601-1987 문자 세트의 규칙에 따라 정해지는 것으로C, 그 숫자의 하위 2-바이트만 사용됩니다. 그러나, DEC_HANGUL은 지정된 정수가 DEC KS C 5601-1987 표에 있는 유효한 문자인지에 대한 정보는 제공하지 않습니다.

문자열 매개변수가 지정되면, DEC_HANGUL 내장 프로시저는 ASCII 또는 DEC KS C 5601-1987 문자표에 따라 문자열의 첫번째 문자와 등가인 정수를 복귀합니다.

오류신호

TPU$_NEEDTO-ASSIGN	ERROR	DEC_HANGUL이 할당문의 오른쪽에 있어야 합니다.
TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_ARGMISMATCH	ERROR	매개변수에 대한 데이터 유형이 틀립니다.

TPU$_NULLSTRING	ERROR	DEC_HANGUL에 길이가 0인 문자열을 전달하였습니다.

예

다음 프로시저는 DEC KS C 5601-1987의 임의의 한 영역에 있는 표를 현재 버퍼에 삽입하는 것입니다. 예를 들어, user_hangul_list(20)를 실행하면, 20번째 영역의 표가 현재 편집 위치에 삽입됩니다.

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

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

COPY_TEXT ("        0 1 2 3 4 5 6 7 8 9 A B C D E F");
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_HANGUL (cnt));
```

```
          ELSE
               COPY_TEXT (’  ’);
          ENDIF;
        cnt := cnt + 1;
        col := 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;
        ELSE
              res := ’0’ + res;
        ENDIF;
ENDLOOP;
user_hex := res;
ENDPROCEDURE
```

다음의 행들이 사용자가 user_hangul_list(3) 명령을 발신했을 때 현재 편집 위치 다음에 표시됩니다.

```
        0 1 2 3 4 5 6 7 8 9 A B C D E F

A3A0      ! " # $ % & ' () * +, −. /
A3B0    0 1 2 3 4 5 6 7 8 9 : ; < = > ?
A3C0    @ A B C D E F G H I J K L M N O
A3D0    P Q R S T U V W X Y Z [₩] ^ _
A3E0    ` a b c d e f g h i j k l m n o
A3F0    p q r s t u v w x y z { | } ¯
```

EDIT

이 내장 프로시저는 지정된 핵심어에 따라 문자열을 수정합니다. EDIT는 DCL 렉시칼 함수 F$EDIT와 동일하지는 않으나 유사합니다 내장 프로시저와 렉시칼 함수(lexical function)의 차이점은 아래의 설명 부분에 기술되어 있습니다.

형식

[return_value:=]EDIT (buffer | range | string ,keyword1[,...][,keyword2][, keyword3])

매개변수

buffer
편집하고자 하는 원문이 들어 있는 범위. 첫번째 매개변수에 버퍼를 지정하면, 세번째 매개변수에 NOT_IN_PLACE를 사용할 수 없음에 주의하십시오.

range
편집하고자 하는 원문이 들어 있는 버퍼, 첫번째 매개변수에 버퍼를 지정하면, 세번째 매개변수에 NOT_IN_PLACE를 사용할 수 없음에 주의하십시오.

string
수정하고자 하는 문자열, 복귀값을 지정한다면, 복귀되는 문자열은 첫번째 매개변수에 지정한 문자열로 구성되며, 이때 이 문자열은 두번째 매개변수와 그 후속 매개변수에서 지정한 방식으로 수정됩니다. 세번째 매개변수에 IN_PLACE를 지정하면, EDIT가 첫 번째 매개변수에 지정한 문자열에 지정된 변경사항을 수행합니다. EDIT는 문자열 상수에는 영향을 미치지 않습니다.

keyword1
다음에 열거한 핵심어를 원하는 수만큼 선택할 수 있습니다.

- COLLAPSE - (전자 공백을 포함한) 모든 공백과 탭을 제거합니다.

- COMPRESS - (전자 공백을 포함한) 다중 공백을 단일 반자 공백으로 대체시킵니다.

- TRIM - (전자 공백을 포함한) 선행 공백과 (전자 공백을 포함한) 어미 공백, 그리고 탭을 제거합니다.

- TRIM_LEADING - (전자 공백을 포함한) 선행 공백(leading spaces)과 탭(Tab)을 제거합니다.

- TRIM_TRAILING - (전자 공백을 포함한) 어미 공백(trailing spaces)과 탭(Tab)을 제거합니다.

- LOWER - (전자 문자를 포함하여) 모든 대문자를 소문자로 바꿉니다.

- UPPER - (전자 문자를 포함한) 모든 소문자를 대문자로 바꿉니다.

- INVERT - (전자 문자를 포함한) 지정된 문자의 현재 문자형을 바꿉니다. 소문자는 대문자로, 대문자는 소문자로 됩니다.

- ASCII_CHAR - 지정된 문자열내에서, DEC KS C 5601-1987 문자표의 제 3영역의 2-바이트 전자 문자와 제 1영역의 '\'와 '~'를 ASCII 문자로 바꿉니다. ASCII_CHAR 하에서는 EDIT가 '₩'에 아무런 영향을 미치지 않습니다.

- FULL_FORM - 지정된 문자열에서 반자의 ASCII 문자를 2-바이트 전자 문자로 바꿉니다. '\'와 '~'를 제외한 나머지 반자 문자가 DEC KS C 5601-1987 문자표의 제3영역의 2-바이트 전자 문자로 바뀝니다. '\'와 '~'는 DEC KS C 5601-1987 문자표의 제 1영역내에 등가 2-바이트 전자 문자를 가지고 있습니다.

- SIZE_INVERT - 지정된 문자열 내의 반자 문자를 전자로 바꾸거나, 지정된 문자열내의 전자 문자를 반자로 바꿉니다. ASCII_CHAR와 FULL_FORM을 사용할 때에 EDIT에 적용되는 제한사항이 SIZE_INVERT에도 적용됩니다.

TRIM 핵심어(TRIM, TRIM_LEADING, TRIM_TRAILING)중 둘 이상 지정했을 때는, 지정한 모든 TRIM 동작이 다 수행됩니다.

문자변환 핵심어(UPPER, LOWER, INVERT, ASCII_CHAR, SIZE_INVERT)를 둘 이상 지정했을 때는, 마지막에 지정된 핵심어

에 따라 문자열 내의 문자가 수정됩니다.

keyword2
마지막 매개변수로 ON 또는 OFF를 지정할 수 있습니다. (생략가능함)

- ON - HTPU로 하여금, HTPU 인용부호로 단일 또는 이중 인용부호를 인식하는 기능을 가동하게 합니다. 이 경우 인용된 부문자열은 변경되지 않습니다.

- OFF - HTPU로 하여금, HTPU 인용부호로 단일 또는 이중 인용부호를 인식하는 기능을 정지하게 합니다. 인용된 부문자열은 지정된 대로 변경됩니다.

keyword3
HTPU가 표시된 변경을 수행하는 위치를 나타내는 핵심어입니다. 이의 유효한 핵심어들과 그 의미는 다음과 같습니다.

- IN_PLACE - 그 위치에서 표시된 변환을 수행합니다. 이것이 기본값입니다.

- NOT_IN_PLACE - 지정된 문자열에 변경을 수행하지 않고 그대로 둔 채, 지정된 편집 동작의 결과 문자열을 복귀합니다. 첫 번째 매개변수에 범위 또는 버퍼를 지정한 경우에는, NOT_IN_PLACE를 사용할 수 없습니다. NOT_IN_PLACE를 사용하려면, EDIT에 대한 복귀값(return value)을 지정해야 합니다.

복귀값

결과에 대한 변수를 지정합니다.

- returned_buffer - 첫번째 매개변수에 버퍼를 지정한 경우에, 수정된 원문이 들어갈 버퍼를 가리키는 버퍼의 유형 변수입니다. 변수 "returned_buffer"는 첫번째 매개 변수에서 지정된 버퍼 변수가 가리키는 것과 동일한 버퍼를 나타냅니다.

- returned_range - 첫번째 매개변수에서 범위를 지정한 경우에, 수정된 원문에 들어갈 범위를 나타냅니다. 이때 복귀되는 범위는 매개변수에 지정된 원문 범위만큼입니다. 그러나, 이들 범위는 별개의 범위입니다. 그 후, 이 범위 중의 하나를 계속하여 변경하거나 또는 삭제하여도, 다른 범위에는 영향이 없습니다.

- return_string - 첫번째 매개변수에 문자열을 지정한 경우, 수정된 원문이 들어갈 문자열입니다. 만일 IN_PLACE를 지정했다 하더라도, EDIT는 문자열을 취합니다.

설명

HTPU는 첫번째 매개변수에서 지정한 버퍼, 범위 또는 문자열을 수정합니다.

만일 keyword2를 지정하지 않았다면, 문자열 내에서 인용된 원문을 수정하지 않습니다. 예를 들면 아래의 명령어,

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

는 WELL을 소문자로 변경하지 않습니다. 따라서 이 문자열은 다음과 같이 됩니다.

```
he sang "WELL"
```

지정한 문자열에 여는 인용부호는 있으나 닫는 인용부호가 없으면, TPU$_MISSINGQUOTE 의 상태가 복귀됩니다. 이 경우에, 닫혀지지 않은 열린 인용부호부터 문자열의 끝까지의 원문이 인용된 문자열 부분으로 간주되어 수정되지 않게 됩니다.

EDIT 내장 프로시저가 HTPU 인용부호로서 단일 또는 이중 인용부호를 인식하는 기능을 정지시키기 위하여 마지막 매개변수에 핵심어 OFF를 사용할 수 있습니다. 따라서 핵심어 OFF를 지정하면 인용부호가 보통 원문과 같이 취급됩니다. EDIT는 DCL 렉시칼 함수 F$EDIT와 유사합니다. 그러나, 다음과 같은 차이점이 있음에 유의하여야 합니다.

- EDIT는 해당 위치의 문자열을 수정하지만 F$EDIT는 결과를 취합니다.

- EDIT는 매개변수로서 핵심어를 취하는 반면, F$EDIT는 편집 명령을 문자열로 지정할 것을 요구합니다. F$EDIT는 한글 문자에 대해서는 동작하지 않습니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 틀립니다.
TPU$_ARGMISM-ATCH	ERROR	EDIT의 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_MISSING QUOTE	ERROR	문자열에 닫는 인용부호가 빠졌습니다.
TPU$_BADKEY	WARNING	틀린 핵심어가 사용되었습니다.

예

다음 문은 문자열 "ＰＲＯＤＵＣＴ ＮＡＭＥ"을 ASCII 문자로 변환하고, 편집된 문자열을 메시지 윈도우에 표시하는 것입니다.

```
pn:='ＰＲＯＤＵＣＴ ＮＡＭＥ';

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

FILL

이 내장 프로시저는 지정된 버퍼나 범위 내의 원문을 재형식화(reformat)하여 원문 행들의 길이를 거의 같은 길이로 만듭니다.

형식

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

매개변수

buffer
재형식화하고자 하는 원문의 버퍼.

range
재형식화하고자 하는 원문의 범위.

string
버퍼의 원문에 대한 단어 분리자(word separator)로 사용하고자 하는 반자 ASCII 문자의 목록. ASCII 공백 문자는 항상 단어 분리자가 되며, 이 문자열에는 복수 바이트 문자를 포함할 수 없습니다.

integer1
왼쪽 가장자리의 값을 나타냅니다. 왼쪽 가장자리 값은 1보다는 크

고 오른쪽 가장자리 값보다는 작아야 합니다. 이 값이 왼쪽 가장자리의 기본값입니다.

integer2
오른쪽 가장자리의 값을 나타냅니다. 오른쪽 가장자리 값은 왼쪽 가장자리 값보다 커야 하지만, 버퍼의 최대 레코드 크기를 초과할 수 없습니다.

integer3
첫번째 행의 들여쓰기(indent) 값을 나타냅니다. 이 값은 첫번째 행의 왼쪽 가장자리를 수정합니다. 그러나, 왼쪽 가장자리에 이 값을 더한 결과가 1보다는 크고, 오른쪽 가장 자리보다는 작아야 합니다. 이 값은 0이 기본값입니다.

설명

FILL 내장 프로시저의 기본적인 설명은 DEC Processing Utility Reference Manual을 참조하십시오.

HTPU에서는, FILL 내장 프로시저가 복수 바이트 문자열을 처리할 수 있도록 기능이 향상되었습니다. 이 FILL 프로시저는, 두번째 문자 위치가 오른쪽 가장자리를 넘어가는 경우에 복수 바이트 문자를 두개의 인접된 문자로 분리시킵니다.

향상된 FILL 내장 프로시저는 예외 문자 (exceptional characters)도 처리할 수 있습니다. 사용자는 SET(FILL_NOT_BEGIN)과

SET(FILL_NOT_END) 내장 프로시저를 사용하여 예외 문자를 지정할 수 있습니다. 지정된 예외 문자들은 FILL이 실행될 때 고려됩니다.

SET(MARGIN_ALLOWANCE)에 MARGIN_ALLOWANCE를 설정하므로써, 오른쪽 가장자리를 넘을수 있는 예외 문자의 수를 지정할 수 있습니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_ARGMISMATCH	ERROR	매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_BADMARGIN	WARNING	필(fill) 가장자리 중 하나를 부정확하게 지정하였습니다.
TPU$_INVPARAM	ERROR	FILL에 대한 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_NOTMODI-FIABLE	WARNING	수정 불가형 버퍼내 원문은 변경할 수 없습니다.
TPU$_CONTROLC	ERROR	FILL 실행 중에 CTRL/C를 눌렀습니다.

TPU$_NOCACHE	ERROR	새로운 캐쉬를 할당할 만한 충분한 메모리가 없습니다.

GET_INFO

이 내장 프로시저는 사용자가 편집 중인 문맥의 현재 상태에 대한 정보를 제공합니다. 표2-4와 표2-5는 GET_INFO가 알려주는 정보 및 관련된 HTPU의 향상된 기능 목록입니다. 원래의 GET_INFO의 기능에 대해서는 DEC Text Processing Utility Reference Manual을 참조하십시오.

형식

```
return_value:=GET_INFO    (parameter1, parameter2)
```

형식

```
return_value:=GET_INFO    (parameter1, parameter2,
                           parameter3)
```

매개변수

parameter1

이것은 HTPU 데이터 유형 또는 핵심어입니다. GET_INFO는 parameter1으로 지정되는 항목에 대한 정보를 제공합니다. 표2-4는 parameter1으로 지정될 수 있는 데이터 유형 및 관련된 HTPU의 향

상된 기능들을 보여줍니다. 표2-5는 parameter1으로 지정될 수 있는 핵심어 및 관련된 HTPU의 향상된 기능들을 보여줍니다.

parameter2
인용된 문자열, 변수명 또는 표2-4나 표2-5에 있는 문자열 상수 중의 하나를 표현하는 표현식입니다. parameter2로 사용되는 문자열은 parameter1에 의해서 지정되는 항목에 대하여 요구된 정보의 종류를 표시합니다. 문자열은 대문자나 소문자로 입력할 수 있습니다.

parameter3
인용된 문자열이나 변수명입니다.

이 매개변수에 대한 자세한 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

설명

다음 페이지에 있는 표는 HTPU의 새로운 기능을 요약한 것입니다.

표 2-4 GET_INFO에서 parameter1으로 사용되는 변수

parameter1	parameter2	복귀값	복귀값에 대한 설명
버퍼 변수	"byte-offset"	정수	현재 문자 위치와 행의 시작 위치와의 간격이 바이트 수로 계산되는 오프셋
	"character_length"	정수	커서가 위치하고 있는 문자의 바이트 수. 커서가 2-바이트 문자에 위치해 있을 경우, 이 값은 2입니다.
	"character_index"	정수	커서 위치와 문자의 첫번째 열 사이의 거리. 커서가 ASCII 문자에 위치해 있을 경우, 이 값은 0입니다. 커서가 2-바이트 문자의 두번째 열에 위치해 있을 경우, 이 값은 1입니다.
표시기 변수	"byte-offset"	정수	오프셋은 표시기(marker)의 바이트와 행의 시작 위치와의 간격이 됩니다.
	"character_length"	정수	표시기가 위치하고 있는 문자의 바이트의 수. 커서가 2-바이트 문자에 위치해 있을 경우, 이 값은 2입니다.
	"character_index"	정수	문자의 첫번째 열과 표시기 위치 사이의 거리. 커서가 ASCII 문자에 위치해 있을 경우, 이 값은 1입니다.
윈도우 변수	"character_index"	정수	HTPU의 내부 커서 위치와 화면 커서 위치 사이의 거리.이 거리는 화면 커서 모드가 ON으로 설정되어 있는 경우에만 0이 아닙니다.

표 2-5 GET_INFO에서 parameter1으로 사용되는 핵심어

parameter1	parameter2	parameter3	복귀값	복귀값에 대한 설명
SYSTEM	"FILL_NOT_BEGIN"		문자열	SET(FILL_NOT_BEGIN) 내장 프로시저에 의해 지정되는 문자열.
SYSTEM	"FILL_NOT_END"		문자열	SET(FILL_NOT_END) 내장 프로시저에 의해 지정되는 문자열.
SYSTEM	"MARGIN_ALLOWANCE"		정수	SET(MARGIN-ALLOWANCE) 내장 프로시저에 의해 지정되는 값.
SYSTEM	"ALIGNMENT_DEFAULT"	"CURSOR_HORIZONTAL","CURSOR_VERTICAL", 또는 "SCROLL"	핵심어	핵심어가 CHARACTERS 또는 BYTES인, 지정된 내장 프로시저의 현재 ALIGNMENT_DEFAULT
SYSTEM	"UNIT_DEFAULT"	"SUBSTR" "INDEX" "LENGTH" 또는 "CURRENT_OFFSET"	핵심어	핵심어가 CHARACTERS 또는 BYTES인, 지정된 내장 프로시저의 현재 UNIT_DEFAULT.

표 2-5 (계속) GET_INFO에서 parameter1으로 사용되는 핵심어

parameter1	parameter2	parameter3	복귀값	복귀값에 대한 설명
화면	"DISPLAY_CURSOR"		핵심어	화면 커서 모드가 ON으로 설정되어 있으면 ON, 화면 커서 모드가 ON으로 설정되어 있지 않거나 또는 OFF로 바뀌면 OFF로 복귀합니다.

오류신호

TPU$_BADREQUEST	WARNING	두번째 매개변수에 의해 표시된 요청이 첫번째 매개변수의 데이터 유형과 일치하지 않습니다.
TPU$_BADKEY	WARNING	첫번째 매개변수로 틀린 핵심어 값 또는 인식될 수 없는 데이터 유형이 전달되었습니다.
TPU$_NOCURRENTBUF	WARNING	현재 버퍼가 정의되지 않았습니다.
TPU$_NOKEYMAP	WARNING	키맵(keymap)이 정의되지 않았습니다.

TPU$_NOKEYMAP-LIST	WARNING	키맵 목록이 정의되지 않았습니다.
TPU$_INVPARAM	ERROR	매개변수의 데이터 유형이 부정확합니다.
TPU$_NEEDTO-ASSIGN	ERROR	GET_INFO 내장 프로시저는 할당문의 오른쪽에만 사용될 수 있습니다.
TPU$_NOBREAK-POINT	WARNING	이 문자열 상수는 오직 구분점 (break point) 다음에만 사용가능합니다.
TPU$_NONAMES	WARNING	요청된 것과 일치되는 이름이 없습니다.
TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_UNKKEYWORD	ERROR	매개변수로 틀린 핵심어가 사용되었습니다.

예

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

이 할당문은 행의 시작 위치에서부터 버퍼의 현재 편집 위치까지의 바이트 오프셋값을 변수 buffer_byte_offset에 할당합니다.

INDEX

이 내장 프로시저는 문자열 내에서 문자 또는 부문자열을 찾아 그것의 문자열 내에서의 위치값을 알려줍니다.

형식

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

매개변수

string1
찾고자 하는 문자나 부문자열이 들어있는 문자열입니다.

string2
string1 내에 위치시키고자 하는 가장 왼쪽 문자 또는 부문자열입니다.

keyword
다음 핵심어 중에서 하나를 지정할 수 있습니다.

- CHARACTERS - 문자열의 시작 위치부터 부문자열 위치까지의 문자수 값을 알려줍니다.
- BYTES - 문자열의 시작 위치부터 부문자열 위치까지의 바이트

수 값을 알려줍니다.

어떤 핵심어도 지정하지 않으면, 기본값은 CHARACTERS입니다. 또한 SET(UNIT_DEFAULT) 내장 프로시저를 사용하여 기본값을 변경할 수도 있습니다. 자세한 설명은 SET(UNIT_DEFAULT) 항목을 참조하십시오.

설명

INDEX에 대한 기본적인 설명은 DEC TEXT Processing Utility Reference Manual을 참조하십시오.
HTPU에서 핵심어 매개변수를 추가하여 문자의 위치를 바이트 또는 문자수로 취하도록 선택할 수 있습니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 부정확합니다.
TPU$_NEEDTO-ASSIGN	ERROR	INDEX는 할당문의 오른쪽에서 사용해야 합니다.

예

```
1 char_loc:=INDEX('1 2 3 4 5 6 7' , '6 7', CHARACTERS)
```

이 할당문은 변수 char_loc에 정수값 6을 할당합니다. 왜냐하면, 문자열 '1 2 3 4 5 6 7'에서 부문자열 '6 7'이 문자위치 6에서 시작되기 때문입니다.

```
2 loc:=INDEX('1 2 3 4 5 6 7' , '6 7', BYTES)
```

이 할당문은 변수 loc에 정수값 11을 할당합니다. 왜냐하면, 부문자열 '6 7'이 문자열 '1 2 3 4 5 6 7'의 바이트 위치 11에서 시작되기 때문입니다.

IS_CLASS

이 내장 프로시저는 지정한 문자열의 문자 종류가 핵심어에 의해 지정된 종류에 속하는지를 조사합니다. IS_CLASS는 조건에 맞으면 1, 아니면 0을 복귀합니다.

형식

integer:=IS_CLASS (string, keyword)

매개변수

string
인용된 문자열, 문자열 상수를 표현하는 변수명 또는 문자열을 표시하는 표현식입니다. 문자열 길이가 한 문자를 초과하면, 첫번째 문자만 검사합니다.

keyword
문자 종류를 표시하는 핵심어입니다. 다음 중 하나를 지정할 수 있습니다.

- ASCII_CHAR - (반자 문자의) ASCII 문자
- FULL_FORM - 2-바이트 전자 문자

- NON_HANGUL - 이 핵심어에 대한 설명은 이 지침서의 그림 1-1를 참조하십시오.

- HANGUL - 이 핵심어에 대한 설명은 본 지침서의 그림 1-1를 참조하십시오.

설명

이 내장 프로시저는 지정한 입력 문자열 내 첫번째 문자의 종류가 핵심어 매개변수에서 지정한 종류에 속하면 1을 복귀하고, 그렇지 않으면 0을 복귀합니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 부정확합니다.
TPU$_NEEDTO-ASSIGN	ERROR	IS_CLASS는 할당문의 오른쪽에 있어야 합니다.

JUSTIFY

이 내장 프로시저는 지정된 버퍼 또는 범위 내의 원문을 왼쪽, 오른쪽 또는 양쪽으로 들여쓰도록 재형식화합니다.

형식

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

매개변수

buffer
정렬할 원문이 있는 버퍼.

range
정렬할 원문의 범위.

keyword
핵심어는 다음의 값들을 갖습니다.

- JUSTIFY_LEFT - 최초의 행을 제외하고, 모든 행을 integer1 또는 왼쪽 가장자리에서 시작되도록 조정합니다. 이 값이 기본값입니다.

- JUSTIFY_RIGHT -마지막 행을 제외하고, 모든 행을 integer1 또는 오른쪽 가장자리에서 끝나게 조정합니다.

- JUSTIFY_BOTH - 최초 행을 제외한 모든 행을 integer1 또는 왼쪽 가장자리에서 시작하도록, 그리고 마지막 행을 제외한 모든 행을 integer2 또는 오른쪽 가장자리에서 끝나게 조정합니다.

integer1
왼쪽 가장자리 값. 왼쪽 가장자리 값은 1보다는 크고, 오른쪽 가장자리보다는 작아야 합니다. 이 값의 기본값은 버퍼의 왼쪽 가장자리 값입니다.

integer2
오른쪽 가장자리 값. 오른쪽 가장자리 값은 왼쪽 가장자리보다는 크고 버퍼의 최대 레코드 크기를 초과할 수 없습니다. 이 값의 기본값은 버퍼의 오른쪽 가장자리 값입니다.

integer3
최초 행 들여쓰기 값. 이 값은 최초 행의 왼쪽 가장자리를 수정합니다. 이 값은 양수 또는 음수일 수 있습니다. 그러나, 왼쪽 가장자리에 이 값을 더한 결과가 1보다는 크고 오른쪽 가장자리보다는 작아야 합니다. 이 값의 기본 값은 0입니다.

설명

JUSTIFY는 1-바이트 문자와 복수 바이트 문자 사이의 공백

(%X20)과 인터페이스를 단어 분리자로서만 인식합니다. 이것은 레코드의 가장자리 설정과 레코드의 분리, 그리고 레코드의 추가를 수행하도록, 처음에는 FILL 내장 프로시저를 가장자리 허용값이 0인 상태로 호출합니다.

JUSTIFY는, 행이 정렬 중인 범위 또는 버퍼의 최초 행이 아닌 한, 행의 왼쪽 가장자리를 왼쪽 가장자리로 정렬시킵니다. 이 경우에, JUSTIFY는 행의 왼쪽 가장자리를, 왼쪽 가장자리에 최초 행 들여쓰기 만큼 더하여 정렬시킵니다. 그러나, 범위 정렬시에 범위가 행의 맨 처음 위치에서 시작하지 않으면, JUSTIFY가 해당 행의 왼쪽 가장자리를 변경하지 않습니다. 그리고, 이 경우에는 최초행 들여쓰기 값을 0으로 지정해 주어야 합니다.

좌측 정렬만 필요한 경우에는 FILL을 호출한 후에 JUSTIFY 내장 프로시저로 복귀합니다. 우측 정렬이 필요한 경우에는, 각 레코드의 맨 앞에 공백을 삽입합니다. 그러나, 범위가 지정되고 시작 오프셋이 0이 아닌 경우에는 범위의 첫번째 레코드의 시작 오프셋(start offset)에 공백이 삽입됩니다.

좌측 정렬과 우측 정렬이 모두 필요한 경우에는, 공백을 포함한 위치, 또는 ASCII 문자와 복수 바이트 문자 사이의 위치에 공백이 삽입됩니다. 이러한 위치가 없는 경우에는 공백이 삽입되지 않고, 범위/버퍼가 오른쪽으로 정렬되지 않습니다. 범위가 정렬되려면, 시작 오프셋 다음에만 공백이 삽입됩니다. 마지막 행이 오른쪽으로 정렬되지 않는 경우에는, 오른쪽 가장자리를 넘을 수도 있습니다.

JUSTIFY 내장 프로시저는 정렬시 버퍼에 공백 문자를 삽입하고 정렬 전에 압축을 수행하지 않습니다. 그러므로써, 이전에 삽입된 공

백이 데이터로 취급되기 때문에, 상이한 가장자리 설정 값을 갖는 JUSTIFY를 연속적으로 호출하였을때 단어가 재배치 되는 것을 방지합니다. 따라서, 정렬의 가장자리 값을 변경하려면 JUSTIFY를 다시 호출하기 전에 압축하기 위하여 EDIT 를 호출해야 합니다.

JUSTIFY는 정렬을 완료한 후에 정렬된 원문의 맨 끝에 커서를 위치시킵니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_ARGMISMATCH	ERROR	JUSTIFY의 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_BADMARGINS	WARNING	가장자리 채우기(fill margins) 중의 하나가 틀리게 지정되었습니다.
TPU$_INVPARAM	ERROR	JUSTIFY의 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_NOTMODIF-IABLE	WARNING	수정 불가형 버퍼내의 원문을 변경할 수 없습니다.

TPU$_CONTROLC	ERROR	JUSTIFY 수행 중 CTRL/C를 눌렀습니다.
TPU$_NOCACHE	ERROR	새로운 캐쉬를 할당할 만큼 충분한 메모리가 없습니다.

예

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

이 문장은 현재 버퍼를 왼쪽 가장자리와 오른쪽 가장자리에 정렬시킵니다. 최초 행은 1 + 10 (=11)번째 열에서 시작하고, 마지막 행은 오른쪽으로 정렬되지 않습니다.

KEY_NAME

이 내장 프로시저는 키 또는 키 조합을 위한 HTPU 핵심어를 복귀합니다.

형식

keyword2:=KEY_NAME ({ integer / key-name / string } ,
[,keyword][,
{ FUNCTION / KEYPAD }])

매개변수

integer
키를 위한 핵심어의 정수 표현 또는 HTPU가 DEC 다국 문자 세트의 문자값으로 해석하는 0에서 255사이의 정수값, 또는 HTPU가 DEC KS C 5601-1987 문자 코드로 해석하는 정수값입니다.

key_name
키의 HTPU 이름인 핵심어

string
주 키보드로부터 입력되는 키 값 문자열

keyword
이 핵심어는 지정된 키를 수정하여 다음의 값들을 취할 수 있습니다.

- SHIFT_KEY - HTPU SHIFT 키를 포함하도록 생성된 키 명칭을 지정하며, HEVE의 GOLD 키로도 불립니다. 복수 바이트 문자는 이 핵심어 수정자만을 받아들입니다. 다른 핵심어 값은 복수 바이트 문자에서 사용될 때 무시됩니다.

- SHIFT_MODIFIED - 키보드 상에서 SHIFT로 표시된 키를 포함하도록 생성된 키 명칭을 지정합니다. 이것은 복수 바이트 문자에 적용될 때 무시됩니다.

- ALT_MODIFIED - ALT 키를 포함하도록 생성된 키 명칭을 지정합니다. 이것은 복수 바이트 문자에 적용될 때 무시됩니다.

- Ctrl_MODIFIED - Ctrl 키를 포함하도록 생성된 키 명칭을 지정합니다. 이것은 복수 바이트 문자에 적용될 때 무시됩니다.

- HELP_MODIFIED - Help 키를 포함하도록 생성된 키 명칭을 지정합니다. 이것은 복수 바이트 문자에 적용될 때 무시됩니다.

FUNCTION
결과 키 이름으로 기능 키(function key)의 이름을 지정하는 매개변수

KEYPAD
결과 키 이름으로 키패드 키(keypad key)의 이름을 지정하는 매개변수

설명

KEY_NAME 내장 프로시저의 기본 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

매개변수 문자열에 복수 바이트 문자를 지정할 수 있습니다.또한, 키를 지정하는데 복수 바이트 문자를 사용할 수도 있습니다.

오류 신호

TPU$_INCKWDCOM	WARNING	핵심어 조합 불일치
TPU$MUSTBEONE	WARNING	문자열의 길이는 한 문자여야함.
TPU$_NOTDEFIN-ABLE	WARNING	두번째 매개변수가 키에 대한 유효한 참조가 아님.
TPU$_NEEDTO-ASSIGN	ERROR	KEY_NAME은 지정 문의 오른편에 있어야함.

TPU$_ARGMISMATCH	ERROR	틀린 유형의 데이터가 내장 KEY_NAME으로 발송됨.
TPU$_BADKEY	ERROR	KEY_NAME 이 SHIFT_KEY, FUNCTION, 또는 KEYPAD를 핵심어 매개변수로 받아들임.
TPU$_TOOFEW	ERROR	내장 KEY_NAME으로 전달된 매개변수가 너무 적음.
TPU$_TOOMANY	ERROR	내장 KEY_NAME으로 전달된 매개변수가 너무 많음.

LENGTH

이 내장 프로시저는 문자열 또는 범위 내의 문자 수나 바이트 수를 표시하는 정수를 복귀합니다.

형식

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

매개변수

buffer
길이를 알고자 하는 버퍼의 이름입니다. 버퍼를 지정할 경우, 행 종료자(line terminators)는 문자로 계산되지 않는다는 점에 주의해야 합니다.

range
길이를 알고자 하는 범위의 이름입니다. 범위를 지정할 경우, 행 종료자는 문자로 계산되지 않는다는 점에 주의해야 합니다.

string
길이를 알고자 하는 문자열입니다.

keyword

다음 중 하나의 핵심어를 지정할 수 있습니다.

- CHARACTERS - 문자열의 길이를 문자 수로 복귀합니다.
- BYTES - 문자열의 길이를 바이트 수로 복귀합니다.

어떤 핵심어도 표시하지 않으면 기본값은 CHARACTERS입니다. 그러나 SET(UNIT_DEFAULT) 내장 프로시저로 기본값을 변경할 수도 있습니다. 자세한 설명은 SET(UNIT_DEFAULT)를 참조하십시오.

설명

LENGTH 내장 프로시저에 대한 기본 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

HTPU에서 핵심어 매개변수를 추가하여 문자열의 길이를 문자 단위 또는 바이트 단위로 복귀할 수 있도록 선택할 수 있습니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.

TPU$_NEEDTOASSIGN	ERROR	LENGTH는 할당문의 오른쪽에 사용해야 합니다.
TPU$_ARGMISMATCH	ERROR	매개변수의 데이터 유형이 부정확합니다.
TPU$_CONTROL	ERROR	LENGTH 수행중에 CTRL/C를 눌렀습니다.

MARK

이 내장 프로시저는 현재의 버퍼에서 현재의 문자에 설정된 표시기를 복귀합니다. 화면상에서 표시기를 표시하는 방법(NONE, REVERSE, BOLD, BLINK, 또는 UNDERLINE)을 지정해야 합니다.

형식

marker:=MARK (keyword [, { buffer / window } [,integer1

[,integer2]]])

매개변수

keyword
다음 핵심어 중 하나를 사용해야 합니다.

- BLINK - 표시기가 점멸되게 합니다.
- BOLD - 표시기가 진하게 표시되게 합니다.
- FREE_CURSOR - 표시기가 자유 표시기가 되게 합니다(즉 표시기가 문자에 제한받지 않습니다). MARK(FREE_CURSOR)가 실행될 때, 커서 위치가 행의 시작 위치 앞, 행의 끝 뒤, 탭의 중간, 또는 버퍼의 하단 아래에 있지 않는 한, 매개변수

FREE_CURSOR를 지정하더라도 자유 표시기가 생성되지 않습니다. 자유 표시기는 영상 속성이 없습니다.

- NONE - 표시기에 어떠한 영상 속성도 적용되지 않도록 지정합니다.

- REVERSE - 표시기가 반전 영상으로 표시되게 합니다.

- UNDERLINE - 표시기가 밑줄로 표시되게 합니다.

buffer
표시기가 위치할 버퍼. 기본적으로, HTPU는 표시기를 현재 버퍼에 위치시킵니다.

window
표시기가 위치할 버퍼로 사상되는 윈도우. 윈도우가 버퍼로 사상되는 경우에는 윈도우 변수를 지정할 수 있습니다. 기본적으로, HTPU는 표시기를 현재 버퍼에 위치시킵니다.

integer1
표시기가 위치할 화면 열 정수. 이 정수에 1부터 32,767 까지 지정할 수 있습니다. 기본적으로는 현재 화면 열 (screen column) 에 해당하는 버퍼 오프셋 (buffer offset) 에 표시기를 위치시킵니다.

integer2
표시기가 위치할 버퍼 내의 레코드에 대한 정수. 기본적으로는 현재 레코드에 표시기가 위치합니다.

설명

MARK 내장 프로시저의 기본 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

표기시는 문자 경계에 설정됩니다. 따라서, 커서가 복수 바이트 문자에 위치해 있는 경우에는, 현재 편집점이 복수 바이트 문자의 두 번째 열에 있어도 표시기 설정이 문자의 첫번째 열에 설정됩니다.

복수 바이트 문자에 비디오 속성을 표시하면 복수 바이트 문자 전체에 해당 비디오 속성이 적용됩니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_NEEDTO-ASSIGN	ERROR	MARK는 할당문의 오른쪽에만 될 수 있습니다.
TPU$_NOCURRENT-BUF	WARNING	표시기를 설정하려면 버퍼에 위치해 있어야 합니다.

TPU$_INVPARAM	ERROR	하나 이상의 매개변수 유형이 틀립니다.
TPU$_BADKEY	ERROR	핵심어가 NONE, BOLD, BLINK, REVERSE, UNDER_LINE 또는 FREE_CURSOR 이어야 합니다.
TPU$_UNKEYWORD	ERROR	인식되지 않는 핵심어를 지정하였습니다.
TPU$_INSVIRMEM	FATAL	표시기를 만들기에 충분한 기억장소가 없습니다.

READ_CHAR

이 내장 프로시저는 키보드로부터 입력되는 다음 문자가 들어있는 문자열을 복귀합니다.

형식

string:=READ_CHAR

복귀값

키보드로부터 입력되는 문자가 들어있는 문자열 유형의 변수

설명

READ_CHAR에 대한 기본적인 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

HTPU는 READ_CHAR 내장 프로시저에서 복수 바이트의 문자 입력이 가능하도록 향상되었습니다.

오류신호

TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_NEEDTOASSIGN	ERROR	READ_CHAR는 할당문의 오른쪽에만 사용할 수 있습니다.

READ_KEY

이 내장 프로시저는 사용자가 키를 누를 때까지 기다리고 있다가, 키를 누르면 해당키의 이름을 복귀합니다. READ_KEY는 또한 에스케이프 시퀀스(escape sequence)와 제어 문자(control character)도 처리합니다.

형식

keyword:=READ_KEY

복귀값

지금 막 누른 키의 키 이름이 들어있는 값을 복귀합니다.

설명

READ_KEY 내장 프로시저에 대한 기본적인 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.
READ_KEY는 복수 바이트 문자의 입력이 가능하도록 향상되었습니다. 입력된 문자가 복수 바이트 문자이면, 해당 문자에 대한 핵심어가 복귀됩니다.

SCROLL

이 내장 프로시저는 버퍼에 있는 원문의 행을 지정된 행 수 만큼 화면에서 아래위로 이동시킵니다.

형식

[integer 2:=]SCROLL (window[,integer1[,keyword]])

매개변수

window
화면 이동시키려는 원문이 있는 버퍼에 할당된 윈도우입니다.

integer1
화면 이동시키려는 원문의 행 수를 가리키는 부호(+/-)있는 정수값입니다. 양수를 지정하면 윈도우에 있는 원문은 화면의 아래에서 위로 이동되며, 음수를 지정하면 화면의 위에서 아래로 이동됩니다. 그리고, 정수값으로 0을 지정하면 화면 이동이 일어나지 않습니다.

이 매개변수는 생략될 수 있습니다. 이 매개변수를 생략하면 원문은 버퍼경계(buffer boundary)에 도달할 때까지, 또는 키를 누르고 있는 동안 계속 이동됩니다. 버퍼의 현재 방향이 FORWARD이면 원문이 버퍼의 끝까지 이동되며, 버퍼의 현재 방향이 REVERSE 이

면 원문이 버퍼의 맨 앞까지 이동됩니다. 명령과 연관된 키를 누르면, 화면 이동이 멈추어지고, HTPU가 그 키에 연관된 명령을 실행합니다.

keyword
다음과 같은 핵심어를 지정할 수 있습니다.

- CHARACTERS - 화면 이동 후에 커서가 복수 바이트 문자의 첫번째 바이트가 아닌 곳에 위치하면, CHARACTERS가 커서를 첫번째 열로 이동시킵니다.

- BYTES - 화면 이동 후에, 아무런 이동 없이 커서 위치로 온 문자의 바이트 위치에 커서가 머무릅니다. 그러나, 커서표시 상태로 설정되어 있다면 내부적인 커서의 위치가 복수 바이트 문자에서 첫 바이트가 아닌 곳에 있게된다 할지라도 그 화면 커서는 항상 복수바이트 문자의 문자 경계에 위치하게 됩니다.

기본 값은 BYTES입니다. SET(ALIGNMENT_DEFAULT) 내장 프로시저로 기본값을 변경할 수 있습니다. 자세한 설명은 SET(ALIGNMENT_DEFAULT)를 참조하십시오.

복귀값

복귀값은 SCROLL을 수행한 결과 실제로 화면 이동된 행수를 표시합니다.

설명

SCROLL 내장 프로시저에 대한 기본적인 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

HTPU는 세번째 매개변수로 지정되는 BYTES나 CHARACTERS 핵심어를 수행할 수 있도록 향상되었습니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	하나 이상의 매개변수 유형이 틀립니다.
TPU$_CONTROLC	ERROR	화면 이동을 멈추기 위하여 CTRL/C를 눌렀습니다.
TPU$_WINDNOT-MAPPED	WARNING	사상(map)되지 않은 윈도우를 화면 이동하려고 하였습니다.

SEARCH

이 내장 프로시저는 버퍼 또는 범위 내의 특정한 문자 배열을 탐색하여 그 문자를 포함하는 범위 값을 복귀시킵니다.

형식

```
[range2 :=]SEARCH (pattern, { FORWARD } [,
                            { REVERSE }

                  { EXCAT    }      [, { buffer }
                  { NO_EXACT }         { range1 }
                  { integer  }
                  ]] )
```

매개변수

pattern
자세한 사항은 DEC Text Processing Utility Reference Manual 을 참조하십시오.

FORWARD
전진 방향으로 탐색되도록 합니다.

REVERSE

역 방향으로 탐색되도록 합니다.

EXACT
문자 SEARCH가 SEARCH의 첫번째 매개변수로 사용된 패턴과 동일한 문자형과 문자크기(전자 또는 반자)와 반드시 일치되도록 합니다.

NO_EXACT
문자 SEARCH가 문자형이나 문자크기(전자 및 반자 알파벳 문자를 동일하게 취급)와 반드시 일치하지 않아도 되도록 합니다. 이것이 기본값입니다.

integer
다른 속성은 무시하고 어떤 한 속성에 일치시키려는 경우에, SEARCH가 어떻게 문자형과 문자크기(전자/반자) 정보를 처리하게 할 것인지를 지정합니다. Digital 사는 이 정수를 지정하는데 사용가능하도록 사전 정의된 상수들을 사용할 것을 권고합니다. 사전 정의되어 있는 상수들은 다음과 같습니다.

- TPU$K_SEARCH_CASE - 정수 1과 같습니다. 이것은 탐색시, 첫번째 매개변수의 문자형과는 일치되어야 하지만 첫번째 매개변수의 문자크기(전자/반자)와는 일치하지 않아도 되도록 지정합니다.

- TPU$K_SEARCH_FORM - 정수 4와 같습니다. 이것은 탐색시, HTPU에서, 첫번째 매개변수의 문자크기(전자/반자)와는 일치되어야 하지만 첫번째 매개변수의 문자형과는 일치하지 않아도 되도록 지정합니다.

핵심어의 효과를 결합시키려면, 사전 정의된 정수의 변수 값들을 단순히 더해주기만 하면 됩니다. 예를 들어, 문자 크기와 문자형 모두를 일치되게 하려면 "정수" 매개변수 값을 TPU$K_SEARCH_CASE + TPU$K_SEARCH_FORM과 같이 지정해 주면 됩니다.

HTPU는 구분발음부호(diacritical mark)가 적용되지 않기 때문에, TPU$K_SEARCH_DIACRITICAL를 그 자체만으로 지정하거나 또는 TPU$K_SEARCH_CASE와 함께 지정하여도 무시됩니다. 그리고 그것은 TPU$K_SEARCH_FORM과 함께 지정되면 오류가 되어 TPU$_MAXVALUE가 신호 처리됩니다.

buffer
탐색할 버퍼.

range1
탐색할 범위.

복귀값
지정된 패턴을 일치시킨 원문을 포함하여 범위로 되돌아갑니다.

설명

SEARCH 내장 프로시저의 기본 설명은 DEC Processing Utility

Reference Manual을 참조하십시오.

HTPU는 SEARCH 내장 프로시저가 탐색될 패턴의 문자형과 문자 크기를 선택적으로 처리하도록 기능이 향상되었습니다.

오류신호

TPU$_STRNOTFOUND	WARNING	성공적으로 처리되지 않은 문자열 또는 패턴을 탐색합니다.
TPU$_TOOFEW	ERROR	SEARCH에 최소한 두 개의 매개변수가 필요합니다.
TPU$_TOOMANY	ERROR	SEARCH는 4개 까지의 매개변수만을 허용합니다.
TPU$_ARGMISMATCH	ERROR	SEARCH의 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_INVPARAM	ERROR	SEARCH의 매개변수 중 하나의 데이터 유형이 틀립니다.
TPU$_BADKEY	WARNING	SEARCH에 틀린 핵심어를 지정하였습니다.
TPU$_MINVALUE	WARNING	SEARCH의 정수 매개변수는 -1보

		다 크거나 같아야 합니다.
TPU$_MAXVALUE	WARNING	SEARCH의 정수 매개변수는 5보다 작거나 같아야 합니다.
TPU$_NOCURR-ENTBUF	ERROR	탐색할 버퍼 또는 범위를 지정하지 않았으면, 탐색 전에 버퍼를 정해야 합니다.
TPU$_CONTROLC	ERROR	SEARCH의 실행 중에 Ctrl/C를 눌렀습니다.
TPU$_ILLPATAS	ERROR	SEARCH 패턴이 현재 문맥에 정의되어 있지 않은 변수에 할당된 패턴 부분을 포함하고 있습니다.

SELECT

이 내장 프로시저는 현재 버퍼의 편집 위치에 표시기(marker)를 설정하고 그 표시기를 복귀합니다. 표시기가 화면에 표시되는 방법(NONE, REVERSE, BOLD, BLINK 또는 UNDERLINE)을 지정해야 합니다.

SELECT에 의해 복구되는 표시기는 선택된 범위 내 첫번째 문자의 위치를 표시합니다. 표시기에 대한 지정한 영상 속성은 선택된 범위의 모든 문자에 적용됩니다. 범위 선택에 관한 정보는 SELECT_RANGE 내장 프로시저의 설명을 참조하십시오.

형식

marker:=SELECT (keyword)

매개변수

keyword
이 핵심어는 복귀된 표시기가 화면에 표시되는 방법을 지정합니다. 다음의 핵심어 중 하나를 사용하여야 합니다.

- BLINK - 화면에서 표시기가 점멸되게 합니다.
- BOLD - 화면에서 표시기가 진하게 표시되게 합니다.
- NONE - 표시기에 어떤 영상 속성도 적용되지 않게 합니다.

- REVERSE - 화면에서 표시기가 역상으로 표시되게 합니다.
- UNDERLINE - 화면에서 표시기가 밑줄로 표시되게 합니다.

설명

SELECT는 선택된 범위의 맨 앞에, 설정된 특별한 표시기가 나타나게 합니다. 표시기는, SELECT 내장 프로시저가 수행되면, 편집 위치인 문자에 위치합니다. 편집 위치가 복수 바이트 문자의 두번째 열에 위치해 있다면, 표시기는 이전 문자와 복수 바이트 문자의 첫번째 열 사이에 위치합니다.

오류신호

TPU$_ONESELECT	WARNING	현재 버퍼에서 SELECT가 이미 사용중입니다.
TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_NEEDTO-ASSIGN	ERROR	SELECT는 할당문의 오른쪽에만 사용할 수 있습니다.

TPU$_NOCUR-RENTBUF	WARNING	SELEC를 사용하기 전에 버퍼에 위치해 있어야 합니다.
TPU$_INVPARAM	ERROR	하나 이상의 지정된 매개변수 유형이 틀립니다.
TPU$_BADKEY	ERROR	SELECT에 틀린 핵심어를 지정했습니다.

SELECT_RANGE

이 내장 프로시저는 SELECT 내장 프로시저에 의해 설정된 표시기와 현재 편집 위치 사이의 모든 문자를 포함하는 범위를 복귀합니다. SELECT_RANGE의 실행시 선택된 범위의 마지막 위치에 현재 문자가 포함되지 않습니다.

형식

range:=SELECT_RANGE

설명

SELECT_RANGE 내장 프로시저의 기본 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

SELECT_RANGE의 실행 중에는, 복수 바이트 문자의 첫번째 이외의 열 위치에 있는 현재 편집점이 첫번째 열에 있을때와 동일하게 동작합니다.

오류신호

TPU$_NOSELECT WARNING 현재 버퍼에서 활동중인 선택 범위

		가 없습니다.
TPU$_SELRANGE-ZERO	WARNING	선택된 범위에 선택된 문자가 하나도 포함되어 있지 않습니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_NEEDTO-ASSIGN	ERROR	SELECT_RANGE는 할당문의 오른쪽에만 쓸 수 있습니다.
TPU$_NOCUR-RENTBUF	WARNING	현재 버퍼에 없습니다.

SET (ALIGNMENT_DEFAULT)

형식

SET (ALIGNMENT_DEFAULT, string, { BYTES / CHARACTERS })

매개변수

ALIGNMENT_DEFAULT
SET가 특정 내장 프로시저를 위하여 커서 정렬을 정의함을 가리키는 핵심어입니다.

string
ALIGNMENT_DEFAULT가 설정될 내장 프로시저의 이름이 들어있는 문자열입니다. 유효한 내장 프로시저는 CURSOR_HORIZONTAL, CURSOR_VERTICAL 및 SCROLL입니다.

BYTES
커서 정렬이 바이트 단위로 되게 하는 핵심어입니다.

CHARACTERS
커서 정렬이 문자 단위로 되게 하는 핵심어입니다.

설명

SET(ALIGNMENT_DEFAULT)는 커서가 복수-바이트 문자의 첫번째 열로 정렬되는지의 여부에 대한 기본값을 지정합니다. CURSOR_HORIZONTAL, CURSOR_VERTICAL, SCROLL에 대한 ALIGNMENT_DEFAULT는 BYTES이며, 이것은 커서가 이들 내장 프로시저에 의해 정렬되지 않게 합니다.

CURSOR_HORIZONTAL, CURSOR_VERTICAL, 그리고 SCROLL도 함께 참조하십시오.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 부정확합니다.

SET(DISPLAY_CURSOR)

형식

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

매개변수

DISPLAY_CURSOR
SET 명령이 HTPU의 화면 커서 모드를 변경한다는 것을 나타내는 핵심어.

ON
화면 커서 모드를 ON 시키도록 지정하는 핵심어.

OFF
화면 커서 모드를 OFF 시키도록 지정하는 핵심어.

복귀값

복귀 값은 이 프로시저를 적용하기 전에 HTPU의 화면 커서 모드를 나타냅니다.

설명

이 내장 프로시저는 HTPU의 화면 커서 모드를 ON 또는 OFF 시킵니다. 그리고 선택적으로 화면 커서 모드의 현재 설정값을 취합니다.

화면 커서 모드가 ON 상태에 있으면, 커서가 복수 바이트 문자에 위치해 있을 경우 화면에서 보이는 커서 (화면 커서라 합니다)는 HTPU에 저장되어 있는 커서 (내부 커서라 합니다)와 다를 수 있습니다.

예를 들어, 내부 커서가 복수 바이트 문자의 두번째 바이트 위치에 있는 경우에는 화면 커서 모드가 OFF 상태이면 화면 커서가 문자의 두번째 열에 있게 되고, 화면 커서 모드가 ON 상태이면 문자의 첫번째 열에 있게 됩니다.

화면 커서 모드는 화면 커서가 항상 문자에 묶이도록 합니다. 화면 커서 모드가 ON 상태이고 내부 커서가 복수 바이트 문자의 첫번째 이외의 바이트에 위치해 있으면, HTPU의 문자 삽입과 겹쳐쓰기 (overstriking) 동작이 내부 커서가 마치 화면 커서 위치에 있는 것처럼 수행되는데, 이는 커서가 문자에 묶여 있는 경우에는 항상 그렇게 됩니다.

화면 커서 모드는 HTPU가 기동될 때에는 기본적으로 OFF 로 설정되지만, HEVE 기동 프로시저로 ON으로 설정할 수 있습니다.

SET(FILL_NOT_BEGIN)

형식

SET (FILL_NOT_BEGIN, string)

매개변수

FILL_NOT_BEGIN
SET이 FILL 과 JUSTIFY 의 실행에 의해서 행의 맨 앞에 올 수 없는 예외 문자를 정의하는 것을 가리키는 핵심어입니다.

string
행의 맨 앞에 올 수 없는 예외 문자가 들어있는 문자열입니다.

설명

SET(FILL_NOT_BEGIN)은 FILL과 JUSTIFY 내장 프로시저를 위한 FILL_NOT_BEGIN 문자를 지정합니다. FILL과 JUSTIFY 는 FILL_NOT_BEGIN 문자가 행의 맨 앞에 나타나지 않게 행을 분리하는 위치를 결정합니다.

FILL, JUSTIFY, SET(FILL_NOT_END) 와 SET(MARGIN_ALLOWANCE) 를 참조하십시오.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 부정확합니다.

SET(FILL_NOT_END)

형식

SET (FILL_NOT_END,string)

매개변수

FILL_NOT_END
SET이 FILL 과 JUSTIFY 의 실행에 의해서 행의 끝에 올 수 없는 예외 문자를 정의하는 것을 가리키는 핵심어입니다.

string
행의 끝에 올 수 없는 예외 문자가 들어있는 문자열입니다.

설명

SET(FILL_NOT_END)는 FILL과 JUSTIFY 내장 프로시저를 위한 FILL_NOT_END 문자를 지정합니다. FILL과 JUSTIFY 는 FILL_NOT_END 문자가 행의 맨 끝에 나타나지 않게 행을 분리하는 위치를 결정합니다.

FILL, JUSTIFY, SET(FILL_NOT_BEGIN)과 SET(MARGIN_ALLOWANCE)를 참조하십시오.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 부정확합니다.

SET(MARGIN_ALLOWANCE)

형식

SET (MARGING_ALLOWANCE, integer)

매개변수

MARGIN_ALLOWANCE
SET이 FILL의 실행 중에 오른쪽 가장자리를 넘어서 위치될 수 있는 예외 문자의 수를 정의하는 것을 가리키는 핵심어입니다.

integer
오른쪽 가장자리를 넘어 위치할 수 있는 예외 문자의 수입니다.

설명

SET(MARGIN_ALLOWANCE)는 오른쪽 가장자리 너머에 위치할 수 있는 예외 문자의 수를 지정합니다. FILL 내장 프로시저는 예외문자(FILL_NOT_BEGIN과 FILL_NOT_END 문자)도 포함하여 행을 채웁니다.

기본 값은 0입니다. 값이 0일 경우 FILL_NOT_BEGIN 문자는 오른쪽 가장자리 너머에 있을 수 없습니다. FILL_NOT_BEGIN 문자가 행의 처음에 나오는 것을 방지하기 위해 몇몇 문자는 앞 행의 끝에서 현재 행의 처음으로 이동됩니다. 따라서 우측 마진까지 꽉차지 않은 행이 있을 수 있습니다. 그러므로 행을 오른쪽 가장자리까지 꽉 차지 않도록 하는 것이 가능합니다.

FILL, SET(FILL_NOT_BEGIN), 및 SET(FILL_NOT_END)도 함께 참복하십시오.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 부정확합니다.

SET(UNIT_DEFAULT)

형식

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

매개변수

UNIT_DEFAULT
SET이 특정 내장 프로시저를 위하여 계수 단위(counting unit)를 정의함을 가리키는 핵심어입니다.

string
UNIT_DEFAULT가 설정될 내장 프로시저의 이름이 들어있는 문자열입니다. 유효한 내장 프로시저는 CURRENT_OFFSET, INDEX, LENGTH 및 SUBSTR입니다.

BYTES
계수 단위를 바이트로 지정하는 핵심어입니다.

CHARACTERS
계수 단위를 문자로 지정하는 핵심어입니다.

설명

SET(UNIT_DEFAULT)는 기본 계수 단위를 바이트나 문자로 지정합니다. CURRENT_OFFSET, INDEX, LENGTH 및 SUBSTR에 대한 UNIT_DEFAULT는 CHARACTERS입니다.

CURRENT_OFFSET, INDEX, LENGTH 및 SUBSTR도 함께 참조하십시오.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_INVPARAM	ERROR	매개변수의 유형이 부정확합니다.

SPLIT_LINE

이 내장 프로시저는 현재의 행을 현 편집 위치에서 두 개의 행으로 분리시킵니다.

형식

SPLIT_LINE

설명

SPLIT_LINE 내장 프로시저에 대한 기본적인 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

현재 편집 위치가 2-바이트 문자의 두번째 열에 있으면, SPLIT_LINE은 현재 문자의 위치가 첫번째 열에 있을 때와 동일한 방식으로 행을 분리시킵니다. SPLIT_LINE의 실행 후에 편집 위치는 그 문자의 첫번째 열에 위치합니다.

오류신호

TPU$_NOCURRENTBUF	WARNING	버퍼에 위치해 있지 않습니다.

TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_NOTMODI-FIABLE	WARNING	수정 불가형 버퍼는 수정할 수 없습니다.
TPU$_NOCACHE	ERROR	새로운 캐쉬를 할당할 메모리가 충분하지 않습니다.

SUBSTR

이 내장 프로시저는 문자열 또는 범위 내에 부문자열을 생성하고, 그 부문자열을 나타내는 문자열을 복귀합니다.

형식

```
                          { buffer  }
String1:=SUBSTR ( integer1 { range   } , integer2 [,
                          { string  }
                            keyword    ])
```

매개변수

buffer
부문자열이 포함된 버퍼

range
부문자열이 포함된 범위

string
부문자열이 포함된 문자열

integer1
부문자열이 시작되는 문자 또는 바이트 위치. 첫번째 문자위치는 1입니다.

integer2
부문자열에 포함될 문자나 바이트의 수

keyword

다음과 같은 핵심어를 사용할 수 있습니다.

- CHARACTERS - 두번째와 세번째 매개변수를 문자 수로 사용합니다.
- BYTES - 두번째와 세번째 매개변수를 바이트 수로 사용합니다.

핵심어를 지정하지 않을 경우의 기본값은 CHARACTERS입니다. 또한 SET(UNIT_DEFAULT) 내장 프로시저를 사용하여 기본값을 변경할 수도 있습니다. 자세한 설명은 SET(UNIT_DEFAULT)를 참조하십시오.

설명

SUBSTR 내장 프로시저에 대한 기본적인 설명은 DEC Text Processing Utility Reference Manual을 참조하십시오.

BYTES를 핵심어로 지정하면 2-바이트 문자열에서의 부문자열은 지정된 바이트의 수에 따라 생성되며, 문자의 중간에서 추출

(extraction)될 경우, 나머지 바이트는 공백 문자로 대체됩니다.

오류신호

TPU$_TOOFEW	ERROR	매개변수가 부족합니다.
TPU$_TOOMANY	ERROR	매개변수가 너무 많습니다.
TPU$_NEEDTO-ASSIGN	ERROR	SUBSTR는 할당문의 오른쪽에만 사용할 수 있습니다.
TPU$_INVPARAM	ERROR	지정된 하나 이상의 매개변수 유형이 틀립니다.
TPU$_ARGMISMATCH	ERROR	SUBSTR의 매개변수 중 하나의 유형이 틀립니다.
TPU$_STRTOOLARGE	ERROR	지정한 길이가 최대로 가능한 문자열의 길이를 초과합니다.

제 3 장

신규 및 수정된 **HEVE** 명령어

이 장은 EVE(Extensible Versatile Editor)의 명령에 새로 추가되거나 향상된 HEVE 명령에 대한 참고 정보를 제공합니다. 이 장은 EVE 참조 메뉴얼 EVE Referemce Manual의 부록으로 사용되어야만 합니다.

이장에서 설명되는 명령은 알파벳 순으로 되어 있습니다. 한글 문자의 사용과 관련이 없는 명령은 이 장에 포함되어 있지 않습니다. 보다 상세한 내용은 EVE Reference Manual을 참조하십시오. 이 장에서 취급되지 않는 명령중 일부도 한글 문자를 처리할 수 있다는 점을 유의하십시오. 한글 문자로 작업하는 것은 ASCII 문자로 작업하는 것과 같습니다. 편집 수행을 위한 HEVE 명령 사용법은 HEVE User Guide를 참조하십시오.

다음 페이지에 있는 표는 이 장에 포함되어 있는 명령어의 이름입니다.

표 **3-1 HEVE**의 신규 및 수정된 명령어

명령어
DELETE
DRAW BOX
DRAW LINE
ERASE CHARACTER
ERASE PREVIOUS WORD
ERASE WORD
EXTEND HEVE
EXTEND HTPU
FILL/FILL PARAGRAPH/FILL RANGE
FILD
FULLFORM WORD
HALFFORM WORD
HTPU
LEFT ADJUST
LEFT INDENT
LOWERCASE WORD
MOVE BY WORD
MOVE DOWN/UP
MOVE LEFT/RIGHT
REPLACE
RIGHT ADJUST

표 3-1(계속) **HEVE**의 신규 및 수정된 명령어

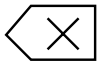
명령어
RIGHT INDENT
SAVE EXTENDED HEVE
SAVE EXTENDED HTPU
SET TABS
SET [NO]DISPLAY CURSOR
SET FIND CASE [NO]EXACT
SET FIND EXACT
SET FIND FORM [NO]EXACT
SET FIND GENERAL
SPECIAL INSERT
SYMBOL
UPPERCASE WORD

일부 명령은 지원되는 단말기의 키들과 연관되어 있습니다. 각 명령에 대한 설명의 형식 부분에 그 명령에 연관된 키를 표시해 놓았습니다. 예를 들면, VT300/VT200 계열은 대응되는 명령이 VT300과 VT200 계열에 속하는 모든 단말기에서 그 아래 키와 연관있다는 것을 표시합니다. 두 종류의 한글 단말기 VT382와 DOOSAN 220은 각각 VT300과 VT200 계열로 구분됩니다.

DELETE

형식

DELETE

VT300, VT200	VT100
	DELETE

설명

커서의 왼쪽에 있는 문자를 삭제합니다. 삽입모드(insert mode)에서는 행의 나머지 문자들이 한 문자만큼 왼쪽으로 이동합니다. 겹쳐쓰기 모드(overstrike mode)에서는 삭제된 문자가 해당 문자의 화면 열 폭(screen column width) 에 해당하는 공백으로 대체됩니다. 즉, 1바이트 문자인 경우에는 1열 폭 만큼, 2바이트 문자인 경우에는 2열 폭만큼의 공백으로 대체됩니다. 행의 맨 처음에서는 이 기능이 캐리지 리턴을 삭제하고 현재 행을 이전 행 위치로 올립니다.

삭제된 부분을 복구하려면, RESTORE CHARACTER 명령을 사용하면 됩니다.

DRAW BOX

형식

DRAW BOX

설명

HEVE 박스 그리기 기능을 호출합니다. 이는 DRAW LINE 명령을 사용한 후 REMOVE 키를 눌러서 박스 그리기 모드로 들어가는 것과 같습니다. 보다 상세한 내용은 DRAW LINE을 참조하십시오.

DRAW LINE

형식

DRAW LINE

설명

HEVE 선 그리기 기능과 박스 그리기 기능을 호출합니다. 이 기능은 간단하면서도 사용자에게 친숙한 인터페이스를 제공합니다. 선과 박스는 커서를 이동하는 대로 표시됩니다. 커서는 상,하,좌,우 화살표 키로 제어합니다. 그리기 기능은 간단한 키조작으로 조정할 수 있습니다. 그리기 상황(drawing status)은 제어키를 누르면 상태 행에서 갱신됩니다.

선 및 박스 그리기의 단계:

1. DO 키를 누르면 명령어 프롬프트가 발신됩니다.

2. 이 명령어 프롬프트에서 DRAW LINE을 입력하십시오. 그러면 HEVE가 선 그리기 모드로 들어갑니다.

3. 이 선 그리기 모드에서 상,하,좌,우 화살표 키를 사용하여 커서를 이동해 가면서 선을 그리거나 지울 수 있습니다.

표 3-2는 선 그리기 용의 모든 제어키를 나타냅니다.

- SELECT 키를 눌러, 그리기 모드 또는 커서 이동 모드를 선택 할 수 있습니다. 커서 이동 모드에서는 화면상의 데이터에 아무런 영향도 미치지 않고 커서를 자유롭게 이동할 수 있습니다.

  INSERT 키를 눌러 굵은 선, 가는 선, 또는 선 지우기를 선택할 수 있습니다. 선 지우기 기능은 커서가 이동하는 대로 선 그리기 문자를 지웁니다.

- REMOVE 키를 누르면 박스 그리기 모드로 들어갑니다. 박스 그리기 모드에서는 제어키의 기능이 선 그리기에서와는 다릅니다.

- RETURN 키를 누르면 선 그리기가 종료됩니다.

4. 박스 그리기 모드에서는 상,하,좌,우 화살표 키를 사용하여 커서를 이동해 가면서 박스를 그릴 수 있습니다. 표3-3은 선 그리기 용의 제어키를 나타냅니다.

   - SELECT 키를 누르면, 박스 그리기를 종료하고 선 그리기로 복귀합니다.

   - REMOVE 키를 누르면, 박스 그리기가 취소되고 선 그리기로 복귀합니다.

   - RETURN 키를 누르면, 박스 그리기와 선 그리기가 모두

종료됩니다.

DO 키도 선 그리기와 박스 그리기 모두를 종료합니다.

선 그리기 문자는 항상 화면의 홀수 열로 정렬됩니다. 각각의 선 그리기 문자는 항상 화면에서 두 개의 열을 차지합니다.

다음 표는 선 그리기와 박스 그리기 모드의 제어키를 요약한 것입니다.

표 3-2 선 그리기 모드에서의 제어키

기능	변환 키
선 그리기/커서 이동	SELECT 또는 KP1
굵은 선/가는 선/선 지우기	INSERT 또는 KP2
박스 그리기로 진입	REMOVE 또는 KP3
선 그리기 종료	RETURN

표 3-3 박스 그리기 모드에서의 제어키

기능	변환 키
박스 그리기 종료	SELECT 또는 KP1
박스 그리기 취소	REMOVE 또는 KP3
박스 및 선 그리기 종료	RETURN

DRAW LINE과 DRAW BOX를 사용한 예제를 참조하시려면 HEVE 사용자 지침서의 5장을 보십시오.

ERASE CHARACTER

형식

ERASE CHARACTER

설명

커서 위치의 문자를 삭제합니다. 삽입 모드에서는 행의 나머지 문자들이 한 문자만큼 왼쪽으로 이동합니다. 겹쳐쓰기 모드에서는 삭제된 문자가 해당 문자의 열 폭에 해당하는 만큼의 공백으로 대체됩니다. 즉, 1바이트 문자인 경우에는 1열 폭 만큼, 2바이트 문자인 경우에는 2열 폭 만큼의 공백으로 대체됩니다. 행의 맨 끝에서는 이 기능이 캐리지 리턴을 삭제하고 다음 행을 현재 행 위치로 올립니다.

삭제된 부분을 복구하려면, RESTORE CHARACTER 명령을 사용하면 됩니다.

ERASE PREVIOUS WORD

형식

ERASE PREVIOUS WORD

설명

커서가 위치한 단어 또는 이전의 단어를 다음과 같이 지웁니다. 커서가 두 개의 단어 사이 또는 단어 시작 부분에 있는 경우, 이전의 단어를 지웁니다. 커서가 단어 중간에 있으면, 그 단어와 어미 공백을 지웁니다. 또 커서가 행의 처음에 있으면, 이전의 행에서 캐리지 리턴만을 삭제하고 현재의 행을 위로 올립니다.

여기서 복수 바이트 한글 데이터는 "단어"의 정의가 변경된 것을 제외하고는 1 바이트 데이터와 같이 취급됩니다. - 한글에서의 단어는 같은 종류로 된 문자가 연속되어 있 는 것을 말합니다. (HTPU의 CHARACTER_CLASS 참조). 이 명령은 현재 문자의 종류가 이전의 문자 종류와 다르거나 또는 행의 끝에 도달했을 경우, 단어의 끝이라고 간주합니다.

지워진 부분을 복구하려면 RESTORE 또는 RESTORE WORD 명령을 사용하면 됩니다.

ERASE WORD

형식

ERASE WORD

VT300, VT200	VT100
CTRL/J	CTRL/J
	keypad COMMA

설명

커서가 위치한 단어를 지웁니다. 이 때 커서가 두 개의 단어 사이에 있으면 다음 단어와 어미 공백을 지웁니다. 커서가 행의 끝부분에 있으면, 현재의 행에서 캐리지 리턴만을 지우고 다음 행을 위로 올립니다.

여기서 한글 복수 바이트 데이터는, "단어"의 정의가 변경된 것을 제외하고는 1 바이트 데이터와 같이 취급됩니다. - 한글에서의 단어는 같은 그룹의 문자가 연속되어 있는 것을 말합니다(HTPU의 CHARACTER_CLASS 참조). 이 명령은 현재 문자의 종류가 이전의 문자 종류와 다르거나 또는 행의 끝에 도달했을 경우, 단어의 끝이라고 간주합니다.

지워진 부분을 복구하려면 RESTORE 또는 RESTORE WORD 명령을 사용하면 됩니다.

EXTEND HEVE

형식

EXTEND HEVE (procedure-name/*)

설명

HEVE를 확장하기 위해 HTPU 프로시저들을 컴파일합니다. 이 명령은 EXTEND HTPU와 동일합니다. 이 기능은 EVE의 EXTEND EVE 기능을 대신하는 것입니다.

매개변수

procedure-name
컴파일할 HEVE 프로시저 이름을 약어로 지정할 수 있습니다. 이때 매치(match)되어야 하는 이름이 하나 이상 있을 경우, HEVE는 사용자가 원하는 프로시저를 선택할 수 있도록 해당되는 이름의 목록을 보여주고 EXTEND HEVE 명령을 다시 호출합니다.

*

버퍼내의 모든 프로시저 및 명령을 컴파일하도록 HEVE에게 지시하는 와일드 카드 기호로 이 매개변수는 EXTEND ALL 명령과 같은 결과를 산출합니다.

예

```
Command : EXTEND HEVE USER_PROC
```

위의 예는 현재 버퍼에서 USER_PROC이라 불리는 프로시저를 컴파일합니다.

EXTEND HTPU

형식

EXTEND HTPU (procedure-name/*)

설명

HEVE를 확장하기 위해 HTPU 프로시저들을 컴파일합니다. 이 명령은 EXTEND HEVE와 동일합니다. 이 기능은 EVE의 EXTEND TPU 기능을 대신하는 것입니다.

매개변수

procedure-name

컴파일 할 HEVE 프로시저 이름을 약어로 지정할 수 있습니다. 이때 매치(match)되어야 하는 이름이 하나 이상 있을 경우, HEVE는 사용자가 원하는 프로시저를 선택할수 있도록 해당되는 이름의 목록을 보여주고 EXTEND HTPU 명령을 다시 호출합니다.

*

버퍼내의 모든 프로시저 및 명령을 컴파일하도록 HEVE에게 지시하는 와일드 카드 기호로 이 매개변수는 EXTEND ALL 명령과 같은 결과를 산출합니다.

예

```
Command : EXTEND HTPU USER_PROC
```

위의 예는 현재 버퍼에서 USER_PROC이라 불리는 프로시저를 컴파일합니다.

FILL/FILL PARAGRAPH/FILL RANGE

형식

FILL/FILL PARAGRAPH/FILL RANGE

설명

한 행에 최대의 단어가 들어갈 수 있도록 현재 버퍼의 가장자리에 맞춰 현재의 단락 또는 선택 범위의 형식을 재설정합니다. FILL RANGE에서는 반드시 범위를 선택하여야 합니다. 이 명령을 사용하려면 :

- 선택적으로, SELECT를 사용하여 FILL이 사용될 원문을 강조 표시합니다. FILL RANGE를 사용할 경우, 반드시 범위를 선택하여야 합니다.

- FILL을 사용하면 강조 표시가 취소되고 커서가 선택된 범위의 끝으로 이동됩니다.

사용자가 원문을 선택하지 않을 경우, FILL에 현재의 단락이 사용됩니다.

단락은 다음 중 하나로 구분됩니다.

- 공백 행(blank lines)
- 버퍼의 맨 위 또는 맨 아래
- 페이지 구분(page break)
- DIGITAL 표준 Runoff 명령

FILL은 단락 또는 범위의 시작 부분과 끝 부분에 있는 탭 및 공백을 제거합니다. 그러나 원문 내의 탭 및 공백에는 영향을 미치지 않습니다.

복수 바이트 문자 처리시, 공백 열(empty column)이 하나 밖에 없고 다음 문자가 복수 바이트일 경우, FILL은 행 끝 부분에 하나의 공백을 남길 수 있습니다.
FILL에서 2 바이트 공백 문자(space character)는 실제 공백(space)이 아니므로, 이는 단어 분리자(word separator)로 사용되지 않습니다.

FIND

형식

FIND (search-string)

VT300, VT200	VT100
Find	PF1

설명

지정된 문자열을 현재 버퍼에서 탐색합니다. 문자열이 발견되면 HEVE는 그 문자열 원문을 강조표시하고 커서를 해당 문자열의 시작 부분에 위치시킵니다.

바로 전에 사용된 문자열을 탐색하려면 FIND 키를 두번 눌러야 합니다. FIND 키를 한 번만 누른 후 문자열을 입력하지 않고 RETURN을 누르면, HEVE는 FIND 명령을 실행하지 않습니다.

현재 방향으로 탐색을 시작하려면, RETURN 키를 누릅니다. 특정 방향으로 탐색을 시작하려면, 방향설정 키(direction-setting key)를 누릅니다. 예를 들면, 사용자가 F11을 눌러 기능을 종결하면, 버퍼의 현재 방향의 반대 방향으로 탐색이 시작됩니다. FORWARD로 정의된 키를 누르면, 버퍼 방향과 상관없이 전진 방향(forward direction)

으로 탐색이 시작됩니다.

이 명령은 우선 사용자가 지정한 방향으로 버퍼를 탐색합니다. 해당되는 문자열이 없으면, FIND는 반대 방향으로 탐색합니다. 해당 문자열이 발견되면, 사용자에게 그 문자열로 커서를 이동시키고 싶은지 묻는 프롬프트가 표시됩니다. 그 문자열로 이동시키려면 RETURN을 누르고, 탐색을 끝내려면 NO를 입력한 후 RETURN을 누릅니다.

선택된 범위와 마찬가지로, 발견된 문자열도 강조 표시됩니다. REMOVE, STORE TEXT, LOWERCASE WORD 등의 명령이 선택된 범위에 사용될 수 있습니다. 강조 표시된 문자열 밖으로 커서를 이동시키면 강조 표시가 취소됩니다.

FIND는 복수 바이트 문자를 포함하는 문자열을 찾을 수 있도록 기능이 향상되었습니다. 탐색 문자열에는 1바이트 ASCII 문자뿐만 아니라 한글 문자도 포함시킬 수 있습니다. 더우기, SET FIND 명령을 사용하기에 따라서는, 다음 표에서와 같이 탐색 문자열에 따라 문자형과 문자크기를 지정한 경우에 FIND 명령 실행시에 문자형과 문자크기를 구분하는 기능을 기동시킬 수도 있습니다.

표 3-4 **FIND**의 문자형 및 문자크기 구분

문자형 구분여부	문자크기 구분여부	탐색 문자열	효과
구분않음	구분않음	소문자만 사용	문자형 및 문자크기 모두 구분하지 않음
구분함	구분않음	대/소문자 사용	문자형 구분함, 문자크기 구분하지 않음
구분않음	구분함	소문자만 사용	문자형 구분하지 않음, 문자크기 구분함
구분함	구분함	대/소문자 사용	문자형 및 문자크기 구분함

SET FIND 명령에 대한 설명도 함께 참조하십시오

매개변수

search-string

찾고자 하는 원문의 문자열.

탐색하고자 하는 문자열 내의 알파벳의 문자형과 문자 크기를 구분하지 않고 탐색하려면 소문자 ASCII 알파벳을 사용하십시오. 그러나, 탐색 문자열의 문자형과 문자크기를 정확하게 구분하여 탐색하려면, 탐색 ASCII 알파벳을 대,소문자를 혼합하여 사용하거나 또는 SET FIND 명령을 사용하십시오.

보다 상세한 내용은 SET FIND 명령을 참조하십시오.

FULLFORM WORD

형식

FULLFORM WORD

설명

현재 단어 또는 강조표시된 원문 내의 모든 1-바이트 문자를 전자 문자로 만듭니다. 이 기능을 사용하려면 다음과 같이 하십시오.

- SELECT 키 또는 FIND 키를 사용하여 원문을 강조하십시오.

- FULLFORM WORD 명령을 사용하십시오. 그러면 강조 표시가 제거되고, 다음 단어가 있으면 커서가 그 단어의 시작 위치로 이동됩니다.

원문에 어떠한 강조표시도 하지 않은 경우, 이 기능은 현재 단어의 알파벳을 전자 문자로 변환시킵니다. 그리고, 커서가 두 개의 단어 사이에 위치해 있는 경우에는 뒷쪽 단어에 영향을 미칩니다.

HALFFORM WORD

형식

HALFFORM WORD

설명

현재 단어 또는 강조표시된 원문 내의 모든 전자 문자를 반자 문자(즉, ASCII 알파벳)로 만듭니다. 이 기능을 사용하려면 다음과 같이 하십시오.

- SELECT 키 또는 FIND 키를 사용하여 원문을 강조 표시하십시오.

- HALFFORM WORD 명령을 사용하십시오. 그러면, 강조 표시가 제거되고, 다음 단어가 있으면 커서가 그 단어의 시작 위치로 이동됩니다.

원문에 어떠한 강조표시도 하지 않은 경우, 이 기능은 현재 단어의 알파벳을 반자 문자로 변환시킵니다. 그리고, 커서가 두 개의 단어 사이에 위치해 있는 경우에는 뒷쪽 단어에 영향을 미칩니다.

HTPU

형식

HTPU (procedure_name/statement)

설명

편집 세션(editing session) 중에 HTPU 프로시저 또는 명령문을 실행합니다. 이 기능은 EVE에서 볼 수 있는 TPU 기능과 동일합니다.

매개변수

procedure-name/statement
수행될 HTPU 프로시저 또는 명령문의 이름입니다.

예

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

위의 예는 FAO 내장 프로시저를 사용하여 현재의 날짜와 시간을 입력시키는 COPY_TEXT 내장 프로시저를 실행하는 것입니다.

LEFT ADJUST

형식

LEFT ADJUST

설명

현재 선택된 범위에서 부분적으로 또는 완전히 포함된 텍스트의 행을, 각행의 선행 공백(leading spaces)을 제거하므로써 현재 버퍼의 왼쪽 가장자리에 맞추어 이동시킵니다.

이 명령은 HEDT의 ADJL 명령과 같습니다.

LEFT INDENT

형식

LEFT INDENT

설명

현재 선택된 범위를 재형식화합니다. 즉, 행이 범위의 시작열로부터 시작하여 버퍼의 오른쪽 가장자리에서 끝나도록 하여 최대 갯수의 단어가 수용되도록 합니다. 첫번째 행을 제외하고, 선택된 범위의 각 행은 LEFT INDENT 명령을 사용하면 범위의 시작 열의 왼쪽 가장자리 값을 요구합니다.

이 기능을 사용하려면 다음과 같이 하십시오.

- LEFT INDENT 시키고자 하는 원문의 범위를 요구된 열에서 시작시키십시오.
- 원문의 범위를 선택하십시오.
- LEFT INDENT 명령을 적용하십시오.

보다 상세한 내용은 HEVE User Guide의 3장을 참조하십시오.

LOWERCASE WORD

형식

LOWERCASE WORD

설명

현재의 단어 또는 강조 표시된 원문을 소문자로 변환합니다. 이 기능을 사용하려면 :

- 옵션으로 SELECT 키 또는 FIND 키를 사용하여 원문을 강조 표시합니다.

- LOWERCASE WORD를 사용합니다. 강조 표시가 취소되고, 다음 단어가 있으면 커서가 다음 단어의 처음으로 이동됩니다.

강조 표시된 원문이 없으면, 현재의 단어가 소문자로 변환되고 커서가 두 개의 단어 사이에 위치하고 있으면 다음 단어가 소문자로 변환됩니다.
이 기능은 FULL-FORM 문자로 분류된 문자를 제외하고는 한글을 변경하지 않습니다. 이 기능은 대문자(uppercase)로 된 FULL-FORM 문자를 소문자(lowercase) FULL-FORM 형식으로 변환합니다. (CHARACTER_CLASS 내장 프로시저 참조).

MOVE BY WORD

형식

MOVE BY WORD

설명

현재 방향으로 한 번에 한 단어 만큼 커서를 이동시킵니다. 전진(forward) 방향에서는 커서가 다음 단어의 첫 문자로 이동되고, 역(reverse) 방향에서는 현재 단어의 첫 문자로 이동되거나, 이미 그 부분에 위치하고 있는 경우에는 이전 단어의 처음으로 이동됩니다. 한글에서의 단어는 같은 그룹의 문자가 연속되어 있는 것을 의미합니다(CHARACTER_CLASS 내장 프로시저 참조). 그러므로 한 번에 한 단어 만큼 커서를 이동하면 항상 다른 그룹의 새 문자로 커서가 이동되거나, 문자가 없는 경우에는 행끝으로 커서가 이동됩니다.

MOVE DOWN/UP

형식

MOVE DOWN/UP

설명

커서를 한번에 한 행씩 위/아래로 이동시키십시오. 이때, 커서는 현재 문자의 첫번째 열에 위치하여야 합니다.

화면 커서가 ON으로 설정되어 있는 경우에(HEVE에서 기본값), 내부 커서의 열 위치는 이 명령으로 변경되지 않습니다. 그러나, 화면 커서가 OFF로 설정되어 있으면 MOVE 명령이 내부 커서를 각 문자의 첫번째 열로 이동시키게 되므로, 결과적으로 MOVE 명령을 연속적으로 실행하면 커서가 계속 왼쪽으로 이동됩니다.
보다 상세한 내용은 SET [NO]DISPLAY CURSOR를 참조하십시오.

MOVE LEFT/RIGHT

형식

MOVE LEFT/RIGHT

설명

한 번에 한 자씩 명령에 지시된 방향으로 커서를 이동시킵니다. 커서가 자유로운 경우, 문자의 유무 여부에 상관없이 버퍼의 아무곳으로나, 커서를 이동시킬수 있습니다. 커서가 행의 처음에 있고 커서가 묶여있는(bound) 경우, 방향이 왼쪽이면 행의 처음에서 이전 행의 끝으로 커서가 이동됩니다. 커서가 행의 끝 부분에 있고 커서가 묶여있는 경우, 방향이 오른쪽이면 다음 행의 처음으로 커서가 이동됩니다. 커서는 항상 한글의 첫번째 바이트에 위치합니다.

REPLACE

형식

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

설명

한 문자열을 다른 문자열로 대체합니다. HEVE는 기존 문자열을 탐색하여, 해당 문자열이 발견되면 이를 강조 표시한 후 다음 후속 조치중 하나를 취하도록 기다립니다.

- YES - 발견된 문자열을 대체하고 다음 문자열을 찾습니다.
- NO - 발견된 문자열을 건너 뛰고 다음 문자열을 찾습니다.
- ALL - 해당 문자열을 모두 대체합니다(더 이상 프롬프트 되지 않음).
- LAST - 발견된 문자열을 대체한후 끝냅니다.
- QUIT - 발견된 문자열을 건너 뛴 후 끝냅니다.

YES와 ALL의 경우, 버퍼가 한 번 이상 탐색되면, 이 명령은 사용자에게 계속할 것인지를 묻습니다. 작업이 종료되면, 대체된 문자열의 수가 화면의 하단에 표시됩니다.

REPLACE 명령은 먼저, 대체될 문자열을 찾기 위하여 FIND 명령을 사용합니다. REPLACE 수행시의 SET FIND 명령의 효과는 FIND 명령을 사용할 때와 같습니다. 보다 상세한 내용은 FIND 명령을 참조하십시오.

해당 문자열이 탐색되면, 탐색된 문자열이 다음 범주 중의 하나에 속하고 또한 새로운 문자열의 알파벳 문자가 모두 소문자 및 반자일 경우에, 탐색된 문자열의 문자형과 문자 크기에 따라 대체가 수행됩니다.

- 모든 영숫자 문자가 소문자이면서 반자 문자
- 모든 영숫자 문자가 대문자이면서 반자 문자
- 모든 영숫자 문자가 소문자이면서 전자 문자
- 모든 영숫자 문자가 대문자이면서 반자 문자
- 모든 영숫자 문자가 소문자이면서 반자 문자. 이 경우, 첫번째 문자가 대문자이면서 반자인 경우는 제외.
- 모든 영숫자 문자가 소문자이면서 전자 문자. 이 경우,첫번째 문자가 대문자이면서 전자인 경우는 제외.

HEVE는 대체 문자열을 대체되는 문자열의 문자형과 문자 크기를 유지하면서 현재 버퍼에 넣습니다. 그렇지 않으면,대체 문자열의 문자형과 문자 크기가 새로운 문자열 매개 변수에 제공된 것과 꼭 같게 됩니다. SET FIND 명령의 설명도 함께 참조하십시오.

매개변수

old-string
제거할 원문. 이것은 1바이트 문자와 복수 바이트 문자로 이루어집니다.

new-string
old-string을 대체하는데 사용할 원문. 이것은 1바이트 문자와 복수 바이트 문자로 이루어집니다.

예

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

위의 예는 "Wrod"라고 쓰인 모든 문자열을 문자열 "Word"로 대체합니다. 8개의 문자열이 대체되었습니다.
여러 개의 단어로 된 구절(phrase)을 대체하려면, 다음의 예와 같이 문자열을 따옴표로 묶습니다 .

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

RIGHT ADJUST

형식

RIGHT ADJUST

설명

현재 선택된 범위에 부분적으로 또는 완전히 포함된 원문의 행을, 행에 선행 공백을 삽입하여 오른쪽 가장자리에 맞추어 이동시킵니다.

이 명령은 HEDT의 ADJR 명령과 같습니다.

RIGHT INDENT

형식

RIGHT INDENT

설명

현재 선택된 범위를 재형식화하므로써, 행이 범위의 첫번째 열에서 시작하여 버퍼의 오른쪽 가장자리에서 끝나도록 하여 최대 갯수의 단어를 수용되게 합니다. 더우기, RIGHT INDENT는 마지막 행을 제외하고는 공백을 삽입하여 각 행이 오른쪽으로 정렬되도록 합니다. RIGHT INDENT를 사용한 후에, 선택된 범위의 각 행은 범위의 시작 열의 왼쪽 가장자리 값을 취하고 버퍼의 오른쪽 가장자리에서 끝납니다.

이 명령은 HEDT의 NDTR 명령과 같습니다.

SAVE EXTENDED HEVE

형식

SAVE EXTENDED HEVE (section-filespec)

설명

앞으로의 편집 작업을 위해 섹션 파일에 있는 현재의 키 정의 및 기타 확장 사항(extension)을 보관합니다. SAVE EXTENDED HTPU는 이 명령과 동일합니다.

섹션 파일은 누적됩니다. 사용자는 키 정의 및 사용자가 사용하고 있는 섹션 파일에 이미 보관된 정의를 포함해 편집 작업 중에 수행된 확장 사항을 보관할 수 있습니다. 그러므로 섹션 파일을 작성하면 사용자 전용의 HEVE 버전을 만들 수 있습니다. 그러나 섹션 파일은 대개 가장자리 설정, 커서 설정 또는 탭 멈춤을 보관하지 않습니다. 사용자별로 이들의 기본 값(default value)을 설정하려면 초기설정 파일을 이용하면 됩니다.

매개변수

section-filespec
사용자가 작성하고자 하는 섹션 파일. 기본 파일 유형(default file type)은 TPU$SECTION입니다.

예

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

위의 예는 MYEVE.TPU$SECTION이라 불리는 섹션 파일을 만드는 것입니다.

SAVE EXTENDED HTPU

형식

SAVE EXTENDED HTPU (section-filespec)

설명

앞으로의 편집 작업을 위해 섹션 파일에 있는 현재의 키 정의 및 기타 확장 사항(extension)을 보관합니다. SAVE EXTENDED HEVE는 이 명령과 동일합니다.

섹션 파일은 누적됩니다. 사용자는 키 정의 및 사용자가 사용하고 있는 섹션 파일에 이미 보관된 정의를 포함해 편집 작업 중에 수행된 확장 사항을 보관할 수 있습니다. 그러므로 섹션 파일을 작성하면 사용자 전용의 HEVE 버전을 만들 수 있습니다. 그러나 섹션 파일은 대개 가장자리 설정, 커서 설정 또는 탭 멈춤을 보관하지 않습니다. 사용자별로 이들의 기본 값(default value)을 설정하려면 초기설정 파일을 이용하면 됩니다.

매개변수

section-filespec
사용자가 작성하고자 하는 섹션 파일. 기본 파일 유형(default file type)은 TPU$SECTION입니다.

예

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

위의 예는 MYEVE.TPU$SECTION이라 불리는 섹션 파일을 만드는 것입니다.

SET [NO]DISPLAY CURSOR

형식

SET [NO]DISPLAY CURSOR

설명

이 명령은 HTPU의 화면 커서 모드를 ON 또는 OFF 시킵니다.

화면 커서 모드가 ON 상태인 경우에는, 화면에 보이는 커서 (화면 커서라 함)가 복수 바이트 문자에 커서가 현재 있으면 HTPU에 저장되어 있는 것과 다를 수 있습니다.

예를 들어, 내부 커서가 복수 바이트 문자의 두번째 바이트에 위치해 있을 경우 화면 커서 모드가 OFF 상태이면 화면 커서가 문자의 두번째 열에 위치하고, 화면 커서 모드가 ON 상태이면 화면 커서가 문자의 첫번째 열에 위치합니다.

화면 커서 모드는 화면 커서를 항상 문자에 묶이게 합니다.
화면 커서 모드가 ON이고 내부 커서가 복수 바이트 문자의 첫번째 이외의 바이트에 위치해 있으면, HTPU와 HEVE에서 문자의 삽입과 겹쳐쓰기는 마치 내부 커서가 화면 커서의 위치에 있을 때처럼 수행됩니다. 이것은 커서가 문자에 묶여 있는 경우에는 항상 그렇습

니다.

기본 값은 SET DISPLAY CURSOR입니다.

SET FIND CASE [NO]EXACT

형식

SET FIND CASE [NO]EXACT

설명

이 명령은 탐색 또는 대체될 문자열의 모든 알파벳 문자가 소문자인 경우에 FIND와 REPLACE 명령 수행시에 문자크기는 구분하지 않고 문자형만을 구분하게 할 것인지를 설정합니다. 문자크기의 구분은 SET FIND FORM [NO]EXACT 명령에 의해 결정됩니다.

SET FIND CASE EXACT는 문자형 구분 기능을 ON 시킵니다.
SET FIND CASE NOEXACT는 문자형 구분 기능을 OFF 시킵니다.

기본 값은 SET FIND CASE NOEXACT 입니다.

SET FIND FORM [NO]EXACT

형식

SET FIND FORM [NO]EXACT

설명

이 명령은 FIND와 REPLACE 명령 수행시에 문자크기를 구분(즉, 전자/반자 문자 구분)할 것인지를 설정합니다. 문자형은 SET FIND CASE [NO]EXACT 명령에 의해 결정됩니다.

SET FIND FORM EXACT는 문자크기 구분 기능을 ON 시킵니다.
SET FIND FORM NOEXACT는 문자크기 구분 기능을 OFF 시킵니다.

기본 값은 SET FIND FORM NOEXACT 입니다.

SET FIND GENERAL

형식

SET FIND GENERAL

설명

이 명령은 SET FIND FORM NOEXACT와 함께 SET FIND CASE NOEXACT와 같습니다.

SET FIND CASE [NO]EXACT 및 SET FIND FORM [NO]EXACT도 함께 참조하십시오.

SET FIND EXACT

형식

SET FIND EXACT

설명

이 명령은 SET FIND FORM EXACT와 함께 SET FIND CASE EXACT와 같습니다.

SET FIND CASE [NO]EXACT 및 SET FIND FORM [NO]EXACT도 함께 참조하십시오.

SET TABS (MOVEMENT)

형식

SET TABS (MOVEMENT)

설명

TAB 키를 누를 때마다 커서가 다음 탭 멈춤으로 이동되도록 설정합니다. 다음 탭 멈춤이 복수 바이트 문자 내에 있는 경우, 커서가 TAB 멈춤 위치 뒤에 있는 문자의 처음으로 이동됩니다. 그러므로 커서는 복수 바이트 문자 내에 위치되지 않고 항상 문자의 처음에 위치하게 됩니다.

SPECIAL INSERT

형식

GOLD/[integer]/GOLD + KP3

설명

ASCII 값을 사용해 문자를 삽입합니다. 이 명령은 127보다 큰 값이 입력되지 않도록 수정되었습니다. 그러므로 복수 바이트 문자의 일부가 원문에 입력될 수 없습니다.

예

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

위의 예는 버퍼 내에 커서의 현재 위치에 LF(linefeed) 문자를 삽입합니다.

SYMBOL

형식

SYMBOL (text-string)

설명

이 명령은 런 타임 라이브러리 HSYLIB가 제공하는 기능 JSY$TRA_SYMBOL을 사용하여 선 및 사각형을 그릴 수 있으며 HEVE에만 있습니다. 이 명령 호출시 사용자가 문자열을 제공하지 않으면, 키보드로는 엑세스하기 어려운, 한글에 있는 이용가능한 모든 기호를 표시하여 사용자가 선택하도록 합니다. 각각의 기호는 특수한 연속된 ASCII 문자에 사상(map)됩니다. 프롬프트에 특수한 연속된 ASCII 문자를 입력하면, 이에 대응되는 특수 기호가 원문의 현재 편집 위치에 삽입됩니다.

기호에 사상되지 못하는 특수한 연속 ASCII 문자를 입력하면, 이에 사상되는 기호가 없으므로 원문의 현재 편집 위치에 기호가 표시되지 않고, 입력된 ASCII 문자들이 삽입됩니다.

매개변수

text-string
원하는 기호를 얻기 위한 특수 ASCII 문자를 포함하는 원문 문자열. 대응되는 기호가 없는 특수 ASCII 문자가 지정되었을 경우, 그 문자는 번역(translate)되지 않은채 번역된 기호와 함께 원문에 삽입됩니다.

예

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

위의 예는 폭이 40컬럼인 사각형의 위 부분을 그립니다.

UPPERCASE WORD

형식

UPPERCASE WORD

설명

현재의 단어 또는 강조 표시된 원문을 대문자로 변환합니다.
이 명령을 사용하려면,

- 옵션으로 SELECT 키 또는 FIND 키를 사용하여 원문을 강조 표시합니다.
- UPPERCASE WORD를 사용합니다. 강조 표시가 취소되고 다음 단어가 있는 경우, 커서가 다음 단어의 처음으로 이동됩니다.

강조 표시된 원문이 없으면, 현재의 단어가 대문자로 변환되고, 커서가 두 개의 단어 사이에 있으면 다음 단어가 대문자로 변환됩니다.

대개 이 기능은 FULL_FORM 문자로 분류된 문자를 제외하고는 한글을 변경하지 않습니다. 이 명령은 소문자 형식의 FULL_FORM을 대문자 FULL_FORM 형식으로 변환합니다
(CHARACTER_CLASS 내장 프로시저 참조).

부록 A

HTPU와 HEVE의 제한 사항

ASCII 내장 프로시저에서, 지정된 정수값 매개변수가 166과 255 사이에 있으면 ASCII는 DECTPU 의 다국 문자 세트에 있는 문자를 복귀합니다. 그러나 HTPU에서는 그렇지 않습니다. HTPU는 복귀된 문자를 2-바이트 문자 부분으로 취급합니다. 그러므로 사용자는 ASCII 내장 프로시저로부터 복귀된 문자를 화면에 표시하는 COPY_TEXT 같은 명령어를 사용하지 않기를 바랍니다.

HTPU에서 FAO 내장 프로시저는 복수 바이트 문자를 처리할 수 없습니다. 복수 바이트 문자가 FAO의 문자열 매개변수로 포함될 경우 그 결과는 예측할 수 없습니다.

HTPU에 있는 READ_LINE 내장 프로시저는 복수 바이트 문자를 올바로 읽지 못합니다. 복수 바이트 문자를 입력하면, READ_LINE이 종료되고, 종료 전에 입력된 문자가 복귀됩니다. 복수 바이트 문자는 복귀될 수 없습니다.

HTPU의 TRANSLATE는 복수 바이트 문자를 처리할 수 없습니다. 복수 바이트 문자가 TRANSLATE의 매개변수로 포함될 경우, 결과를 예측할 수 없습니다.

HEVE의 WILDCARD FIND는 \+ 기호의 사용을 지원하지 못합니다. 왜냐하면 이 기호는 128부터 255 사이의 ASCII 값을 갖는 1-바이트 문자용으로만 사용되기 때문입니다.
EVE$BUILD는 본 버전의 HTPU와 HEVE에서는 지원되지 않습니다.

부록 B

추가 HTPU와 HEVE 메시지

이 부록은 HTPU와 HEVE의 추가 메시지를 수록하고 있습니다.

아래의 표에 해당 메시지가 호출되는 오류의 설명문과 조치 사항이 표시되어 있습니다.

다른 메시지들에 대해서는 DEC Text Processing Utility Reference Manual을 참조하십시오.

B.1 HTPU 메시지

표 **B-1 HTPU** 메시지 및 심각도 레벨

축약어	메시지	심각도 레벨
RIMP_NOT_EXIST	입력 방법에 대한 RIMP가 DECwindows 서버에 없음.	정보